Models, To Model, and Modelling
Towards a Theory of Conceptual Models and Modelling
Towards a Notion of the Model

Collection of Recent Papers

Bernhard Thalheim
Christian-Albrechts-University Kiel, Department of Computer Science, 24098 Kiel, Germany
thalheim@is.informatik.uni-kiel.de

Content of this collection

   Preliminary version: Lecture Notes in Computer Science 5833, [Tha09]
conceptual modelling, [Tha11b] ................................................................................................................ 58-93
4. The Science and Art of Conceptual Modelling. TLDKS VI, LNCS 7600, [Tha12b] ........................................... 94-123
   Preliminary version: Lecture Notes in Computer Science 6860, [Tha11a]
6. The Definition of the (Conceptual) Model. EJC’13, [Tha13a,Tha14] ......................................................... 135-148
7. Das Modell des Modelles. EWE’15 [Tha15] .......................................................... 149-151
8. Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung.
   The main book on the Kiel approach to models: [TN15].
10. A Conceptual Model for Services. CMS’15@ER’15 [TD15] ......................................................... 164-174

References

   http://www.jucs.org/jucs_16_20/towards_a_theory_of.
[Tha11b] B. Thalheim. The theory of conceptual models, the theory of conceptual modelling and foundations of conceptual modelling. In The
[Tha12c] B. Thalheim. Syntax, semantics and pragmatics of conceptual modelling. In NLDB, volume 7337 of Lecture Notes in Computer Science,
[Tha14] B. Thalheim. The conceptual model ≡ an adequate and dependable artifact enhanced by concepts. In Information Modelling and Knowl-
   2015.
   2015. Springer.
Towards a Theory of Conceptual Modelling

Bernhard Thalheim
(Christian Albrechts University Kiel, Department of Computer Science
Olshausenstr. 40, D-24098 Kiel, Germany
thalheim@is.informatik.uni-kiel.de)

Abstract: Conceptual modelling is a widely applied practice and has led to a large body of knowledge on constructs that might be used for modelling and on methods that might be useful for modelling. It is commonly accepted that database application development is based on conceptual modelling. It is however surprising that only very few publications have been published on a theory of conceptual modelling. Modelling is typically supported by languages that are well-founded and easy to apply for the description of the application domain, the requirements and the system solution. It is thus based on a theory of modelling constructs. At the same time, modelling incorporates a description of the application domain and a prescription of requirements for supporting systems. It is thus based on methods of application domain gathering. Modelling is also an engineering activity with engineering steps and engineering results. It is thus engineering. The first facet of modelling has led to a huge body of knowledge. The second facet is considered from time to time in the scientific literature. The third facet is underexposed in the scientific literature.

This paper aims in developing principles of conceptual modelling. They cover modelling constructs as well as modelling activities as well as modelling properties. We first clarify the notion of conceptual modelling. Principles of modelling may be applied and accepted or not by the modeler. Based on these principles we can derive a theory of conceptual modelling that combines foundations of modelling constructs, application capture and engineering.

A general theory of conceptual modelling is far too comprehensive and far too complex. It is not yet visible how such a theory can be developed. This paper therefore aims in introducing a framework and an approach to a general theory of conceptual modelling. We are however in urgent need of such a theory. We are sure that this theory can be developed and use this paper for the introduction of the main ingredients of this theory.

Key Words: modelling, conceptual modelling, modelling act(ivity), principles of models and modelling, general theory of models
Category: H.2.1, H.2.2, H.1.0, M.4, I.6.5, I.6.4, H.0, L.1.0

1 Introduction

The main purpose of conceptual modelling is classically understood as the elicitation [Chen et al.(1998); Olivé(2007); Thalheim(2000)] of a high-quality conceptual schema of a (software, information, workflow, ...) system. This understanding mainly concentrates on the result of conceptual modelling and hinders the development of a general theory of conceptual modelling. Modelling is based on languages which might be sophisticated [Chen et al.(1998)] and well understood [Thalheim(2000)] like the ER modelling language or might be fuzzy with
lazy semantics development like the UML. Let us analyse the complexity of the modelling task and then let us draw some conclusions for the modelling “act”.

1.1 The Three Dimensions of Conceptual Modelling

Conceptual modelling is often only discussed on the basis of modelling constructs and illustrated by some small examples. It has however three fundamental dimensions:

1. **Modelling language constructs** are applied during conceptual modelling. Their syntactics, semantics and pragmatics must be well understood.

2. **Application domain gathering** allows to understand the problems to be solved, the opportunities of solutions for a system, and the requirements and architecture that might be prescribed for the solution that has been chosen.

3. **Engineering** is oriented towards encapsulation of experiences with design problems pared down to a manageable scale.

The first dimension is handled and well understood in the literature. Except few publications, e.g. [Bjørner(2009)], the second dimension has not yet got a sophisticated and well understood support. The third dimension has received much attention by data modelers [Simsion(2007)] but did not get through to research literature. It must therefore be our goal to combine the three dimensions into a holistic framework.

1.2 Alternatives for a Notion of a Theory

The notion of a theory itself is be a matter of intensive research [Deppert(2009); Mittelstraß(2004)]. We base our understanding of the notion of a theory on the three dimensions and the main goal of conceptual modelling. This understanding is covered by the understanding of the notions of a theory\(^1\).

The classical treatment of the notion of a theory is based on mathematical and philosophical logics and is far too strict. We may however inherit certain elements of such logics. Already the notion of semantics provides a larger number of choices [Schewe and Thalheim(2008)] beyond those that are taken for granted

\(^1\) Websters dictionary [Web(1991)] distinguishes several understandings of the notion of theory: 1: the analysis of a set of facts in their relation to one another; 2: abstract thought; speculation; 3: the general or abstract principles of a body of fact, a science, or an art; 4a: a belief, policy, or procedure proposed or followed as the basis of action; 4b: an ideal or hypothetical set of facts, principles, or circumstances; 5: a plausible or scientifically acceptable general principle or body of principles offered to explain phenomena; 6a: a hypothesis assumed for the sake of argument or investigation 6b: an unproved assumption; conjecture 6c: a body of theorems presenting a concise systematic view of a subject; hypothesis
in Computer Science. The notion of a theory is based on a theory of truth that is based on a notion of truth and on a number of supporting theories such as a correspondence theory for truth, a coherence theory for truth, and a consensus theory for truth.

Theories of conceptual modelling must step beyond axiomatic and analytical theories. They must also be operational and ‘genetic’. Theories of conceptual modelling can be developed in the frameworks of logical empiricism, of context theories (‘context of use’, ‘language game’), and of constructivism. The first framework supports to define purposes of conceptual modelling, to emphasise threats that should be handled with the help of models, to select appropriate modelling languages and methods, to reason on the quality of the model depending on the purpose of the model, to select measures for the quality of models, and to guide the process of modelling. It may use development experiments, case studies, evaluation surveys, and implementation studies. The second framework relates models to the application domain, to the stakeholders participating in the development process, to the aspects reflected within a model, and to the resources provided either by the system and by the knowledge from the application domain. It requires to base conceptual modelling on application domain theories. The last framework provides a basis for a general structure by a language of constructs that can be applied for the development of a model, a set of constructors that can be applied to combine models into a new model, and a number of quality properties for characterisation of usage of certain constructs.

1.3 Implications for a Theory of Conceptual Modelling

The three dimensions of conceptual modelling must be integrated into a framework that supports the relevant dimension depending on the modelling work progress. The currently most difficult dimension is the engineering dimension. Engineering is inherently concerned with failures of construction, with incompleteness both in specification and in coverage of the application domain, with compromises for all quality dimensions, and with problems of technologies currently at hand.

At the same time, there is no universal approach and no universal language that cover all aspects of an application, that have a well-founded semantics for all constructions, that reflect any relevant facet in applications, and that support engineering. The choice of modelling languages is often a matter of preferences and case, empirical usage, evolution history, and supporting technology. Models are at different levels of abstraction and particularisation. We therefore are going to develop a number of different models that reflect different aspects of the system that is under development. [Thalheim(2009)] introduces model suites as a set of models with explicit associations among the models, with explicit controllers for maintenance of coherence of the models, with application schemata for their
explicit maintenance and evolution, and tracers for establishment of their coherence. Model suites and multi-model specification increases the complexity of modelling. Interdependencies among models must be given in an explicit form. Models that are used to specify different views of the same problem or application must be used consistently in an integrated form. Changes within one model must be propagated to all dependent models. Each singleton model must have a well-defined semantics as well as a number of representations for display of model content. The representation and the model must be tightly coupled. Changes within any model must either be refinements of previous models or explicit revisions of such models. The change management must support rollback to earlier versions of the model suite. Model suites may have different versions.

1.4 Quality Assessment, Control and Improvement

According to [Kangassalo(2007)] the result of conceptual modelling depends on information available about the UoD; on information about the UoD, regarded as not relevant for the concept or conceptual model at hands, and therefore abandoned or renounced; on the philosophical background to be applied in the modelling work; on the additional knowledge included by the modeler, e.g. knowledge primitives, conceptual ‘components’, selected logical or mathematical presuppositions, mathematical structures, etc.; on the collection of problems that may be investigated in this environment; on the ontology (or better language with its lexical semantics [Schewe and Thalheim(2008)]) used as a basis of the conceptualization process; on the epistemological theory, which directs how ontology should be applied in recognizing and formulating concepts, conceptual models or theories, and in constructing information, data, and knowledge, on different levels of abstraction; on the purpose and goal of the conceptual modelling work; on the collection of methods for conceptual modelling; on the process of the practical concept formation and modelling work; and finally on the knowledge and skill of the person making modelling, as well as those of the people giving information for the modelling work.

Quality properties are

- static qualities of models such as the development quality (pervasiveness, analysability, changeability, stability, testability, privacy of the models, ubiquity, development parsimony), internal quality (accuracy, suitability, interoperability, robustness, self-contained, independence, internal parsimony), and quality of use (understandability, learnability, operability, attractiveness, appropriateness, user parsimony), and

- dynamic qualities within a selected development approach (executability, refinement quality, scope restriction, effect preservation, context explicitness, completion tracking, resource parsimony).
The information system modelling process is intentionally or explicitly ruled by a number of development strategies, development steps, and development policies. These quality properties may alternatively be grouped into correctness, generality, usefulness, comprehensibility, and novelty. Models have very different purposes. Therefore, assessment of quality may vary a lot. Figure 1 displays the general view, a specific portfolio of quality requirements for realisation, and another one for the communication purpose of a model. It shows that quality considerations are not equally relevant during development and modelling.

We may concentrate on some of these properties. Our first choice for quality properties that drive other quality properties are an explicit statement about the
applicability of the concept, modality of the concept for the model, and confidence of the concept inclusion into the model.

1.5 Outline and Tasks of the Paper

The three dimensions of conceptual modelling within a constructive, empirical, contextual, and quality framework lead directly to a separation of concern into a theory of modelling constructs, a theory of modelling activities, a theory of modelling properties, a theory of application domain reflections, and a theory of engineering. These theories can be support by other theories such as a theory of model management and a generalised activity theory such as a theory of modelling styles and pattern. These theories must have their constituents.

The paper aims in introducing a theory of conceptual modelling. The paper is directly structured by these theories. Section 2 discusses the matter of the choice of modelling constructs and model constructors. We use one example that demonstrates the impact of wrong choices. Section 3 provides an insight into different actions, activities and general tactics and strategies that can be used during conceptual for modelling. Section 4 discusses whether we can enhance conceptual modelling by properties. Finally, Section 5 summarises the achievements of this paper and derives a research agenda for a theory of conceptual modelling.

The paper targets on a general understanding of the field on conceptual modelling. Figure 2 displays a general structuring of this field into strategic, tactical and tool support layers.

It has not been our intention to survey the modelling literature\(^2\). This would be far too large already for the conceptual modelling research.

This paper also extends approaches of “design science” [Hevner et al.(2004)] that relates models to its purposes. Modelling creates and evaluates models intended to solve identified organisational problems. The process of modelling enables stakeholders to understand both the problem addressed by the model and the feasibility of the model to the problem solution. The theory of conceptual modelling also aims in both the study of the modelling process and the models themselves from one side and the development of methods to enhance the body of knowledge developed for a theory of modelling from the other side. This paper is thus only a first try for a general theory of conceptual modelling.

\(^{2}\) Good sources to this research are [Chen et al.(1998); Olivé(2007); Thalheim(2000)].
Towards a Theory of Modelling Constructs

2.1 The Notion of Model in Science Theory and in Information Systems Development

Information system models are typically representations (how specified) of certain application solutions (origin, whereof) for a community of users (whom), for certain application goals and intentions (for what), within a certain time span (when), and with certain restrictions (normal, exception and forbidden cases). Therefore, information systems models are a corrected, rectified, regimented, and in many instance idealised version of the application domain we gain from immediate observation of the application domain. Characteristically, one first eliminates flaws and errors in the application domain and then present the concepts in a ‘neat’ way. These two steps are commonly referred to as concept reduction and application fitting. Models are thus vehicles for learning about the application. Once we have knowledge about the model, this knowledge is
translated into knowledge about the application domain.

A model represents subjects or things
· based on an analogy of structuring, functionality, or behaviour,
· considering certain application purposes, and
· providing a simple handling or service or consideration of the things under consideration.

The model definition given is one [Stachowiak(1992)] of many options. A model has typically a model capacity:
· the model provides some understanding of the original;
· the model provides an explanation of demonstration through auxiliary information and thus makes original subject easier or better to understand;
· the model provides an indication and facilities for making properties viewable;
· the model allows to provide variations and support optimisation;
· the model support verification of hypotheses within a limited scope;
· the model supports construction of technical artifacts;
· the model supports control of things in reality;
· the model allows a replacement of things of reality and acts as a mediating means.

Typically these functions are used simultaneously and in competition. Therefore to model means different activities at the same time: to plan or form after a pattern or shape, to make into an organization (as an army, government, or parish), to produce a representation or simulation to model a problem, and to construct or fashion in imitation of a particular model.

This competition of meanings results in a number of problems of conceptual modelling such as competing attitudes and profiles of modelers, varieties of styles of specification, multi-model reasoning, and integration into a general coherent model ensemble (or model suite). Therefore we face the “grand challenge of harmonisation”.

Models are typically given by the triple (original, image mapping) that is extended by properties of the image, of the mapping, of the system under consideration, that are based on a common modelling “culture” or understanding, and depends on the aim of the model. Therefore we envision that modelling can be considered as an art similar to the ‘art of programming’.

### 2.2 A Theory of Modelling Concepts

Concepts are the basis for conceptual modelling. They specify our knowledge what things are there and what properties things have. They typically represent categories, i.e., sets of abstract or concrete entities that share a set of common properties. Concepts are used in everyday life as a communication vehicle and as a reasoning chunk. Concepts can be based on definitions of different kinds.
Modelling concepts must be represented through representations such as symbols, icons, annotations, ontology units or topics. They are based on annotations given in some language. The binding of concepts to their representations cannot be strict. Consider, for instance, the annotation ‘bridge’ and its dozen of meanings. Typically, the binding between concepts and representations remains to be stable over a longer period of time. It depends however on context within the application domain for certain users or user groups. We therefore bind representations \( r \) to their meaning or their concepts \( \text{concept}(r, g, w) \) within a world \( w \) for a group \( g \).

Modelling judgements
\[
(r, t, (a, m, c), g, w)
\]
are elements of certain languages
- for the representation \( r \)
- of things \( t \) under consideration,
- with restrictions for their applicability \( a \), with a rigidity or modality \( m \), with a confidence \( c \) on their validity,
- based on a common understanding of a group \( g \) within their world \( w \) or culture or application domain context.

These languages are typically well-structured. For instance, representations can be based entirely on the extended ER model [Thalheim(2000)].

The relation among modelling judgements may be based on entailment relations (e.g., material or strict logical implications), contrariety (e.g., exclusion constraints among properties), contradiction (e.g., existence exclusion), and independence (e.g., interaction-free interpretation) beyond the fundamental structural relations discussed below [Albrecht et al.(1998)].

Judgements of models are additionally characterised by their quantity (universal, particular, singular), their quality (affirmative, negative, infinite), their relation (categorical, hypothetical, disjunctive), and their modality (problematical, assertorical, apodictical) [Deppert(2009)]. We may limit our characterisation to modality.

A concept has typically a manyfold of definitions. Their utilisation, exploration and application depend on the user (e.g. the education profile), the usage, and context. Concepts typically also depend on the application context, i.e. the application area and the application schema. The association itself must be characterised by the kind of association. Concepts are typically hierarchically ordered and can thus be layered. We assume that this ordering is strictly hierarchical and the concept space can be depicted by a set of concept trees. A concept is also dependent on the community that prefers this concept. A concept is also
typically given through an embedding into the application domain and into the knowledge space.

The main part of our concept definition is a tree-structured structural expression of the following form

\[
\text{SpecOrderedTree}(\text{StructuralTreeExpression} \quad (\text{DefinitionItem}, \text{Modality}(\text{ Sufficiency, Necessity}), \\
\text{ Fuzziness, Importance, Rigidity,} \\
\text{Relevance, GraduationWithinExpression, Category})) ) .
\]

This general understanding allows a number of approaches to modelling such as the styles considered below (e.g., Russian doll, Venetian, ...; see below) or the collection style that collects all quadruples without any additional structuring or layering. If in the application domain things can be categorized we might use this categorisation also for a general skeleton with a model.

We may therefore restrict the theory of models to a set of judgements \((r, t, (a, m, c), g, w)\) and the meaning of these representations \(\text{concept}(r, g, w)\). Representations as well as concepts are expressions within a language that is appropriate for the model purpose. Consequently we need to define these languages in a flexible and expressive way.

2.3 The Fundamental Structural Relations

The five fundamental structural relations used for construction abstraction are aggregation/participation, generalisation/specialisation, exhibition/characterisation, classification/instantiation, and separation between introduction and utilisation.

Aggregation (agglomeration)/participation characterizing which object consists of which object or resp. which object is part of which object. Aggregation is based on constructors such as sets, lists, multisets, trees, graphs, products etc. It may include naming. Generalization/specialization characterizing which object generalizes which object or resp. which object specializes which object. Hierarchies may be defined through different classifications and taxonomies. So, we may have a different hierarchy for each point of view. Hierarchies are built based on inheritance assumptions. So, we may differentiate between generalization and specialization in dependence on whether characterization are not or are inherited and on whether transformation are or are not applicable. Exhibition/characterization specifying which object exhibits which object or resp. which object is characterized by which object. Exhibitions may be multi-valued depending of the data type used. They may be qualitative or quantitative. Classification/instantiation characterizing which object classifies which object or resp. which object is an instance of which object. Introduction/utilisation allows to distinguish between an introduction of an object, the shared or exclusive utilisation and the finalisation of an object.
2.4 Building Principles for Modelling Languages

Models are expressions in a modelling language. The language itself may be built on principles such as the following ones:

**Compositionality** supports combination of models. Given any two models \( M_1, M_2 \) combined into a complex one \( M_1 \oplus M_2 \) (for any composition operator \( \oplus \) of the language syntax), the semantics of \( M_1 \oplus M_2 \) is defined by \( \text{Sem}(M_1 \oplus M_2) = \text{Sem}(M_1) \oplus \text{Sem}(M_2) \).

**Inductivity** is typically based on inductive construction of expressions in a language. For instance, ER logics separates attribute, entity, relationship, and cluster types and uses also some concept of variables for a layered construction of models.

**Separation of characterisation and coexistence** scopes the attention either to the properties of an object itself or to the associations of the object (or things) to other objects or things.

**Separation of introduction and co-use** allows to distinguish for the CRUD (Create-Read-Update-Delete) lifecycle of objects between the CUD features and the R features for the object itself from one side and the co-use (through R) or referencing mechanism that co-use these objects form the other side. Entity types are used to introduce and to structure objects and relationship or cluster types reference and co-use these objects. Reference is typically based on some concept of identification (tuple identifier, key value for one of the keys, identifier as surrogate or artificial construct) [Beeri and Thalheim(1999)].

**Context-free expression construction** implies the coincidence theorem and allows to limit consideration of language expressions only on the expression itself.

These principles are typically taken for granted in formal languages. They are neither naturally given in an application nor generally achievable within a modelling process. For instance, natural languages use idioms that support clustering and encapsulation, noun compounds that allow to combine nouns into a singleton one, implicit active zones that allow to tighten meaning of constructs, and complex categories with prototype semantics [Schewe and Thalheim(2008)]. From the other side, these principles are very powerful and useful. Inductivity and context-freeness allow to manage model constructs in separate.

Typical pragmatic assumptions applied to conceptual models are the unique name assumption, unique flavour assumption, existence of full-fledged domains, non-triviality of structures of associations, strict hierarchical structures, and non-triviality of identification.

A typical example of principles is the set of principles used for the extended entity-relationship model: inductivity (updates are essentially atomic),
compositionality for any type construction pragmatic assumptions such as the usage of names through noun as standard markers, closed schemata, context-free specifications, canonical semantics (e.g. sets instead of multisets, ...), value-identifiability of objects, and restrictions such as limiting computation to functions of low computational complexity.

2.5 Inductive and Abstraction Layered Typed Modelling Constructs

Typically, a model is defined in a certain language. A model language \( \mathcal{L} \) for a model uses some signature \( \mathcal{S} \) and a set of constructors \( \mathcal{C} \) that allows to build a set of all possible expressions in this language. Typically constructors are defined by structural recursion [Thalheim(2000)]. The set of constructors may allow to build expressions that do not fulfill certain quality or more generally integrity conditions. Therefore we introduce a set \( \Sigma_{\mathcal{S}, \mathcal{C}} \) well-formedness conditions.

Well-formedness conditions separate ‘normal’ expressions from ‘abnormal’ ones. The latter can be separated into construction abnormality and semantic abnormality. We may allow such abnormalities that are corrigible to normal expressions. The avoidance of abnormality is still research in progress. Kinds of abnormality that should be handled within a theory of conceptual modelling are pleonasm (e.g., redundancy), semantic clashes (e.g., contradictions), Zeugma (e.g., overloading of constructs, combining separable semantic units into one concept), and improbability (e.g., almost empty classes).

Well-formedness restrictions influence the modelling style [Klettke(2007)].

– The Strong Venetian style rigidly separates basic constructs and builds a fully compositional structuring. ER schemata and UML class diagrams are typically based on this style.

– The Weak Venetian style separates constructs to same degree but not more than it is necessary. Performance-tuned physical relational schemata are typically based on this style.

– The Strong Russian Doll style is based on a full expansion of objects, i.e. objects in a database are potentially expandable through navigational substructures.

– The Weak Russian Doll style uses a layered representation similar to tree languages.

ER modelling is typically based on the Salami slice style whereas XML modelling typically uses the strong Russian doll (DTD style) or the weak Venetian or weak Russian doll (XML schema) style. The weak Venetian blind style is also the basis for component-based development of models since amalgams constructs as small models of coexisting and coevolving facets of objects.
A model type $T_{L_S} = (L_S, \Sigma_{L_S})$ is defined by a pair consisting of the language of the model and by constraints $\Sigma_{L_S} \in L(\Sigma_{WellFormed})$ applicable to all models defined in the given language.

Model languages $L_{S_1}, ..., L_{S_n}$ may be bound to each other by partial mappings $R_{i,j} : L_{S_i} \rightarrow L_{S_j}$ based on their signatures. These mapping typically define the association of elements among the languages.

A model is based on an expression in the given language. Typically, it has a structure definition, a semantics definition, and a pragmatics definition. Semantics restricts the models we are interested in. Pragmatics restricts the scope of the users of models. We explicitly define a model $M$ by an expression $struct_M$ in a language $L_S$ that obeys $\Sigma_{L_S}$, by a set of constraints $\Sigma_M$ defined in the logics of this language. Therefore, each model has its model type. We denote by $M_T$ or $M_i$ for some $i$ the set of all models of this type.

2.6 Coexistence of Equivalent Models

Models support a number of purposes such as construction of systems, communication, analysis, examination and check, documentation, mastering of application complexity, improvement, evolution and realisation and construction. For instance, we can distinguish construction models that are product-focussed and business user models that are use-focussed. The last kind also refers to the broader societal context in which the information system is going to be used beside the interaction of the business user with the information system. We might overburden a model in order to satisfy all these purposes. Instead, we better use a number of models for each of its purposes. These models are then bound to each other. The binding may be implicit or explicit. Implicit binding may lead to incoherence. Therefore, we shall better request a coherent coevolution and coexistence of models. This coexistence may either be based

- on model suites or ensembles that use an explicit association among the constructs of the models or

- on global and specific models that are based on a global model which combines all aspects of the specific models and an ensemble of specific models which are views of the global model and which reflect the different purposes.

The first approach seems to be the most appropriate one since a stakeholder needs a model on the appropriate abstraction level that is not overburden by any unnecessary detail. We have been introducing the notion of a model suite for the support of this approach [Thalheim(2009)].

The later approach to model coexistence may be supported by two approaches:
Global model as combined view of the specific models: The global model is constructed from the constructs of the local models. It may not reflect all constructs of the specific models. It has however to reflect all those constructs that are common in at least two of the specific models.

Specific models as views of the global model: The specific model are obtained through filtration (e.g., by application of view construction, aggregation or summarisation functions) from the global model.

2.7 Integration of Static Integrity Constraints

A number of approaches are used for the integration of static integrity constraints into a model.

Built-in semantics: Each constructor in the language has its constraints that are built in.

Parameterised constructs: Constructors in the language are parameterised by attachable constraints. The constructor may be chosen without completion of the constraints. These constraints are considered to be parameters of the constructors that can be instantiated at a later development stage.

Constraint logic: The model $M$ (or the model type $T_{LS}$) allow an introduction of a specific logic $L_M$. Formulas $\Sigma_M$ of this language restrict the instances of this model.

The last approach is the most flexible one but requires a sophisticated logic background. For instance, models can be unsatisfiable. This approach has been chosen for the definition of the notion of the model. The second approach is used in many graphical models such as object-role modelling [Halpin(2009)]. The first approach is used for specific constructors such as the Is-A constructor.

2.8 Restricting Modelling by the Choice of Modelling Languages

Each modelling framework uses a number of modelling languages. Therefore, we are bound to the conceptions of the language, the expressivity of the language, and the methodology for language utilisation. These modelling languages are typically used in a sequential or partial concurrent way. This application of languages layers the development process. Stages (layers) and steps of modelling activities follow one another. They may iterate and can be applied in parallel. They also restrict what quality or capability properties must be satisfied. The design and development quality depends on main success factors: structuring of the process itself, culture of people involved, skills of actors, and process capabilities.
Languages may however also restrict modelling. The Sapir-Whorf hypothesis [Whorf(1980)] or the “principle of linguistic relativity” postulates that actors skilled in a language may not have a (deep) understanding of some concepts of other languages. This restriction leads to problematic or inadequate models or limits the representation of things and is not well understood.

3 Towards a Theory of Modelling Activities and Acts

Modelling activities are based on modelling acts. Modelling is a specific form and we may thus develop workflows of modelling activities. These workflows are based on work steps [Thalheim(2000)] such as ‘decompose’ or ‘extend’, abstraction and refinement acts, validation and verification, equivalences of concepts, transformation techniques, pragmatic solutions and last but not least the domain-specific solutions and languages given by the application and implementation domains.

3.1 The “Act” of Modelling

Modelling typically means the construction of models which can be used for detection of insights or for presentation of perceptions of systems. Modelling is typically based on languages and thus has a semiotic foundation.

The act of modelling consists of

1. a selection and construction of an appropriate model depending on the task and purpose and depending on the properties we are targeting and the context of the intended system and thus of the language appropriate for the system,

2. a workmanship on the model for detection of additional information about the original and of improved model,

3. an analogy conclusion or other derivations on the model and its relationship to the real world, and

4. a preparation of the model for its use in systems, to future evolution and to change.

The modelling act can be understood as a generalisation of the speech act. We may distinguish a number of activities depending of the subject. For instance, Figure 3 displays the conceptualisation act for application domain gathering, understanding, and modelling. Conceptualisation aims to collect objects, concepts and other entities that are assumed to exist in some area of interest and the relationships that hold among them. It is thus is an abstract, simplified view of the world that we wish to represent. A similar modelling act may be observed for evaluation. In this case, the resource dimension will be the the evaluation background instead of the application domain.
The act of modelling is based on an activity that is characterised by the work products, the aspects under consideration (scope), the resources used in an activity, and the partners involved into the activity. Additionally we might extend this characterisation by activity goals and intentions (for what), time span (when), and restrictions (normal, exception and forbidden cases) or obligations for later activities.

We may distinguish a number of activities and acts, e.g. the following ones:

**Understand:** The understanding act support reasoning within the application domain. It results typically in drafts that can be used for development of conceptualisations. The problems and possible solutions are comprehended. Conjectures are drawn.

**Conceptualise:** The conceptualization act aims in formalising the part of the application domain that is of interest. We form a number of concepts of the application domain and represent those formally within the concept language. These concepts are conceptually interpreted in the application domain.

**Abstract:** The abstraction act aims in outlining main problems that must be supported by the information system. It generalises these problems and abstracts from unnecessary details on the basis of forgetful mappings.

**Define:** The definition act is used to unambiguously specify, to delineate, and to delimit main concepts or annotations used for the development of the model. Definitions might be given in a variety of forms. We can use also different languages and target on visualisation of concepts.

**Construct:** The construction act is often considered to be the main act during modelling. It aims in creating a model by organizing and linking ideas, judgements, or concepts. It may include the sub-act of rebuilding, i.e., reconstructing, framing up, and customising. When we talk about anticipated behaviour it includes activities of conjecturing and hypothesising.
Refine: The refinement act is a basic act of iterative development. The model itself becomes enriched, more elaborated or sophisticated while preserving its main structures and behaviour. It matches thus in a better, more precise manner the needs of the application. The refinement act is typically based on some evaluation or assessment and on analysis for improvement potential. Refinement uses scoping for restricting changes to a necessary extent.

Evaluate: The evaluation act is based on a set of quality characteristics that we have agreed to satisfy in advance. These quality characteristics are typically given in an abstract form and are not solely based on metrics. Evaluation is typically applied to a model or parts of it. It results in determination of the value of the judgements under evaluation.

3.2 Principles of Understanding

Principles of understanding are not well developed in the information systems modelling field. Understanding is based on judgements and believes. We have to figure out why the part of the application domain is under consideration. We need to scope the part of the application domain. Finally, if we already know the modelling target and how to approach it, then we need to be consistent within the model.

Understanding also aims in becoming knowledgeable in the application domain and to develop a sense and sensibility for the application tasks. This includes an understanding of evolution within the application domain, of context for the application problems, of causal mechanisms within this domain, of basic units thereof, of structure and functioning, of levels and forms of organisation, of information flow within the application domain, and of stability with this application.

Application problems are driven by complex and deep motivations that must be understood in order to support their solution. They are limited in their capabilities and are shaped by past experience and approaches. They are using solutions that might be based on faulty logic and decision processes. Also a number of preferences has been selected in the past. We need to capture the potential of the new solutions.

A number of theories might be used for better understanding the application domain problems and the potential of solutions: Attribution theory, constructivism, focusing effect, framing, just-world phenomenon, objectification, organisational structures, and life case and story models.

3.3 Principles of Conceptualising

Principles of conceptualisation generalise the seven principles of Universal Design [Patil et al.(2003)]. We may differentiate between mandatory and optional
principles. These principles are best reflected by requirements to the model and to the judgements. Their realisation is an open research issue.

Typical mandatory principles are usefulness, flexibility, simplicity, realisability, and rationality. The model should be useful and marketable to people with diverse abilities. It provide the same means of use for all users, i.e., identical whenever possible and equivalent when not. It includes certain views that support quality properties such as privacy, security, and safety. The model must be flexible in use and accommodates a wide range of individual preferences and abilities. It facilitate different viewpoints depending on the stakeholder and depending on a wide range of individual preferences and abilities. The model must be simple and intuitive in use. The judgements are easy to understand, regardless of the user’s experience, knowledge, language skills, or current concentration level. A side-principle is the elimination of unnecessary complexity. Models must be consistent with user expectations and intuition, must accommodate a wide range of stakeholder skills, should arrange judgements consistent with its importance. The model can be used efficiently and comfortably and with a minimum of fatigue. It does not require additional reasoning capabilities or skills from its user. The model must be checkable within a validation, verification or testing procedure depending on the model’s purpose. Its adequateness for the purpose must be checkable and improvable.

Optional conceptualisation principles are perceptability, error-proneness, and parsimony. The model must be perceptible. It communicates necessary information effectively to the stakeholder, regardless of ambient conditions or the stakeholder’s abilities. It may use different representations depending on its use by the stakeholder, may be redundant presentation of essential information, and should allow to enlighten different viewpoints. The model does not allow unintended uses and is error-prone. It minimizes hazards and the adverse consequences of accidental or unintended actions. Elements that may be interpreted in unintended ways are avoided. The model can be surveyed without wrong interpretations. The model uses in an appropriate way space and resources. It is parsimonious both at the systems and at the stakeholder’s side. It provide a clear line of sight to important elements for any usage. It accommodate variations in hand and grip size and is well-supported.

3.4 Principles of Abstraction

The development of a model is the result of modelling. It relates things $\mathcal{D}$ under consideration with concepts $\mathcal{C}$. This relationship $\mathcal{R}$ is characterised by restrictions $\rho$ to its applicability, by a modality $\theta$ or rigidity of the relationship, and by the confidence $\Psi$ in the relationship. The model is agreed within a group $\mathcal{G}$ and valid in a certain world $\mathcal{W}$. Stachowiak [Stachowiak(1992)] defined three characteristic properties of models: the mapping property (have an original),
truncation property (the model lacks some of the ascriptions made to the original), and pragmatic property (the model use is only justified for particular model users, tools of investigation, and period of time). We can additionally consider the extension property. The property allows that models represent judgments which are not observed for the originals. In computing, for example, it is often important to use executable models. Finally, the distortion property is often used for improving the physical world or for inclusion of visions of better reality.

These principles result typically result in forgetful mappings from the origin to the model.

3.5 Principles of Definition

Defining is an act of determining, especially the logical meaning. A definition is either a statement expressing the essential nature of something or a statement of the meaning of a thing. The result meaning of definition the action or the power of describing, explaining, or making definite and clear. Definitions may use the clarity of visual presentation. They provide a sharp demarcation of outlines or limits.

The definition act is a formal passage describing the meaning of a concept or annotation. The concept to be defined is the definiendum. A concept may have many subtly different meanings and may be defined in a variety of ways. For each such meaning, a definiens is an expression in a certain language that defines that specific meaning of the concept or annotation.

We distinguish six different kinds of definitions. Typically, modelling uses distinct kinds of definitions in a combination. We thus may compose a definition by selection of definition parameters that are set by the hexagon in Figure 4.

The real or matter definition refines generic terms (genus proximum) by kind generating properties (differentia specifica). Specific kinds are genetic definitions and declarations. The nominal or stipulative definition is a type of definition in which a new or currently-existing concept or annotation is given a new meaning for the purposes of modelling in a given context. Specific kinds are precising definitions that narrow down the set of things meeting the definition. The assignment or attribution definition determines the relation among already defined concept or annotation. Specific kinds of such definitions are referential and association definitions that directly relate the concept or annotation to already existing ones. The inductive definition uses an initial concept or annotation, a set of constructors and a closure condition and typically defines a set of concept or annotation each of which is then considered to be a singleton one. The axiomatic definition uses a set of axioms and implicitly defines the definiens. The direct or explicit definition clearly separates definiens and definiendum. The indirect or implicit definition does not use such explicit separation. A definition may be complete or partial. Furthermore, definitions may provide syntactical,
Figure 4: Ingredients for selection of the style and of the formulation used of stating a definition

*semantical*, or *pragmatical* issues. Intensional definitions are a specific kind of definitions. They give the meaning of a concept or annotation by specifying all the necessary and sufficient conditions for belonging to the collection of objects being defined.

We might use during a definition act also other forms of definitions beside these ‘precise’ definitions. The *lexical* definition of a of a concept or annotation is the meaning of the concept or annotation in common usage. An *extensional* definition of a concept or annotation formulates its meaning by specifying its extension, that is, every object that falls under the definition of the concept or annotation in question.

### 3.6 Principles of Construction

Construction principles have mainly been developed for construction methods such as IDEF or specific paradigms such as object-oriented software construction. They might however be developed in a more general setting. We know a number of requirements to construction: right orientation and methodology, expectation orientation, user and usage focusing, realisability, satisfaction of main quality criteria such as safety, novel and creative decisions, and team orientation both for development and for deployment.

These requirements directly lead a number of engineering principles that are applicable to construction of models. *Engineering* is oriented towards encapsulation of of experiences with design problems pared down to a manageable scale. *Real-life engineering* is full of uncertainties and risks, impossible to replicate effectively in a formalised way in a text. An *engineering component* means any
engineering structure, which may be constructed from several interconnected elements into a single entity. Effectively withstanding loads is defined as the capacity to accept service loads without exceeding either the specified maximum stress, specified maximum deflection, or both of this specifications. Service loads are those loads, specified or unspecified, that the designer considers as creditable to be imposed on the component during its service life. Engineering may be performed in a systematic way based on conceptual modelling methodologies.

Therefore conceptual modelling inherits most principles of engineering. Modularity is probably the most fundamental principle of good engineering design. A large system should be analysed into smaller subsystems. Modularisation may either be explicit based on an architecture or implicit based on a name space. Systems should be constructed through subsystems. A hierarchical ordering should be specified so that modules are divided into layers, where each layer may interact with the layers just above and below, but not with distant layers. Therefore it is often useful to define system layers explicitly.

Engineering is based on a system architecture. We might distinguish a number of different architectural views such as the component architecture based on modules, the application architecture based on views depending on tasks under consideration, the infrastructure architecture based on embeddings into supporting systems, evaluation architectures depending on the purposes of the model (communication, construction, detection of solutions, exploration of the application domain, etc.), and interface architectures for embedded information systems.

Typically, information systems use a meta-structure called skeleton that either explicates the component structure with associations on the basis of views or interface for the component association or separates different concerns in the application from each other. Therefore, top-down development may start with drafting a skeleton and refining elements step by step. Similarly, bottom-up or inside-out development may use the skeleton for zooming-out unessential component details.

There are specific principles that are based on paradigms for object-relational information systems construction such as the Liskov-substitution-principle, the open-closed-principle, the dependency-inversion-principle, the interface-segregation-principle, the reuse-release-equivalence-principle, the common-closure-principle, the common-reuse-principle, the acyclic-dependencies-principle, the stable-dependencies-principle, the stable-abstractions-principle, and the law of Demeter. There are other principles that might also be of interest such as the linguistic-modular-units-principle, the few-interfaces-principle, the small-interfaces-principle, the explicit-interfaces-principle, and the information-hiding-principle. Most of these principles are well-known for software construction and can be used in the same way for conceptual modelling.
3.7 Principles of Refinement

Refinement relations are the key to formalise modelling activities and the modelling process. Classically [Broy(1997)], refinement methods are separated into

- *property refinement* - enhancing requirements - allows us to add properties to a specification,

- *glass box refinement* - designing implementations - allows us to decompose a component into a distributed system or to give a state transition description for a component specification, and

- *interaction refinement* - relating levels of abstraction - allows us to change the granularity of the interaction, the number and types of the channels of a component.

These notions of refinement are sufficient to describe all activities needed in the idealistic view of a strict top down hierarchical system development.

The theory of conceptual modelling may also be used for a selection and development of an assembly of modelling styles. Typical well-known refinement styles [Thalheim(2000)] for structure specification are the following ones:

**Inside-out refinement**: Inside-out refinement uses the given specification for extending it by additional part. These parts are hocked onto the current specification without changing it.

**Top-down refinement**: Top-down refinement uses decomposition of functions in the vocabulary and refinement of rules. Additionally, the specification may be extended by functions and rules that have not yet been considered.

**Bottom-up refinement**: Bottom-up refinement uses composition and generalisation of functions and of rules to more general or complex ones. Bottom-up refinement also uses generation of new functions and rules that have not yet been considered.

**Modular refinement**: Modular refinement is based on parqueting of applications and separation of concern. Refinement is only applied to one module and does not affect others. Modules may also be decomposed. Modules are typically associated through a skeleton that reflects the application architecture or the technical architecture.

**Mixed skeleton-driven refinement**: Mixed refinement is a combination of refinement techniques. It uses a skeleton of the application or a draft of the architecture. This draft is used for deriving plans for refinement. Each component

---

3 A theory of refinement for functionality, control or interface specification is still an open research issue.
or module is developed on its own based on top-down or bottom-up refinement. These different kinds of refinement styles allow one to derive plans for refinement and and primitives for refinement.

3.8 Principles of Evaluation

Evaluation and assessment of quality depends on the purposes of the model. Figure 1 shows that rather different quality criteria apply to a model depending on the goal of the model. Assessment is not targeting the quality of the modelling language\(^4\). It cannot be the final goal of a modelling act to develop a most correct or a most adequate model.

Evaluation assessment is one of the research issues in software development. There are many quality criteria that might be applied in different stages and life cycles [Jaakkola and Thalheim(2005)]. Most of these ISO/IEC 9126 or 25010, SPICE or CMMI standards introduce a large variety of quality characteristics and propose to measure quality fulfillment on the basis of metrics. Since judgements cannot be of equal rights these metrics fail for conceptual modelling.

A systematic way to quality-driven development is introduced in the Quality Attribute Driven Software Design Method by [Bass et al.(2003)]. This approach is based on the use of quality scenarios, in which a stimulus activates the operations specified by the selected quality tactics (operations). The execution of the tactics will cause an expected response in the system to meet the requirements of a quality attribute.

We might evaluate quality of the model or of the modelling acts. The first evaluation direction is easier to access and can be used for evaluation of quality assessment as already discussed in Section 1:

Development quality acts: Models, representations and judgements are subject to revision during development. The impact of each revision must be trackable for and understandable by the developer. Therefore, quality improvement acts include activities supporting analysability, changeability, stability, and testability of models. These activities must be pervasive and ubiquitous. At the same time, these activities must be efficient (e.g. use the resources in an appropriate way, be parsimonious). Since these tasks cannot be performed by a singleton person development acts are based on collaboration of developers with stakeholder and among developers. Each partner in a development process reflects his/her view on the application and on the system. Therefore, a tool support becomes essential.

\(^4\) The model language is also characterised by its expressive power and thus might support or hurt the model (size of the model, artificial constructs, independence of constructs used, reasoning capabilities, correctness of transformations, model size, etc.). Additionally, the representation language might also introduce difficulties.
Developers require that models must be easy to maintain, to analyse, to change, to test and must be stable. Therefore, a specific quality activity is the localisation of stable, evolving and immature model parts. Developers and coders are interested in facilities that support efficiency (time behaviour, resource utilization, efficiency compliance, scalability). Therefore, model languages must also be extended by performance improvement acts.

**Internal quality acts:** We may distinguish between functionality and general quality characteristics for internal quality. Functionality includes accuracy, suitability, interoperability, security, flexibility and self-contained/independence. The co-design approach to development includes development of structures and functionality. Therefore, it also includes acts that improve these quality characteristics.

Information systems must also be accurate, suitable, robust, self-contained, independent and internally parsimonious. Additionally they must be portable (adaptable, installable, replaceable, deployable, transferable, reusable, ...) and reliable (mature, fault tolerant, recoverable, ...). Therefore, quality assurance acts are folded into refinement activities. Classical information system models do neither provide measures and reasoning facilities for this second kind of internal quality nor have appropriate transformation techniques that improve quality.

**Quality in use acts:** Domain application engineers highly rate documentation activities that improve direct understandability and surveyability. User parsimony is an issue for business users. At the opposite side, developers are interested in evaluation of the model for extensibility, appropriateness, and operability. Quality in use deeply depends on the quality of representation. Therefore, quality improvement acts directly influence representation. Business user request availability, usability compliance and configuratability for their information systems applications. Therefore, quality improvement acts also use activities that scope applications to users’ needs.

Assessment of model quality is defined as an open process that encompasses the following principles:

- It is mission-focused, at both the institutional and programmatic levels.
- It is systematic, iterative, collaborative, documented, and adaptable.
- It applies multiple measures, both qualitative and quantitative.
- It identifies strengths and areas that warrant improvement.
- It informs planning and decision-making for the purpose of ascertaining learning and development, thereby improving programs, services, functions, performance, and the overall value of the educational experience.
Our framework is currently based on a seven-level model. The *specification level* is used for a description of quality depending on the purposes of the model and its main quality characteristics. The description consists of a specification of the quality property, the measurement, and the policies for evaluation. It can be extended by specific policies for various development methods such as agile development, by transformations of quality properties into others, and by associations among quality properties. Finally, we may derive constraints for the application of the quality property. The *control or technical level* deals with the application of the quality model. It provides guidance for the control procedures such as setting the control management, deriving the scope of control, definition of the control tasks and its actors. The application of the quality framework is based on a quality property portfolio. The portfolio consists of tasks and the necessary supporting instruments. They generalize portfolio known in project management. The *application or technology level* handles the management of quality evaluation within software etc. projects based on the technology of development. The *establishment or organizational level* is based on a methodology and may be supported by a quality maintenance system. The *value or prediction level* provides facilities for handling satisfaction of quality properties and for predicting changes in satisfaction whenever software evolves. The *optimisation level* integrates quality management into the optimisation of the software development process. The *maturity level* uses experiences gained for the innovation and adaptation of other processes and products that have not yet reached this maturity.

### 3.9 General Principles

Conceptual modelling acts are typically well organised. Therefore we may observe a number of general principles. The *conceptualization principle* restricts development of models to exclusively conceptual aspects of the application domain. The *95% -principle* requires that almost all relevant aspects of the application domain should be described in the conceptual schema. The *formalization principle* explicitly requires a potential formalisation of models and thus guarantees existence of a realisation. The *semiotic principle* restricts models to those that are easily interpretable and understandable. The *correspondence condition for representation* requires that the modellens should be such that the recognizable constituents of it have a one-to-one correspondence to the relevant constituents of the modellum. The *invariance principle* restricts models to those things in the application domain that are invariant during certain time periods within the application area. The *sub-schemata principle* explicitly bases modelling on skeletons.

In the case of multi-layered modelling acts we may derive a number of additional principles. The *downward-dependency principle* requests that the main
data dependency structure is top-down. Objects at a higher level depend on objects at a lower level. The upward-notification principle restricts objects at a higher level to act as subscribers to database changes at lower level. They may decide whether they eagerly or lazily enforce observed changes at lower level. Objects at lower level report however their changes. The neighbor-communication principle restricts object data exchange data to those objects at the same layer. The neighborhood may also require that neighboring databases should be synchronised. The explicit association principle request that the data exchange between database components is explicitly documented and recorded. Whenever a database at a higher level perceives data from a lower level then this exchange is logged. The cycle elimination principle brakes cyclic data exchange between layers based on the log information. The layer naming principle requires that data can be identified at their level.

3.10 Methods of Conceptual Modelling

The theory of conceptual modelling is based on a small number of methods. The main methods are abstraction, modularisation, inheritance, generalisation/refinement, transformations, selection and application of modelling styles, and separation of concern. Abstraction and refinement are well understood. Modularisation is based on an architectural decomposition of a large model into components and a development of a linking or binding scheme for the separated components. Typically, conceptual modelling only considers one transformation technique for the mapping among layers, e.g. from conceptual to logical schemata. The mapping technique and the mapping rules may however vary depending on the goals. Separation of concern allows to provide a clear understanding of parts of the application.

We are not going to develop a theory of these methods in depth although it is necessary for a theory of conceptual modelling. Abstraction is discussed in [Thalheim(2000)]. Modularisation and inheritance has found a deep foundation for programming languages. Refinement is still an open research issue. Transformations, modelling styles and pattern are dependent on languages chosen for specification.

4 Towards a Theory of Modelling Properties

4.1 Properties describing the Purpose of Conceptual Modelling

Models are developed with different goals, different scope, within different context, with different appeal to the receiver of the model, with different granularity, with different background, and with different realisation forms. Therefore we have to explicitly handle modelling purpose properties.
The mission of modelling is described by scope of the model, the users community, the tasks the model might support, the major and minor purposes satisfied by the model and the benefits obtained from the model for the given user community. The goals of a model are based on the impact of the model, restricted by the relationships among users and their roles they are playing. The brand of the model is given by the who-what-whom-action pattern. The metamodel can be used to provide information about the model such as the context of the model, the context in which the model might be useful to the auditory, the usage of the model, and the restrictions of the model.

It surprises that these model properties are not explicitly handled in most modelling approaches. The same surprise can be observed for a declaration of the main goals of the modelling act such as

construct a model, a part of the model, a concept or a judgement, etc. (describe, delineate, fabricate, master),
communicate the judgements, the observations, the concepts, etc. (explain, express, verbalise or display),
understand the application domain, the system opportunities, etc. (cognise, identify, recognise, percep),
discover the problems, the potential, the solutions, etc. (interact, identify),
indicate properties of importance, relevance, significance, etc. (visualise, measure, suggest, inform),
variate and optimise a solution, a judgement, a concept, a representation depending on some criteria,
verify or validate or test a model, a solution, a judgement, a representation or parts of those,
control the scope of modelling, the styles or pattern, parts of a model, judgements, etc. (rule, govern, proofread, confirm, restrain, administer, arrange, stratify, standardise),
alternate or compensate or replace or substitute or surrogate models or parts of them, judgements, concepts, etc. (transfer, reassign, evolve, migrate, balance, correct, novate, truncate, ersatz).

The first and last four goals lead to a datalogical model that is structured according to technology. The other goals result in an infological model that is delivered to the needs of the user. We thus use a different frame of reference. The application of the results may thus be descriptive or prescriptive, constitutive or prognosticating, categorical or exegetic or contemplative or formulaic.

4.2 Properties of the Modelling Process

The modelling process can be characterised by a number of (ideal) properties:
Monotonicity: The modelling process is monotone if any change to be applied to one specification leads to a refinement. It thus reflects requirements in a better form.

Incrementality: A modelling process is iterative or incremental if any step applied to a specification is only based on new requirements or obligations and on the current specification.

Finiteness: The modelling process is finite if any quality criteria can be checked in finite time applying a finite number of checks.

Application domain consistency: Any specification developed corresponds to the requirements and the obligations of the application domain. The appropriateness can be validated in the application domain.

Conservativeness: A modelling process is conservative if any model revision that cannot be reflected already in the current specification is entirely based on changes in the requirements.

Typical matured modelling processes are at least conservative and application domain consistent. Any finite modelling process can be transformed into a process that is application domain consistent. The inversion is not valid but depends on quality criteria we apply additionally. If the modelling process is application domain consistent then it can be transformed in an incremental one if we can extract such area of change in which consistency must be enforced.

4.3 The Implicitly Assumed Modelling Context Properties

Modelling is inherently incomplete, biased and ruled by scoping by the initiators of a project, by restricting attention to parts of the application that are currently under consideration and ruling out any part of the application that will never be under consideration. This intentional restriction is typically not communicated and directly results in the “modelling gap” [Kaschek(2003)]. Additionally, modelling culture results in different models and different understandings.

Incompleteness of specifications is caused by incomplete knowledge currently available, incomplete coverage of the specification or by inability to represent the knowledge in the application. Incompleteness may be considered as the main source of the modelling gap beside culture and skills of modelers. The application is either partially known, or only partially specified, or cannot be properly specified. This incompleteness leads to the modelling gap displayed in Figure 5.

To overcome the problems of specifications we may either use

– negated specifications that specify those cases which are not valid for the application,
Figure 5: Modelling decisions made in the database development process

- robust specifications that cover the main cases of the applications, or
- approximative specifications that cover the application on the basis of control parameters and abstract from order parameters.

5 Conclusion

5.1 Goals of this Paper

Modelling and especially conceptual modelling is not yet well understood and misinterpreted in a variety of ways. The first goal of the paper is to overcome some myths of conceptual modelling such as:

1. Modelling equals documentation.
2. You can think everything through from the start.
3. Modelling implies a heavyweight software process.
4. You must “freeze” requirements and then you can start with modelling.
5. Your model is carved in stone and changes only from time to time.

6. You must use a CASE tool.

7. Modelling is a waste of time.

8. The world revolves around data modelling.

9. All developers know how to model.

10. Modelling is independent on the language.

The second goal of this paper is the development of a framework to modelling. Modelling is based on an explicit choice of languages, on application of restrictions, on negotiation and on methodologies. Languages are defined through their syntactics, their semantics, and their pragmatics. We typically prefer inductive expression formation based on alphabets and behaviour defined on expressions. Restrictions depend on logics (deontic, epistemic, modal, belief, preferences) and use shortcuts, ambiguities, and ellipses. Negotiation support management or resolution of conflicts and the development of strategies to overcome these strategic, psychological, legal, and structural barriers. Development methodology are based on pragmatism and on paradigms. Since modelling is an activity that involve a number of actors the choice of languages becomes essential. Modelling is a process and based on modelling acts. These modelling acts are dependent from the purpose of modelling itself. Therefore we can distinguish different modelling acts such as understand, define, conceptualise, communicate, abstract, construct, refine, and evaluate. Depending on the purpose of model development we might use modelling act such as construct and evaluate as primary acts.

The third goal of this paper is to draw attention to explicit consideration of modelling properties both for the models themselves and for the modelling acts. This side of conceptual modelling is often only considered in an implicit form. The modelling process is governed by goals and purposes. Therefore, we must use different models such as a construction model, a communication model or a discussion model. Modelling is restricted by the application context, the actor context, the system context and the theory and experience context. These kinds of context restrict the model and the modelling process.

5.2 The Theory Framework

The aim of the paper has not been to develop a complete theory of conceptual modelling. Instead we aimed at the development of a programme for the theory. We described the general purpose of this theory, demonstrated how different paradigms can be selected, and showed which scope, modelling acts, modelling methods, modelling goals and modelling properties might be chosen for this theory.
5.3 Towards Modelling Principles

A theory of conceptual modelling can be based on a system of guiding principles. This paper shows that at least three guiding principles must be explored in detail:

**Internal principles** are based on a set of ground entities and ground processes.

**Bridge principles** explain the results of conceptual modelling in the context of their usage, for instance for explanation, verification/validation, and prognosis.

**Engineering principles** provide a framework for mastering the modelling process, for reasoning on the quality of a model, and for termination of a modelling process within a certain level of disturbance tolerance (error, incompleteness, open issues to be settled later, evolution).

Information systems modelling principles are specialisations of design principles [Denning(2007)]. They are conventions for planning and building correct, fast or high performance, fault tolerant, and fit information systems. Conceptual modelling is based on architecture of a system into components, uses their interactions, and pictures their layout. Modelling is the process of producing models. It is thus adapted from engineering and may thus use the separation of activities into requirements, specification, development, and testing.

Depending on the purpose of the model several quality criteria may be preferred. For instance, construction models should fulfill criteria for good models such as correctness of models, refinement to highly effective systems, fault tolerance of systems, ubiquity of systems, and fitness of systems.

Modelling principles are not laws of nature, but rather conventions that have been developed by modellers to the most success when pursuing quality properties. Therefore, various sets of principles might be developed depending on the community. For instance, modelling based on extended ER models is based on compositionality, incrementality, structure-orientation, and conservativeness. Modelling principles for sets of models such as UML are far more difficult to develop and to maintain.

5.4 Future Work

The programme requires far more work. The theory needs a variable taxonomy that allows a specialisation to languages chosen for a given application domain, must be based on a mathematical framework that allows to prove properties, must be flexible for coping with various modelling methodologies, must provide an understanding of the engineering of modelling, and finally should be supported by a meta-CASE tool that combines existing CASE to to a supporting workbench.
The programme aiming in the development of a general modelling theory becomes more crucial since model-driven approaches are going to be applied to practice and since modelling is going to be the programming of the future.

The theory of conceptual models is developed in [Thalheim(2010)]. It is based on a theory of languages that are used for conceptual modelling, on a notion of a conceptual model and concepts deployed for the model, on an explicit treatment of the information exchange between the stakeholders that are involved into conceptual modelling, on language-based mapping between an original and the model, and on postulates of conceptual modelling.

5.5 Towards an Agenda for Research

Finally we derive a number of open research fields for a theory of conceptual modelling:

**Adjustable selection of principles depending on modelling goals:** Since models satisfy different needs and purposes we should be able to unerringly and purposefully select target-aimed principles, to adjust models and modelling acts to these principles and to govern the process of modelling by appropriate properties.

**Model suites with explicit model association:** Since different application areas, different participating stakeholders, different modelling cultures and different modelling theories and experience result in a large variety of languages and a “Babylonian language confusion and muddle”, novel methods for model coexistence, for development of views on the same general model and for model management become more important.

**Development of a language culture:** Many languages and standards have been developed without insight into theories and without consideration of achievements of research in the past. The knowledge or culture seems to vanish. Holistic compilations of modelling culture and handbooks of conceptual modelling might a starting point if they find their way into teaching and research.

**Models 2.0:** Models are evolving and maturing. Although this evolution is well reported in papers or books old variants of models are still taken as the starting point without consideration of improvements. These results are however scattered and not compiled or collected. They are neither integrated into the body of knowledge obtained so far nor evaluated for their appropriateness.

---

5 Programming languages have been developed from first generation languages to fourth generation languages. The next generation of programming languages is going to be based on models that are used for transformation to programs. The claim in [Embley et al.(2010)] that programming is actually “conceptual-model programming” is the first starting point.
The evolution, progress and maturation should however be taken into account whenever a variant of the model is used. Therefore, a task of a science community is to garden models and to provide support for any newcomer.

**Explicit treatment of model value:** Model results leave a narrow gap to complete models. The model itself has a value according to the goal, maturity, and usage. The value depends on whether a models is used as an artifact, used for construction, used for negotiation or contracting among stakeholders, used for documentation or used for services and continuous evolution. The development effort for development of a model should match its value.

**Coexistence of theory, languages, and tools:** Modelling languages are often exclusively syntax-defined, often do not have an adequate theoretical foundation, overestimate the value of sophisticated graphical notations, and do have only partial tool support. Instead we need well-grounded languages with at least both syntax and semantics, with a variety of representations and with tool support that allow customisation to the needs of modellers.

**Adequate representation variants of models:** Models must be easy to learn and to understand. Users from any community should easily develop a familiarity with the representations. Domain-specific representations and consistency with user expectations do not distract users from the content of the model. Standard meanings support pragmatical treatment of models. Robustness and error-proneness allow the user to concentrate on the essential elements of the model and to abstract from the unessential shapes etc. of visual or textual elements.

**Compiler development for models:** Models are becoming omnipresent and omnipotent. Application engineers will be able to develop their models. These models are already ‘programs’ at a higher level. They are currently mainly interpreted and will become compiled to executable code in the future.

**Model families and variants:** Since modelling must support different goals and purposes models should be adaptable to those purposes and should be extensible in dependence on changing purposes.

**Acknowledgement**

This work is not yet complete and was a matter of discussions with colleagues or within workshops over a longer period of time. I am deeply indebted to Klaus-Dieter Schewe and Roland Kaschek for their interest, discussions, challenges, common work on this topic, etc. I am indebted to many of my colleagues to support me with their comments on this work, especially Bernd Mahr and Ulrich
Frank. I would like to acknowledge the discussions at the workshops Model-lierung 2006 and Modellierung 2009. I want to express my gratitude to IIftC for inspiration and (re-)questioning this approach.

References


Abstract. Conceptual modelling is one of the central activities in Computer Science. Conceptual models are mainly used as intermediate artifact for system construction. They are schematic descriptions of a system, a theory, or a phenomenon of an origin thus forming a model. A conceptual model is a model enhanced by concepts. The process of conceptual modelling is ruled by the purpose of modeling and the models. It is based on a number of modelling acts, on a number of correctness conditions, on modelling principles and postulates, and on paradigms of the background or substance theories. Purposes determine the (surplus) value of a model. Conceptual modelling is performed by a modeller that directs the process based on his/her experience, education, understanding, intention and attitude.

Conceptual models are products that are used by other stakeholders such as programmers, learners, business users, and evaluators. Conceptual models use a language as a carrier for the modelling artifact and are restricted by the expressiveness of this carrier. This language is often also used for the description of the concepts that are incorporated into a modelling result. Concepts can be explicitly defined or can be implicitly assumed based on some common sense within an application domain, a Computer Science sub-culture and within a community of practice.

A theory of conceptual models and a theory of modelling acts have been developed in [26,27]. This paper aims at a development of a general theory of modelling as an art in the sense of [9]. A general theory of modelling also considers modelling as an apprenticeship and as a technology. We distinguish between the art of modelling within a creation and production process, the art of modelling within an explanation and exploration process, the art of modelling within an optimisation and variation process, and the art of modelling within a verification process. This distinction allows to relate the specific purpose with macro-steps of modelling and with criteria for approval or refusal of modelling results.

Keywords. conceptual modelling, modelling workflow, foundations of modelling.

1. The Conceptions of a ‘Model’, of ‘To Model’ and of ‘Modelling’

Conceptual modelling is a widely applied practice in Computer Science and has led to a large body of knowledge on constructs that might be used for modelling and on methods that might be useful for modelling. It is commonly accepted that database application development is based on conceptual modelling. It is however surprising that only very few publications have been published on a theory of conceptual modelling. We continue the approach [27,26] and aim in a theory of modelling within this paper. An approach to a theory of models has been developed in [27]. An approach to a theory of model activities is discussed in [26].

1thalheim@is.informatik.uni-kiel.de  http://www.is.informatik.uni-kiel.de/~thalheim
1.1. Three Guiding Concerns during Conceptual Modelling

Conceptual modelling is often only discussed on the basis of modelling constructs and illustrated by some small examples. It has however three guiding concerns:

1. **Modelling language constructs** are applied during conceptual modelling. Their syntactics, semantics and pragmatics must be well understood.
2. **Application domain gathering** allows to understand the problems to be solved, the opportunities of solutions for a system, and the requirements and architecture that might be prescribed for the solution that has been chosen.
3. **Engineering** is oriented towards encapsulation of experiences with design problems pared down to a manageable scale.

The first concern is handled and well understood in the literature. Except few publications, e.g. [2], the second concern has not yet got a sophisticated and well understood support. The third concern has received much attention by data modelers [19] but did not get through to research literature. It must therefore be our goal to combine the three concerns into a holistic framework.

1.2. Implications for a Theory of Conceptual Modelling

The three concerns of conceptual modelling must be integrated into a framework that supports the relevant concern depending on the modelling work progress. The currently most difficult concern is the engineering concern. Engineering is inherently concerned with failures of construction, with incompleteness both in specification and in coverage of the application domain, with compromises for all quality dimensions, and with problems of technologies currently at hand.

At the same time, there is no universal approach and no universal language that cover all aspects of an application, that have a well-founded semantics for all constructions, that reflect any relevant facet in applications, and that support engineering. The choice of modelling languages is often a matter of preferences and case, empirical usage, evolution history, and supporting technology.

1.3. Differences between ‘Model’, ‘To Model’ and ‘Modelling’

The conceptions of model, of the activity ‘to model’ and of modelling are often used as synonyms. We must however distinguish these conceptions for a theory of models, a theory of model activities and a theory of the modelling process.

Based on the notions in the Encyclopedia Britannica [17] we distinguish between the conception of a model, the conception of a model activity, and the conception of modelling processes.

**The model as an artifact:** The model is something set or held for guidance or imitation of an origin and is a product at the same time. Models are enduring, justified and adequate artifacts from one side. From the other side, models represent the state of comprehension or knowledge of a user.

**To model as an activity:** ‘To model’ is a scientific or engineering activity beside theoretical or experimental investigation. The activity is an additive process. Corrections are possible during this activity. Modelled work may be used for construction of systems, for exploration of a system, for definition and negotiation, for communication, for understanding and for problem solving.
Modelling as a systematically performed technological process: Modelling is a technique of systematically using knowledge from computer science and engineering to introduce technological innovations into the planning and development stages of a system. At each stage the modeller is likely to ask both why and how, rather than merely how. Modelling is thus based on paradigms and principles.

Additionally, the notion of model may be used in an adjective sense as serving as or capable of serving as a pattern or being a usually miniature representation of something. This notion is often used for sample representations such as a ‘model chair’. Another notion of the model that is not of interest within this paper is the miniature representation of something.

1.4. The Simultaneity of Art, Technology and Techniques in Modelling

Modelling can be understood as a technique\(^2\) or as a technology\(^3\). [17] distinguishes between science and technology: Technology is the systematic study of techniques for making and doing things; science is the systematic attempt to understand and interpret the world. While technology is concerned with the fabrication and use of artifacts, science is devoted to the more conceptual enterprise of understanding the environment, and it depends upon the comparatively sophisticated skills of literacy and numeracy.

At the same time, modelling is an art\(^4\). Modelling is a highly creative process. It requires skills in planning, making, or executing. It is often claimed that it is not to be formalisable. It requires deep insight into the background as well as skills, careful simplification, experience and ingenuity. Due to the variety of viewpoints, modelling is also based on judgement and clever selection with different alternatives.

1.5. Conceptual Modelling: Modelling Enhanced by Concepts

An information system model is typically a schematic description of a system, theory, or phenomenon of an origin that accounts for known or inferred properties of the origin and may be used for further study of characteristics of the origin. Conceptual modelling aims to create an abstract representation of the situation under investigation, or more precisely, the way users think about it. Conceptual models enhance models with concepts that are commonly shared within a community or at least between the stakeholders involved in the modelling process. A general definition of concepts is given in [8,27]. Concepts specify our knowledge what things are there and what properties things have. Concepts are used in everyday life as a communication vehicle and as a reasoning chunk. Conceptualisation aims at collection of objects, concepts and other entities that are assumed to exist in some area of interest and the relationships that hold among them. It is thus an abstract, simplified view or description of the world that we wish to represent.

\(^2\)I.e., the fashion, manner, mode, modus, system, way, wise in which a system etc. is mastered. Techniques consist of methods of accomplishing a desired aim.

\(^3\)Technology is an element of engineering. It consists of the practical application of knowledge especially in a particular area. It provides a capability given by the practical application of knowledge. Therefore, it is a manner of accomplishing a task especially using technical processes, methods, or knowledge.

\(^4\)Art requires capability, competence, handiness, and proficiency. Art is based on finesse, i.e. on refinement or delicacy of workmanship. Models and art share a Janus head evaluation: The judgement of beauty evaluates the model within a community of business users. The judgement of the sublime evaluates the model against its technical realisation. A model has thus both an extrinsic and intrinsic value. Art will be used in the sequel in this paper within the wide understanding of [9].
1.6. The Theory of Conceptual Models

The theory of conceptual models [4] extends the framework [22,23]. A model can be characterised by four main dimensions: (1) purpose, (2) mapping of an origin, (3) use of languages as a carrier, and (4) providing a value.

The purpose of a model covers a variety of different intentions and aims such as perception support for understanding the application domain, explanation and demonstration for understanding an origin, preparation to management and handling of the origin, optimisation of the origin, hypothesis verification through the model, construction of an artifact or of a program, control of parts of the application, simulation of behaviour in certain situations, and substitution for a part of the application. Depending of the purpose we shall use different models.

Models are author-driven and addressee-oriented. They depend therefore on the culture, attitude, perceptions, education, viewpoints etc. of the stakeholders involved into the modelling process. Models are purposeful/situated/easily-modifiable/sharable/reusable/multi-disciplinary/multi-media chunks of knowledge about the application domain. They are both bigger and smaller than theories, i.e., bigger since they integrate ideas from different theories, since they use different representations, and since they are directed by their purpose; smaller since they are created for their purpose in a specific situation and since they are developed to be sharable and reusable. One of the most important quality characteristics of a model is that it should be easy to modify and to adapt.

1.7. The Theory of Model Activities as a Process

Activities for ‘to model’ (model activities) are based on modelling acts. The process for ‘to model’ is a specific form of a process. We may thus develop workflows of such activities. These workflows are based on work steps [24] such as ‘decompose’ or ‘extend’, abstraction and refinement acts, validation and verification, equivalences of concepts, transformation techniques, pragmatistic solutions and last but not least the domain-specific solutions and languages given by the application and implementation domains.

The act of ‘to model’ is based on an activity that is characterised by the work products, the aspects under consideration (scope), the resources used in an activity, and the partners involved into the activity. Additionally we might extend this characterisation by activity goals and intentions (for what), time span (when), and restrictions (normal, exception and forbidden cases) or obligations for later activities. We may distinguish a number of activities and acts, e.g., understand, conceptualise, abstract, define, construct, refine, document and evaluate. A theory of model activities has been developed in [26]. Model activities should be governed by good practices which can be partially derived from modelling as an apprenticeship or technology.

1.8. Orientation of this Paper

This paper explores modelling as an art. We base the discussion on a theory of models and of model activities. We abstract therefore in this paper from micro-, meso-, macro-models used in many natural sciences or model suites [25], e.g., model ensembles used in UML or OWL. We do not yet consider modelling competency or MDA/D.

Based on this understanding we may conclude that modelling requires apprenticeship and technology. The orientation towards an expert mode can be reached if modelling
is based on systematic development and if modelling is considered to be a craft of model activities. This approach shows that modelling incorporates design science in a wider sense as it has been considered in the literature.

We base our ideas on our observations on model developments for very large database schemata and very large database systems. Such systems require a well organised modelling process. They must be evolution-prone and revision-prone. Therefore, the model is typically on much higher quality than those models in textbooks. The paper concentrates thus one of the main workflows for information system development: description of application worlds followed by prescription for system worlds.

2. The Model World and its Dimensions

2.1. Main Dimensions of Models

Models are artifacts. They can thus be characterised by main (primary) dimensions:

- **Purpose** (“wherefore”) of models and modelling with the intentions, goals, aims, and tasks that are going to be solved by the model,
- **Result of mapping** (“whereof”) with a description of the solution provided by the model, the characterisation of the problem, phenomena, construction or application domain through the model,
- **Language** (“wherewith”) based on a careful selection of the carrier (or cargo) or language with all its restrictions that allows to express the solution, the specification of the world or the construction, and
- **Value** (“worthiness”) of a model by explicit statement of the internal and external qualities, and the quality of use, e.g. either by explicit statement of invariance properties relating the model to its associated worlds or by preservation properties that are satisfied by the model in dependence on the associated worlds.

The mapping associates the origin and the artifact or the artifact and its realisation. The mapping dimension is discussed in [26]. As far as we are interested in modelling of information systems, we may use a (semi-)formal language for language dimension of the artifact. The main dimensions of models and modelling govern the model and the modelling acts. The value dimension can be described based on [7].

The main dimensions are extended by different context dimensions that are used to shape and to adapt the model, e.g., (a) the **application domain dimension** describes the scope and the neglection of the model, (b) the **systems dimension** reflects the realisation

---

5Due to our involvement into the development and the service for the CASE workbenches (DB) and ID we have collected a large number of real life applications. Some of them have been really large or very large, i.e., consisting of more than 1,000 attribute, entity and relationship types. The largest schema in our database schema library contains more than 19,000 entity and relationship types and more than 60,000 attribute types that need to be considered as different. Another large database schema is the SAP R/3 schema. It has been analyzed in 1999 by a SAP group headed by the author during his sabbatical at SAP. At that time, the R/3 database used more than 16,500 relation types, more than 35,000 views and more than 150,000 functions. The number of attributes has been estimated by 40,000. Meanwhile, more than 21,000 relation types are used. The schema has a large number of redundant types which redundancy is only partially maintained. The SAP R/3 is a very typical example of a poorly documented system. Most of the design decisions are now forgotten. The high type redundancy is mainly caused by the incomplete knowledge on the schema that has been developed in different departments of SAP.
opportunities, weaknesses, strengths and threats, and (c) the user or stakeholder dimension describes the viewpoint, orientation and background of users involved.

2.2. The Model as a Physical or Virtual Artifact

The main product of modelling and model activities is the model, i.e. an artifact that is considered to be worth for its purpose by the author. The model can, for instance, be used for the description of the world of origins or for the prescription of constructions. There are a number of explicit choices an author makes and that rule applications of models. Modelling of information systems

depends on the abstraction layer, e.g. requirements, specification, realisation or implementation layer,

depends on chosen granularity and precision of the work product itself,

depends on resources used for development of a model such as the language,

depends on level of separation of concern such as static/dynamic properties, local/global scope, facets,

depends on quality properties of the input, e.g. requirements, completeness, conciseness, coherence, understandability,

depends on decomposition of the work products in ensembles of sub-products, and satisfies quality characteristics such as quality in use, internal quality, and external quality.

The task of model development is never completed (τα παντα ρηι, ‘the rivers flow’; narrative: everything flows). Models are changing artifacts due to changes imposed by

scope insight for conscious handling of restriction, capabilities, opportunities, 
guiding rules for convenience, for completion, refinement, and extension,
development plans for partial delivery of models, partial usage and deployment, 
theories supporting development of models, 
quality characteristics for model completion, model evolution, model engineering, and
mapping styles for mapping models among abstraction layers.

2.3. The Purpose Dimension

The purpose dimension is ruling and governing the model, the development process and the application process because of the main reason for using a model is to provide a solution to a problem. Therefore the purpose is characterised by the solution to the problem provided by the model. We may distinguish a number of concerns such as

the impact of the model (“whereto”) for a solution to a problem,
the insight into the origin’s properties (“how”) by giving details how the world is structured or should be structured and how the functionality can be described,
restrictions on applicability and validity (“when”) of a model for some specific solutions, for the validity interval, and the lifespan of a model,
providing reasons for model value (“why”) such as correctness, generality, usefulness, comprehensibility, and novelty, and
the description of functioning of a model (“for which reason”) based on the model capacity.
The purpose dimension governs the workflows applied in conceptual modelling. It also governs the kind of model application. We may distinguish a number of workflows in conceptual modelling such as the following ones:

**Description-prescription workflows** result in creation of models that are used for production of systems.

**Explanation workflows** result in new insight into the world of the origins.

**Optimisation-variation workflows** result in an improvement and adaptation of the origins.

**Verification-validation-testing workflows** result in an improvement of the one of the subjects considered, in most cases in an improvement of models.

**Reflection-optimisation workflows** are typical for mathematical modelling of the world of origins.

**Explorative workflows** are using models for learning about origins.

**Hypothetical workflows** are typical for discovery science, e.g., science used for climate research.

**Documentation-visualisation workflows** target on better understanding and comprehension of models.

These workflows can be intertwined or shuffled with each other. They may be performed one after another. In this paper we concentrate on the description-prescription (or creation-production) workflow which seems to be central for information systems.

### 2.4. The Language Dimension

Models are represented by *artifacts* that satisfy the pragmatic purposes of users. We restrict our discussion within this subsection to formal languages that are typically used for conceptual models. In this case, artifacts are linguistic expressions that describe the model. Linguistic expressions are built within a language with some understanding. Therefore, artifacts use syntax, semantics and pragmatics built within the chosen language.

Figure 1 displays the relationship between an artifact and its objectives and properties. A model should support its objectives. Optimally, these objectives $\Psi(G)$ can be expressed in the same language $L_G$ that is also used for the model $G$. A model has a number of properties. Some of them are of interest and used for characterisation of the model, e.g., $\Phi(G)$. This characterisation depends on the model and its purpose.

![Figure 1](https://via.placeholder.com/150)

**Figure 1.** Artifacts with a language, their properties and objectives within a given language for the artifact

Constructive languages are a special case and support

- the prescription of the objectives or postulates that restrict the judgement that an artifact can be accepted as a model,
• the scope of our attention to those artifacts that can be considered for a model or for parts of a model, and
• the orientation of the user on certain properties that are of interest for the purpose of modelling.

2.5. The Context Dimensions

2.5.1. The User Dimension

A number of users are involved into the development of models. The user dimension thus reflects intentions, the understanding, the comprehension and other characteristics of users in a variety of roles, e.g.,

the role of an author ("by whom") that results in reflections of the educational level, application of templates, pattern or reference models,
the role of an addressee ("to whom") that restricts the utilisation of the model or that supports the extended application beyond the purpose originally intended, and
the role of broad public ("whichever") that develops a common understanding of the model depending on the group or the culture of the public.

2.5.2. The Application Domain Dimension and the World of Origins

The application domain consists of people, organisational systems, and technical systems that interact to work towards a goal. This dimension clarifies

the domain depending on models purpose ("for what") such as an application domain, properties reflected or neglected,
the scope to specific elements ("what") that are considered to be typical and whose properties should be reflected,
the attention within the domain depending on models purpose ("where") that limits the model to the ‘normal’ aspects,
the orientation of the domain ("wherefrom") that restricts the attention and the issues for the current activities supported by the model,
the sources for origins or the infrastructure ("whence") considered for the model, and
the restrictions of the world ("wherein") associated with the model.

3. The Modelling Process

3.1. Conceptual Modelling Activities Governed by its Purpose

Models are developed with different goals, different scope, within different context, with different appeal to the receiver of the model, with different granularity, with different background, and with different realisation forms. Therefore we have to explicitly handle modelling purpose properties.

The mission of modelling is described by scope of the model, the users community, the tasks the model might support, the major and minor purposes satisfied by the model and the benefits obtained from the model for the given user community. The goals of a model are based on the impact of the model, restricted by the relationships among users and their roles they are playing. The brand of the model is given by the who-
what-whom-action pattern. The meta-model can be used to provide information about
the model such as the context of the model, the context in which the model might be
useful to the auditory, the usage of the model, and the restrictions of the model.

It surprises that these model properties are not explicitly handled in most modelling
approaches. The same surprise can be observed for a declaration of the main goals of the
modelling act such as

construct a model, a part of the model, a concept or a judgement, etc. (describe, delineate,
fabricate, master),
communicate the judgements, the observations, the concepts, etc. (explain, express, ver-
balise or display),
understand the application domain, the system opportunities, etc. (cognise, identify,
recognise, perpect),
discover the problems, the potential, the solutions, etc. (interact, identify),
indicate properties of importance, relevance, significance, etc. (visualise, measure, sugg-
est, inform),
variate and optimise a solution, a judgement, a concept, a representation depending on
some criteria,
verify or validate or test a model, a solution, a judgement, a representation or parts of
those,
control the scope of modelling, the styles or pattern, parts of a model, judgements, etc.
(rule, govern, proofread, confirm, restrain, administer, arrange, stratify, standardise),
alternate or compensate or replace or substitute or surrogate models or parts of them,
judgements, concepts, etc. (transfer, reassign, evolve, migrate, balance, correct, novate,
truncate, ersatz).

The first and last four goals lead to a datalogical model that is structured according to
technology. The other goals result in an infological model that is delivered to the needs
of the user. We thus use a different frame of reference. The application of the results may
thus be descriptive or prescriptive, constitutive or prognosticating, categorical or exegetic
or contemplative or formulaic.

3.2. Properties of Activities To Model

Activities to model form a process and can be characterised by a number of (ideal) prop-
erties:

Monotonicity: Activities are monotone if any change to be applied to one specification
leads to a refinement. It thus reflects requirements in a better form.
Incrementality: Activities are iterative or incremental if any step applied to a specifica-
tion is only based on new requirements or obligations and on the current specifici-
cation.
Finiteness: Activities are finite if any quality criteria can be checked in finite time ap-
plying a finite number of checks.
Application domain consistency: Any specification developed corresponds to the re-
quirements and the obligations of the application domain. The appropriateness can
be validated in the application domain.
Conservativeness: Activities are conservative if any model revision that cannot be re-
lected already in the current specification is entirely based on changes in the re-
quirements.
Typical matured activities to model are at least conservative and application domain consistent. Any finite sequence of activities can be transformed into a process that is application domain consistent. The inversion is not valid but depends on quality criteria we apply additionally. If the modelling process is application domain consistent then it can be transformed in an incremental one if we can extract such area of change in which consistency must be enforced.

3.3. Towards Modelling Principles

A theory of conceptual modelling can be based on a system of guiding principles. We conclude that at least three guiding principles must be explored in detail:

**Internal principles** are based on a set of ground entities and ground processes.

**Bridge principles** explain the results of conceptual modelling in the context of their usage, for instance for explanation, verification/validation, and prognosis.

**Engineering principles** provide a framework for mastering the modelling process, for reasoning on the quality of a model, and for termination of a modelling process within a certain level of disturbance tolerance (error, incompleteness, open issues to be settled later, evolution).

Information systems modelling principles are specialisations of design principles [3]. They are conventions for planning and building correct, fast or high performance, fault tolerant, and fit information systems. Conceptual modelling is based on architecture of a system into components, uses their interactions, and pictures their layout. Modelling is the process of producing models. It is thus adapted from engineering and may thus use the separation of activities into requirements, specification, development, and testing.

Depending on the purpose of the model several quality criteria may be preferred. For instance, construction models should fulfill criteria for good models such as correctness of models, refinement to highly effective systems, fault tolerance of systems, ubiquity of systems, and fitness of systems.

Modelling principles are not laws of nature, but rather conventions that have been developed by modellers to the most success when pursuing quality properties. Therefore, various sets of principles might be developed depending on the community. For instance, modelling based on extended ER models is based on compositionality, incrementality, structure-orientation, and conservativeness. Modelling principles for sets of models such as UML are far more difficult to develop and to maintain.

4. Modelling of Information Systems as an Engineering Science

In the sequel we concentrate on one of the workflows: the prescription of systems imposed by the description of an application domain and of the problems under solution. This workflow is often considered to be one of the main workflows. We may also use other workflows. The description-prescription workflow is however a typical example of an engineering workflow⁶. Engineering is nowadays performed in a systematic and

---

⁶The difference between scientific exploration and engineering is characterised by [18] as follows: ‘Scientists look at things that are and ask ‘why’; engineers dream of things that never were and ask ‘why not’. Engineers use materials, whose properties they do not properly understand, to form them into shapes, whose
well-understood form. We can thus include engineering approaches to modelling. We first review contributions made by design science, e.g., [5,6,12,28] and then develop our approach.

4.1. The Design Science Approach

MIS design science aims at the development of a general theory for models, model activities and modelling. We shall use the approach for a deeper insight into modelling. Models are called ‘design’ in [6].

The management information system community characterises the modelling process by seven guidelines [6]:

1. models are purposeful IT artifacts created to address a problem;
2. models are solutions to relevant and important problems;
3. the utility, quality, and efficacy of models must be evaluated by quality assessment;
4. modelling research must contribute to the state of the art;
5. modelling research relies upon the application of rigorous methods;
6. modelling is a search process and use termination conditions;
7. models must be communicated both to technology-oriented as well as to management audiences.

We observe that guidelines (1), (2), and (7) are characterising the model. Guidelines (3), (6) characterise model activities. Guideline (3), (5) is related to modelling as a technology. Guideline (4) is a general statement that relates modelling to a science.

Main ingredients of modelling can be derived from these guidelines [1,5,12,14,20,29]. Core components are purpose and scope (causa finalis), artifacts (causa materialis), the oneness of form and function (causa formalis), artifacts mutability, testable propositions about the model, and theoretical underpinning. Additional requests are the potential implementation (causa efficiens) and utility for exposition and testing [5].

Design science separates three cycles [28]: the relevance (or description) cycle, the design (or modelling) cycle, and the rigor (or conceptualisation) cycle.

4.2. Reasoning Support for Modelling

Design science [6] has been targeting on an explicit support for the modelling process. This support includes an explicit consideration of the quality of the model, of the quality of the modelling process, and of the quality of supporting theories. We may combine the informal discussions with our approach and separate the modelling acts by the things that are under consideration. Figure 2 displays the different ways of working during a database systems development. We use here the two-phase model: Description followed by prescription.

These different “ways of working” characterise

- the modelling acts with its specifics; [26]
- the foundation for the modelling acts with the theory that is going to support this act, the technics that can be used for the start, completion and for the support of the modelling act, and the reasoning techniques that can be applied for each step;

geometries they cannot properly analyse, to resist forces they cannot properly assess, in such a way that the public at large has no reason to suspect the extent of their ignorance.” Modelling incorporates both engineering and science. It is thus considered to be an engineering science.
• the partner involved with their obligations, permissions, and restrictions, with their roles and rights, and with their play;
• the aspects that are under consideration for the current modelling acts;
• the consumed and produced elements of the artifact that are under consideration during work;
• the resources that must be obtained, that can be used or that are going to be modified during a modelling act.

Consider, for instance, the way or requiring. It includes specific facets such as
• to command, to require, to compel, and to make someone do something with supporting acts such as requesting, ordering, forbidding, proscribing;
• to consider obligatory, to request and expect with specific supporting acts such as transmitting, communicating, calling for, demanding;
• to need with supporting acts of wanting, requiring;
• to necessitate, to postulate, to take, to involve, and to require as useful, to just, or to proper.

The ways of functioning, understanding, elicitation, modelling, reasoning, assessment, and construction can be characterised in a similar form.

The realisation stage may be replaced by other stage that support different purposes. We concentrated on prescription and construction of new systems. Another application is model refinement.

Design science aims at another kind of model refinement by adding more rigor after evaluation of a model. This refinement is essentially model evolution and model evaluation. Another refinement is the enhancement of models by concepts. This refinement is essentially a ‘semantification’ or model conceptualisation of the model.

Experimentation and justification of models is a third kind of adding rigor to (conceptual) models.
4.3. Observations for Information Systems Model Engineering

Engineering of conceptual models inherits both facets of didactically ruled learning [21] and of engineering [18]. The following characteristics of engineering sciences are observed also for conceptual modelling:

(α) The origin of a model is partly a product of creativity.
Systems developed in our field are a product of developers and thus dependent on these stakeholders. They must be understood, well-explained and used with a purpose.

(β) The origin of a model is a complex system.
The attention focuses both on the creation of complex artifacts and on conceptualisations of the application world. They are typically modularly constructed. Modularisation is only one of the underlying design principles. Conceptual modelling targets at useful concepts. It goes through a series of iterative design cycles in dependence on its purpose.

(γ) Models satisfy the purpose, are sharable, useful and reusable.
Models are not developed just as an intermediate result of the implementation process and for satisfaction of purpose. They are shared within a community and are reusable in other situations. Moreover, models support a better understanding of the origins.

(δ) The origins are continuously changing and thus the models too.
The application domain is continuously evolving. Models must correspondingly evolve too. Significant changes tend to be applied to the starting model so that the original concepts become unrecognisable after model evolution. Models are also used for changing the application world. This change must again be reflected within the model.

(ε) Origins being modelled are influenced by social constraints and affordances.
Models are influenced as much by purposes as by physical and economical aspects of the contexts in which they are used. These influences are changing and evolving as well. Therefore, models are going to be used in ways their stakeholders did not imagine. Models are influenced as much by socially generated capital, constraints, and affordances as by the capabilities of stakeholders who created them.

(ζ) No single “grand theory” is likely to provide realistic solutions to realistic complex application problems.
In realistic modelling situations that involve information systems, there almost never exist unlimited resources. Relevant stakeholders have typically conflicting goals. Therefore, ways of working displayed in Figure 2 usually need to integrate approaches drawn from different disciplines.

Artifacts that serve as models are developed through a series of model activities and are iteratively tested and revised in dependence on the purpose.
**Consequences for model engineering:** The modelling process itself also changes the application domain and the understanding of the origin. Therefore, modelling is not reducible to condition-action rules. Modelling is a matter of engineering. Experienced modellers not only right develop a model but they also develop the right model - by developing models at the right time, with the right background and context, and for the right purpose. Model engineering is therefore based on advanced skills of handcrafting, i.e., making substitutions and adaptations depending on purpose and application situation, understanding which compositions perform best, continuously adapt the result of the process, and understand difficult-to-control things in their handcraft environment. The same situation is valid for information systems development if performance of the system becomes crucial and heavily depends on the DBMS.

### 4.4. Modelling Generalising Engineering Approaches

The development of models offer many challenges. Modelling is essentially synthetic rather than analytic in substance. Identifying the real task of the modelling problem is probably the greatest challenge. Models can be based on building blocks. Another challenge is to find the right modelling method. Engineering targets at capacity of products to withstand service load both effectively and efficiently during their service life [18]. Efficiency also considers performance of the system. Engineering is also concerned with avoidance of technical, operational or unpredictable failures, i.e. to develop a system that deflect all service loads.

Engineering science for modelling is based on many different supporting sciences and technologies: industrial design, ergonomics, aesthetics, environments, life sciences, economics, mathematics, marketing, and manufactoring and forming processes. Engineering design includes five facets: design for effective function, design for manufacture, design for human users, socially responsible design, and economically responsible design.

Engineering distinguishes three dimensions: the stakeholder dimension, the procedure or process dimension, and the product dimension. It uses many techniques such as enformulation for structuring the purposes and objectives, problem decomposition together with component engineering, problem evolution, organising the engineering process, result evaluation, and result management. It also considers economic, social and environmental issues.

Therefore, it seems to be natural to use achievements of engineering for understanding modelling. This similarity is not only applicable to the description-prescription workflow but also for all the other workflows.

### 4.5. Maieutics for Mastering Iterations

Modelling of information systems is not only aiming at achieving a nominal system but aims too at satisfaction of real interests of all stakeholders involved into modelling. It must consider all relevant aspects of an application and thus results in co-design of structuring, functionality, and supporting systems such as view and interaction support [24]. Stakeholders (or users) iteratively obtain a deeper insight and understanding about the necessities and conditions of the problem and the strengths, weaknesses, opportunities and threats of the solution depending of the purpose of the modelling within a modelling process. Therefore, modelling integrates ideas developed for maieutics [10,13].
The maieutics frame [15] is essentially a specific form of a dialogue. In conceptual modelling, it consists (1) of an open-ended process, (2) of the elaboration of ideas that are grounded in references to the application domain, to the users, prior knowledge and experience, and to the languages as carriers, and (3) of the discussion (in form of conceptualisation, interpretation, explanation, diverging ideas, and new understandings) that is inductive and exploratory rather than deductive and conclusive.

Modelling requires to utilise the knowledge in dependence on the purpose of the model. Answers found during modelling may not be evident in the material on hand; modellers may have to delve into subtleties or ambiguities they had not thought of. Information systems modelling is based on elaboration and conceptualisation of model elements. The inductive and exploratory discussion facilitates the development of argumentation by fostering the (re)consideration of alternatives and versions.

Conceptual modelling is based on references to the application domain, connections across the model, elaboration based on prior knowledge and/or experience, interpretations, explanations and conceptualisations, diverging ideas, and new understandings. Therefore the modelling process is highly iterative and revising/remastering decisions that have already been made.

5. The Description-Prescription Information Systems Modelling Workflow

Modelling is based on an evolutionary process and thus consists of at least three sub-processes:

- selection including rigorous testing against the origin,
- communication for generation of a common understanding and a productive way of thinking within a community, and
- accumulation of results and integration of these results into future developments.

5.1. Examples of Workflows

The description-prescription workflow is one of the most prominent workflows in information system modelling. Methodologies developed for software engineering can be directly applied to this workflow. They are however mainly oriented towards system construction. The systems dimension is not as well explored. The combination of these two sub-workflows is shown in Figure 3. We need to include into this combination also the quality dimension. The body of knowledge of software engineering includes also a large set of quality characteristics. [7] develops an approach to systematic quality development. We integrate this systematic quality management.

We can also develop other workflows such as agile modelling, spiral modelling and incremental modelling workflows. We restrict our attention in the sequel to the workflow in Figure 3 due to the length of this paper. This workflow separates three different worlds: the world of applications, e.g., the application domain in dependence on the purpose; the world of models, e.g. conceptual models used for information system development; the world of systems, e.g., information systems combined with presentation systems. Based on this separation we can distinguish three stages: the relevance stage, the modelling stage, and the realisation stage. This workflow reflects the separation into objectives, the artifact and the properties within the language dimension.
5.2. Sub-Workflows

Modelling integrates the classical problem solving four-phase cycle [16] in each of the sub-workflows as well:

1. Developing an understanding of the task: The task is analysed within its context and is compared with the goal for completion of the task. The initial situation is characterised. In the case of modelling processes the understanding is based on objectives derived from the purpose or from previous steps.

2. Development of a plan for the solution of a task: The instruments and tools for the solution of the task are reconsidered. The plan consists of heuristic forward and backtracking steps, steps for problem restructuring and for quality control.

3. Application of the plan for the development of the solution: The plan is consecutively applied for the generation of the solution. If certain steps are considered to be inappropriate then the plan is revised as well.

4. Development of an understanding of the solution: The result is evaluated based on criteria that either follow from the purpose or from the problem. Properties of the solution are derived.

This approach uses four state spaces:

The state space consists of the collection of all those states that are reachable from the initial state. Some of the states are considered to be desirable, i.e. are goal states. States can be modelled through languages such as ER. States may have properties such as suitability for certain purposes.
The actions allow to move from one state to another state under certain conditions. We may assume that the effect of the actions is observable to a certain extent by the user. User may use several actions in parallel. Actions may be blocked or enabled depending on conditions. Actions may be used at some cost.

The goal test determines whether a given state or state set satisfies the goals. The goal test may be defined through a set of states or through properties. The goal test may also allow to state which quality has the state set for the problem solution.

The controller evaluates the actions undertaken by the stakeholder. Some actions may be preferred over other, e.g. have less costs, or are optimal according to some optimality criterion. Controllers can be based on evaluators of the paths from the initial state to the current state.

Creation steps are the most complex steps in modelling. They typically consist of an orientation or review substep, of a development step performed in teams, and of a finalisation substep. Creation steps are composed of a number of substeps that can be classified into:

- Review of the state-of-affairs: The state of the development is reviewed, evaluated, and analysed. Obligations are derived. Open development tasks can be closed, rephrased or prioritised.
- Study of documents and resources: Available documents and resources are checked whether they are available, adequate and relevant for the current step, and form a basis for the successful completion of the step.
- Discussions and elicitation with other partners: Discussions may be informal, interview-based, or systematic. The result of such discussions is measured by some quality criteria such as trust or confidence. They are spread among the partners with some intention such as asking for revision, confirmation, or extension of the discussion.
- Recording and documentation of concepts: The result of the step is usually recorded in one work product or consistently recorded in number of work products.
- Classification of concepts, requirements, results: Each result developed is briefly examined individually and in dependence of other results from which it depends and to which it has an impact.
- Review of the development process: Once the result to be achieved is going to be recorded the work product is examined whether is has the necessary and sufficient quality, whether is must be revised, updated or rejected, whether there are conflicts, inconsistencies or incompleteness or whether more may be needed. If the evaluation results in requiring additional steps or substeps then the step or the substep is going to be extended by them.

This cycle follows the ‘maieutics’ cycle proposed by Sokrates [10,13]. Maieutics uses the methods of irony (or background elicitation), of induction towards generalisation and of definition (or concept development). For the modelling stage we derive the fol-
lowing correspondence between the maieutics cycle and the modelling stage:

<table>
<thead>
<tr>
<th>problem initiation</th>
<th>problem differentiation and understanding</th>
<th>problem evaluation and selection by relevancy</th>
<th>model justification and consensus</th>
<th>experimentation and field exploration</th>
<th>application</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O$ origin</td>
<td>$\Phi(O)$ origin properties</td>
<td>$\Psi(G)$ modelling objectives</td>
<td>$G$ model properties</td>
<td>$\Psi(Y)$ implementation objectives</td>
<td>$Y$</td>
</tr>
</tbody>
</table>

We therefore arrive at a modelling process in Figure 4 that refines the general workflow in Figure 3.

![Figure 4](image_url)  
**Figure 4.** The sub-workflows for description-prescription modelling processes

We may zoom-in into these sub-workflows. For instance, one of the most interesting steps is step (3) in the model activities. This step consists of a number of substeps. Since the rigor or conceptualisation stage is orthogonal, the framework to database design in [24] is extended by conceptualisation in Figure 5.

![Figure 5](image_url)  
**Figure 5.** Sub-steps of the modelling and concept derivation step in the modelling sub-workflow
Conceptualisation is based on the notion of concepts introduced in [8,27]. Design science [5,6,12,28] uses the rigor cycle as one of its three cycles aiming at model development. The rigor cycle has not yet been defined. We can use the maieutics approach for the development of a conceptualisation cycle and arrive at a conceptualisation cycle displayed in Figure 6.

![Conceptualisation Cycle Diagram](image)

**Figure 6.** The sub-workflow for conceptualisation steps within the modelling step

6. Conclusion

Models are artifacts that can be specified within a \((W^4+W^{17}H)\)-frame based on the classical rhetorical frame introduced by Hermagoras of Temnos\(^7\). They are primarily characterised by \(W^4\): wherefore (purpose), whereof (origin), wherewith (carrier, e.g., language), and worthiness ((surplus) value). Secondary characterisation \(W^{17}H\) is given by:

- user or stakeholder characteristics: by whom, to whom, whichever;
- characteristics imposed by the application domain: wherein, where, for what, wherefrom, whence, what;
- purpose characteristics characterising the solution: how, why, whereto, when, for which reason; and
- additional context characteristics: whereat, whereabout, whither, when.

Modelling is the art, the systematics and the technology of model (re)development and model application. It uses model activities and techniques. This paper is going to be extended by more specific aspects of the modelling art. One of them is modelling competency. Information systems modelling competency means being able to autonomously and insightful carry out all aspects of the modelling process in a certain context. It thus means the ability to identify relevant questions, variables, relations, postulates and assumptions in a given application domain, to translate these into computer engineering notions and to interpret and validate the model in relation to the purpose of modelling, as well as to analyse or compare given models by investigating the postulates and assumptions being made, checking properties and scope of given models, etc. This competency can be instantiated to the different ways of working in Figure 2.

---

\(^7\)Quis, quid, quando, ubi, cui, quem ad modum, quibus adminiculis \((W^7):\) Who, what, when, where, why, in what way, by what means). The Zachman frame uses a simplification of this frame.
References


Contents

17  The Theory of Conceptual Models, the Theory of Conceptual
    Modelling and Foundations of Conceptual Modelling           1
    Bernhard Thalheim
    17.1  Towards a Theory of Conceptual Models and Conceptual
          Modelling ................................................................. 1
          17.1.1  Artifacts, Concepts and Intentions ....................... 3
          17.1.2  Dimensions of Models and Modelling ..................... 5
          17.1.3  Postulates of Modelling ................................... 10
          17.1.4  Artifacts and Models ...................................... 12
    17.2  The Theory of Conceptual Models ................................. 13
          17.2.1  Conceptual Models and Languages .......................... 13
          17.2.2  Concepts and Models ...................................... 20
          17.2.3  Information Exchange of Stakeholders based on Models . 21
          17.2.4  Mappings among Models and Originals .................... 24
          17.2.5  Development Phases that use Models ..................... 28
          17.2.6  Properties of the Models-Origin and the Models-
                  Reflections Analogies ................................. 30
    17.3  Conclusion .............................................................. 32
    References ........................................................................ 33
Chapter 17
The Theory of Conceptual Models, the Theory of Conceptual Modelling and Foundations of Conceptual Modelling

Bernhard Thalheim

Abstract Conceptual modelling is a widely applied practice and has led to a large body of knowledge on constructs that might be used for modelling and on methods that might be useful for modelling. It is commonly accepted that database application development is based on conceptual modelling. It is however surprising that only very few publications have been published on a theory of conceptual modelling. Modelling is typically supported by languages that are well-founded and easy to apply for the description of the application domain, the requirements and the system solution. It is thus based on a theory of modelling constructs. Modelling is ruled by its purpose, e.g., construction of a system, simulation of situations in real world, theory construction, explanation of phenomena, documentation of an existing system, etc. Modelling is also an engineering activity with engineering steps and engineering results. It is thus engineering.

17.1 Towards a Theory of Conceptual Models and Conceptual Modelling

Models are different for different purposes. We may develop a model for analysis of an application domain, for construction of a system, for communicating about an application, for assessment, and for governance. These different purposes result in different goals and task portfolios. Models are an essential part in computer science. While preparing a survey on models we realised that computer science uses more than 50 different models. Analysing these different models we discover however four commonalities of these models:
Purpose: Models and conceptual models are governed by the purpose. The model preserves the purpose. Therefore the purpose is an invariant for the modelling process.

Mapping: The model is a mapping of an origin. It reflects some of the properties observed or envisioned for the origin.

Language as a carrier: Models are using languages and are thus restricted by the expressive power of these languages. Candidates for languages are formal or graphical languages, media languages, illustration languages, or computer science constructions.

Value: Models provide a value or benefit based on their utility, capability and quality characteristics.

The purpose of a model covers a variety of different intentions and aims. Typical purposes are:

- **perception support** for understanding the application domain,
- **explanation and demonstration** for understanding an origin,
- **preparation** to management and handling of the origin,
- **optimisation** of the origin,
- **hypothesis verification** through the model,
- **construction** of an artifact or of a program,
- **control** of parts of the application,
- **simulation** of behaviour in certain situations, and
- **substitution** for a part of the application.

Depending of the purpose we shall use different models.

Models are author-driven and addressee-oriented. Therefore, the association between an origin and the model as a mapping is given in Figure 17.1.

![Fig. 17.1 The origin-model-author-addressee relationship for models](image)

A model is typically a schematic description of a system, theory, or phenomenon of an origin that accounts for known or inferred properties of the origin and may be used for further study of characteristics of the origin. Conceptual modelling aims to create an abstract representation of the situation under investigation, or more precisely, the way users think about it. Therefore conceptual models enhance models by concepts that are commonly sharing within a community or at least between the stakeholders involved into the modelling process.
This chapter extends the theory of conceptual models, conceptual modelling and modelling act proposed by [20] and systematises the four main dimensions of models: purpose, mapping, language, and value. It is based on an explicit application of concepts and constructs of languages.

The value of a model is given by the objective value and by the subjective value.

Models as enduring, justified and adequate artifacts: The artifact can be qualified as an ‘objective’ model, if

1. the artifact is adequate by certain notion of ‘adequacy’,
2. is reusable in a rule system for new models and refinement of models and
3. is not equivalent to models, which can be generated with the aid of facts or preliminary models in the particular inventory of models by a rule system.

Models as the state of comprehension or knowledge of a user: Models are used for comprehension of a user or stakeholder. Therefore, a model can be understood as the knowledge of a user. Different kinds of to know are:

1. The state or fact of knowing.
2. Familiarity, awareness, or understanding gained through experience or study.
3. The sum or range of what has been perceived, discovered or learned.
4. Learning; erudition: teachers of great knowledge.
5. Specific information about something.
6. Carnal knowledge.

Therefore, we conclude that it is necessary to deliver models as enduring, justified and adequate artifacts to users depending on context, users demands, desiderata and intention, whereby these aspects are supported by the environment, the profile and tasks of the users. The tasks of users require a special model quality.

17.1.1 Artifacts, Concepts and Intentions

17.1.1.1 The Conceptual Model Space

At the same time we may distinguish four different aspects of conceptual models. Conceptual model use concepts. Therefore, the model space is characterised through (1) its origin, (2) its concepts, (3) its representation of model elements, and (4) its comprehension by users or stakeholders involved. Model elements cannot be considered in an isolated form. For this reason we imagine to use model chunks as a suite of model elements consisting of images of pieces observed for the origin, concepts, representations and comprehension. These aspects are interdependent from each other. Figure 17.2 displays the conceptual model space.
17.1.1.2 Intentions Driving Modelling

Modelling and especially conceptual modelling is not yet well understood and misinterpreted in a variety of ways. The first goal of the chapter is to overcome some myths of conceptual modelling such as:

1. Modelling equals documentation.
2. You can think everything through from the start.
3. Modelling implies a heavyweight software process.
4. You must “freeze” requirements and then you can start with modelling.
5. Your model is carved in stone and changes only from time to time.
6. You must use a CASE tool.
7. Modelling is a waste of time.
8. The world revolves around data modelling.
9. All developers know how to model.
10. Modelling is independent on the language.

The second goal of this chapter is the development of a framework to modelling. Modelling is based on an explicit choice of languages, on application of restrictions, on negotiation and on methodologies. Languages are defined through their syntax, their semantics, and their pragmatics. We typically prefer inductive expression formation based on alphabets and behaviour defined on expressions. Restrictions depend on logics (deontic, epistemic, modal, belief, preferences) and use shortcuts, ambiguities, and ellipses. Negotiation support management or resolution of conflicts and the development of strategies to overcome these strategic, psychological, legal, and structural barriers. Development methodology are based on pragmatism and on paradigms. Since modelling is an activity that involve a number of actors the choice of languages becomes essential. Modelling is a process and based on modelling acts. These modelling acts are dependent from the purpose of modelling itself. Therefore we can distinguish different modelling acts such as understand, define, conceptualise, communicate, abstract, construct, refine, and evaluate. De-
pending on the purpose of model development we might use modelling act such as construct and evaluate as primary acts.

The third goal of this chapter is to draw attention to explicit consideration of modelling properties both for the models themselves and for the modelling acts. This side of conceptual modelling is often only considered in an implicit form. The modelling process is governed by goals and purposes. Therefore, we must use different models such as a construction model, a communication model or a discussion model. Modelling is restricted by the application context, the actor context, the system context and the theory and experience context. These kinds of context restrict the model and the modelling process.

### 17.1.2 Dimensions of Models and Modelling

#### 17.1.2.1 Main Dimensions of Modelling

We use commonalities observed above for Computer Science for an introduction of main dimensions of models and modelling that considers

- **purpose** ("wherefore") of models and modelling with the intentions, goals, aims, and tasks that are going to be solved by the model,
- **mapping** ("whereof") with a description of the solution provided by the model, the characterisation of the problem, phenomena, construction or application domain through the model,
- **language** ("wherewith") with a careful selection of the the carrier or cargo [10] that allows to express the solution, the specification of the world or the construction, and
- **value** ("worthiness") of a model by explicit statement of the internal and external qualities, and the quality of use, e.g. explicit statement of invariance properties relating the model to its associated worlds or by preservation properties that are satisfied by the model in dependence on the associated worlds.

These main dimensions of models and modelling govern the model and the modelling acts. There are extended by secondary dimensions that are used to shape and to adapt the model. We are going to discuss these dimensions in the sequel after a discussion of the ruling dimension: the purpose dimension.

The task of model development is never completed (p’tamoi rh’ousi ‘the rivers flow’). Models are changing artifacts due to changes imposed by

- **scope insight** for conscious handling of restriction, capabilities, opportunities, guiding rules for convenience, for completion, refinement, and extension, development plans for partial delivery of models, partial usage and deployment, theories supporting development of models, quality characteristics for model completion, model evolution, model engineering, and mappings styles for mapping models among abstraction layers.
17.1.2.2 The Purposes Dimension

The purpose dimension is ruling the development of models and the application of models. The main reason for using a model is to provide a solution to a problem. We thus may describe the purpose by characterisation of the solution to the problem by the model. We may distinguish a number of concerns such as

- the impact of the model ("where to") for a solution to a problems,
- the insight into the origin's properties ("how") by giving details how the worlds is structured or should be structured and how the functionality can be described,
- restrictions on applicability and validity ("when") of a model for some specific solutions, for the validity interval, and the lifespan of a model,
- providing reasons for model value ("why") such as correctness, generality, usefulness, comprehensibility, and novelty, and
- the description of functioning of a model ("for which reason") based on the model capacity.

This general characterisation of purposes of models can be specialised for database and information system models. The main purposes of information system models are given within Gregor's taxonomy [6]:

I. Analysis: Says what is.
   The model does not extend beyond analysis and description. No causal relationships among phenomena are specified and no predictions are made. It thus provides a description of the phenomena of interest, analysis of relationships among those constructs, the degree of generalisability in constructs and relationships and the boundaries within which relationships, and observations hold.

II. Explanation: Says what is, how, why, when, and where.
   The model provides explanations but does not aim to predict with any precision. There are no testable propositions. The model provides an explanation of how, why, and when things happened, relying on varying views of causality and methods for argumentation. This explanation will usually be intended to promote greater understanding or insights by others into the phenomena of interest.

III. Prediction: Says what is and what will be.
   The model provides predictions and has testable propositions but does not have well-developed justificatory causal explanations. It states what will happen in the future if certain preconditions hold. The degree of certainty in the prediction is expected to be only approximate or probabilistic in IS.

IV. Explanation and prediction: Says what is, how, why, when, where, and what will be.
   The model provides predictions and has both testable propositions and causal explanations. A special case of prediction exists where the model provides a description of the method or structure or both for the construction of an artifact (akin to a recipe). The provision of the recipe implies that the recipe, if acted upon, will cause an artifact of a certain type to come into being.
V. Design and action: Says how to do something.

The model gives explicit prescriptions (e.g., methods, techniques, principles of form and function) for constructing an artifact.

Based on this characterisation of the purpose we infer a number of requirements to languages used for modelling and to modelling methodologies:

Means of representation: The model must be represented physically in some way: in words, mathematical terms, symbolic logic, diagrams, tables or graphically. Additional aids for representation could include pictures, models, or prototype systems.

Constructs: These refer to the phenomena of interest in the model (Dubins “units”). All of the primary constructs in the model should be well defined. Many different types of constructs are possible: for example, observational (real) terms, theoretical (nominal) terms and collective terms.

Statements of relationship: These show relationships among the constructs. Again, these may be of many types: associative, compositional, unidirectional, bidirectional, conditional, or causal. The nature of the relationship specified depends on the purpose of the model. Very simple relationships can be specified.

Scope: The scope is specified by the degree of generality of the statements of relationships (signified by modal qualifiers such as “some”, “many”, “all”, and “never”) and statements of boundaries showing the limits of generalizations.

Causal explanations: The model gives statements of relationships among phenomena that show causal reasoning (not covering law or probabilistic reasoning alone).

Testable propositions (hypotheses): Statements of relationships between constructs are stated in such a form that they can be tested empirically.

Prescriptive statements: Statements in the model specify how people can accomplish something in practice (e.g., construct an artifact or develop a strategy).

17.1.2.3 The Artifact Dimension

The main product of modelling is the model, i.e. an artifact that is considered to be worth for its purpose by the author. The model can, for instance, be used for the description of the world of origins or for the prescription of constructions. There are a number of explicit choices an author makes and that rule application of models. Modelling of information systems

depends on the abstraction layer, e.g. requirements, specification, realisation or implementation layer,
depends on chosen granularity and precision of the work product itself,
depends on resources used for development of a model such as the language,
depends on level of separation of concern such as static/dynamic properties, local/global scope, facets,
depends on quality properties of the input, e.g. requirements, completeness, conciseness, coherence, understandability,
depends on decomposition of the work products in ensembles of sub-products, and satisfies quality characteristics such as quality in use, internal quality, and external quality.

17.1.2.4 The User Dimension

A number of users are involved into the development of models. The user dimension thus reflects intentions, the understanding, the comprehension and other characteristics of users in a variety of roles, e.g.,

the role of an author (“by whom”) that results in reflections of the educational level, application of templates, pattern or reference models,

the role of an addressee (“to whom”) that restricts the utilisation of the model or that supports the extended application beyond the purpose originally intended, and

the role of broad public (“whichever”) that develops a common understanding of the model depending on the group or the culture of the public.

Users are different and thus modelling has different results because of

attitudes of users and their preferences

the ability to understand, to model, to reason, to survey, to communicate with others, to analyse, to construct systems, to validate or to verify or to test models, to use or develop documentations

mastering of complexity, improvements, and realisations,

knowledge, skills, competency of users for representing world or for coping with representations,

restricted expressivity due to restricted leads or due to human preference of local reasoning instead of global consideration of all properties of an artifact,

experience to cope with varieties of problem solutions through generic problem solving, and

referential solutions to be used for solution of similar problems together with refinement of the given approach.

One important relationship among the users is the form of partnership during the development or application of models The partnership is characterised by roles during activities such as stakeholder, developer, consultant, supplier, contractor, documentation developers, or finally business user,

by the practised collaboration partnership based on communication acts, cooperation business processes, and coordination agreements,

by the teamwork during all activities with separation of different tasks,

by historical people such as teachers, legacy (better heritage) developers, coders, and builders of earlier models.
Finally, the user dimension imposes an important restriction to the development, the application, the understanding of models: Models tend to be too large for a singleton person.

### 17.1.2.5 The Domain Dimension

The domain dimension clarifies:

- the domain depending on models purpose ("for what") such as an application domain, properties reflected or neglected,
- the scope to specific elements ("what") that are considered to be typical and whose properties should be reflected,
- the attention within the domain depending on models purpose ("where") that limits the model to the ‘normal’ aspects,
- the orientation of the domain ("wherefrom") that restricts the attention and the directions for the current activities supported by the model,
- the sources for origins or the infrastructure considered ("whence") for the model, and
- the restrictions of the world ("wherein") associated with the model.

A typical influence of the application domain can be illustrated by an example in [10]. Areas in Königsberg are connected through bridges. The question is whether there is a path that uses each bridge but only once. Such path is called Euler path.

![Application Domain](image)

The two models display the same problem as in the original domain. We might also use a tree model that enumerates each starting point and associates a node with its predecessor and potential next point if there is an unused bridge. This model is inadequate for the general problem whether there is an Euler path within a topographical model. The main quality property for the models is the preservation of the Euler path problem.

### 17.1.2.6 The Context Dimension

The context dimension is typically used for restricting a model to a specific scope and thus limits the general utilisation of models. It additionally requires an explicit consideration of these restrictions if the model is used outside its main application.
area. Context abstraction is a useful vehicle for restricting attention. Typical specific context restrictions to models are

the worlds ("whereat") considered for the model such as the world that is currently accepted, the world that will be never considered, and the world that might be considered in future in dependence on the model value,
the background knowledge ("whereabout") that forms the model and limits the model, and envisioned evolution paths ("whither") for the adaptation of model to future requirements.

17.1.3 Postulates of Modelling

17.1.3.1 General Properties of Models

This discussion can be summarised in a number of postulates that are of importance for models, modelling, and modelling acts.

*Mapping* property: Each model has an origin and is based on a mapping from the origin to the artifact.

*Truncation* property: The model lacks some of the ascriptions made to the original and thus functions as an Aristolean model by abstraction of irrelevant.

*Pragmatic* property: The model use is only justified for particular model users, tools of investigation, and period of time.

*Amplification* property: Models use specific extensions which are not observed for the original.

*Distortion* property: Models are developed for improving the physical world or for inclusion of visions of better reality, e.g. for construction via transformation or in Galilean models.

*Idealisation* property: Modelling abstracts from reality by scoping the model to the ideal state of affairs.

The first three properties are based on Stachowiaks theory of models [16, 12]. The fourth property has been formulated in [17]. The fifth property has been discussed in [9]. The sixth property has been developed within Natural Sciences, e.g. Chemistry.

17.1.3.2 Prescription by Models and Description for Models

This association between origin and artifacts is depicted in Figure 17.3.

In a similar form we may describe the application of models for construction of other artifacts such as software and hardware. In this case, the reflection of the postulate is given in Figure 17.4.
17 The Theory of Conceptual Models

**Fig. 17.3** The association of worlds and resulting postulates of models

<table>
<thead>
<tr>
<th>Purpose: description of the world</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>worlds of origins</strong></td>
</tr>
<tr>
<td><strong>worlds of artifacts considered to be models</strong></td>
</tr>
<tr>
<td>(1) mapping</td>
</tr>
<tr>
<td>(2) truncation property</td>
</tr>
<tr>
<td>(3) pragmatic property</td>
</tr>
<tr>
<td>(6) idealisation property</td>
</tr>
<tr>
<td>descriptions</td>
</tr>
<tr>
<td>observations</td>
</tr>
<tr>
<td>experiments</td>
</tr>
<tr>
<td>language</td>
</tr>
<tr>
<td>assessment of model value in dependence on purpose</td>
</tr>
<tr>
<td>derivation of properties of interest in dependence on purpose</td>
</tr>
</tbody>
</table>

**Fig. 17.4** The association of models with artifacts constructed with models as a blueprint

<table>
<thead>
<tr>
<th>Purpose: prescription of constructions</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>world of artifacts used as models</strong></td>
</tr>
<tr>
<td><strong>worlds of utilisation of models</strong></td>
</tr>
<tr>
<td>(6) calibration mapping</td>
</tr>
<tr>
<td>(4) amplification property</td>
</tr>
<tr>
<td>(5) distortion property</td>
</tr>
<tr>
<td>hypotheses</td>
</tr>
<tr>
<td>experiments</td>
</tr>
<tr>
<td>language</td>
</tr>
<tr>
<td>descriptions</td>
</tr>
<tr>
<td>observations</td>
</tr>
<tr>
<td>learning by/with the model</td>
</tr>
<tr>
<td>prediction with the model</td>
</tr>
<tr>
<td>evaluation of the value of the model depending on purpose</td>
</tr>
<tr>
<td>application for theory development and hypothesizing</td>
</tr>
</tbody>
</table>
17.1.3.3 The Model Capacity

The model
- is based on an analogy of structuring, functionality, or behaviour,
- satisfies certain model purposes, and
- provides a simple handling or service or consideration of the things under consideration.

Any model is therefore characterised by a model capacity that describes
- how the model provides some understanding of the origin or can be used depending on the purpose,
- how the model provides an explanation of demonstration through auxiliary information and thus makes the origin or the associated elements easier or better to understand,
- how the model provides an indication and facilities for making properties viewable,
- how the model allows to provide variations and support optimisation,
- how the model supports verification of hypotheses within a limited scope,
- how the model supports construction of technical artifacts,
- how the model supports control of things in reality, or
- how the model allows a replacement of things of reality and acts as a mediating means.

17.1.3.4 Resulting Restrictions to be Accepted by Stakeholders

Models are governed by their purpose. They may support this purpose or not. They have a value and may thus be used depending on their capacity.

Prohibition of estrangement: Models serve a purpose and cannot be used in general outside the scope of the purpose.

17.1.4 Artifacts and Models

The four aspects of the conceptual model space in Figure 17.2 are interwoven. Models use artifacts. Models have their specific representation. Models are supported by conceptualisations. The interrelationship between models, representations and concepts should be very flexible. We can assume that models may use different representations or artifacts. Artifacts may contain sub-artifacts. Conceptual models are based on concepts. Concepts may be typical for a model within a certain degree of typicality. Concepts may consist of sub-concepts. Therefore, we may associate a model with one concept that has its sub-concepts. We assume that concepts are independent on representations. Additionally, we may assume that representations are dependent on the language and some ontology to be used. They are typically
commonly accepted or shared within a community or culture. This understanding leads to a structure displayed in Figure 17.5.

![Diagram](image)

**Fig. 17.5** The association between artifacts, representations, and concepts

### 17.2 The Theory of Conceptual Models

#### 17.2.1 Conceptual Models and Languages

##### 17.2.1.1 The Language Dimension

Models are represented by *artifacts* that satisfy the pragmatic purposes of users. We restrict our discussion within this subsection to formal languages that are typically used for conceptual models. In this case, artifacts are linguistic expressions that describe the model. Linguistic expression are built within a language with some understanding. Therefore, artifacts use syntax, semantics and pragmatics built within the chosen language.

Semantic annotation in current content management systems is usually restricted to preselected ontologies and parameter sets. Rich conceptual data models are only available in more sophisticated systems. They are adapted to certain application domains incorporate preselected and tailored ontologies.

The model-artifact association is agreed within a community. This community is based on a web of knowledge of their members (see Chapter ??).
17.2.1.2 Languages used for Representation of Models

Languages are the carrier for models. We may accept a logics approach to semiotics and define the language similar to Chapter ???. Each constructive language is based on a signature, on a set of base items, and a set of constructors. The language consists of words that are allowed due to well-formedness constraints. Languages $\mathcal{L}$ are used for a number of reasons, e.g. reasoning, representation, illustration, etc. These reasons are driven by the model purposes in our case.

We may now use a subset of words and accept those as postulates for a model. Each model to be considered faithful or useful must satisfy these postulates. We may develop different artifacts as a potential models. We are however interested in some properties that these models must satisfy. We therefore develop a general understanding of artifacts that are used for models within a language. Postulates must be explicitly given. They might be changed whenever the purpose of modelling is changing. They do not restrict models to one artifact. Instead we might also use a number of artifacts in parallel. Figure 17.6 displays the relationship between an artifact and its postulates and properties.

![Fig. 17.6 Artifacts with a language, their properties and postulates](image)

Constructive languages thus support

- to prescribe postulates that restrict the judgement that an artifact can be accepted as a model,
- to scope our attention to those artifacts that can be considered for a model or for parts of a model, and
- to orient the user on certain properties that are of interest for the purpose of modelling.

This approach is very general. It can be applied in many areas. Consider, for instance, the following table:

<table>
<thead>
<tr>
<th>$\mathcal{L}_G$</th>
<th>$\Psi(G)$</th>
<th>corresponds</th>
<th>$G$</th>
<th>scope</th>
<th>$\Phi(G)$</th>
<th>postulates</th>
<th>accepted</th>
<th>supports</th>
<th>observation</th>
<th>properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>logics</td>
<td>axioms</td>
<td>satisfy</td>
<td>structure</td>
<td>satisfy</td>
<td>essential properties</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$\mathbb{N}$</td>
<td>Peano axioms</td>
<td>satisfy</td>
<td>standard model</td>
<td>derivable</td>
<td>Peano arithmetics</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>empirism</td>
<td>postulates</td>
<td>accepted</td>
<td>artifact</td>
<td>supports</td>
<td>observation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>technics</td>
<td>construction</td>
<td>enforce</td>
<td>product</td>
<td>has</td>
<td>properties</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
This approach also carries classical approaches used in mathematical logics:

<table>
<thead>
<tr>
<th>( \mathcal{L}_G )</th>
<th>( \Psi(G) )</th>
<th>corresponds</th>
<th>( G )</th>
<th>scope</th>
<th>( \Phi(G) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>logics</td>
<td>axioms</td>
<td>satisfy</td>
<td>structure</td>
<td>consider</td>
<td>essential properties of ( G )</td>
</tr>
<tr>
<td>logics</td>
<td>axioms</td>
<td>satisfy</td>
<td>structure</td>
<td>satisfy</td>
<td>relevant theorems of ( (\mathcal{L}_G, \Psi(G)) )</td>
</tr>
</tbody>
</table>

We, therefore, may define a theory by the pair \( (\mathcal{L}_G, \Psi(G)) \). The model class \( \text{Mod}_{\mathcal{L}_G}(\Psi(G)) \) is defined to be the set of all structures that satisfy \( \Psi(G) \). A structure \( G \) is a model of \( \Psi(G) \) and thus \( G \in \text{Mod}_{\mathcal{L}_G}(\Psi(G)) \). A theory \( \text{Th}(\mathcal{K}) \subseteq \mathcal{L}_G \) is given for a class \( \mathcal{K} \) of structures and consists of all language expressions that are satisfied by each of the structures in \( \mathcal{K} \).

The same approach can also be used for conceptual modelling:

<table>
<thead>
<tr>
<th>( \mathcal{L}_G )</th>
<th>( \Psi(G) )</th>
<th>corresponds</th>
<th>( G )</th>
<th>scope</th>
<th>( \Phi(G) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>database</td>
<td>requirements</td>
<td>realise</td>
<td>DB schema</td>
<td>satisfy</td>
<td>integrity constraints</td>
</tr>
<tr>
<td>workflow</td>
<td>requirements</td>
<td>realise</td>
<td>WF schema</td>
<td>satisfy</td>
<td>integrity constraints</td>
</tr>
</tbody>
</table>

Therefore ingredients used for modelling of databases, information systems and workflow systems are languages, restrictions, negotiations for the property to be a model, and methodologies for artifact development. Languages are given with syntactics, semantics, and pragmatics. We typically use inductive expression formation based on alphabets. It also supports the description of behaviour defined on expressions. Restrictions depend on logics to be used, e.g., first-order hierarchical predicate logics [18], deontic logics, epistemic logics, modal logics, logics for belief reasoning of for preference derivation. Negotiations provide a means to identify, define analyse barriers and manage or resolve conflicts. Methodologies of development are based on engineering approaches and are guided by certain pragmatism and a number of paradigms.

### 17.2.1.3 Principles of Language Use

Languages may however also restrict modelling. This restriction may either be compensated by over-development of language components or by multi-models. Over-development of language components has been observed within the theory of integrity constraints in the relational model of data. More than 95 different and necessary classes of integrity constraints have been developed. Multi-modelling is extensively used for UML. The Sapir-Whorf hypothesis [22] results in the following principle:

**Principle of linguistic relativity:** Actors skilled in a language may not have a (deep) understanding of some concepts of other languages. This restriction leads to problematic or inadequate models or limits the representation of things and is not well understood.
The principle of linguistic relativity is not well understood. Therefore, we illustrate this principle by a discussion that highlights the deficiencies we need to overcome.

### 17.2.1.4 The Matter of Language Choice

Let us consider a well-known example: traffic light control. Given a crossroad, e.g. consisting of two streets (north-south, east-west) with an intersection and of traffic lights that direct traffic. We assume at the first glance that traffic lights might switch from red to green and from green to red. We also might assume that both opposite cross lights show the same colour. Software engineering approaches, Petri net approaches, process algebra approaches etc. typically start with a model for each cross light. Next the interdependence among the state changes is either modelled through integrity constraints or through implicit modelling constructs. The best solution we know so far is the Petri net solution depicted in Figure 17.7. It uses an external timer and switches between the directions.

![Fig. 17.7 Traffic control based on Petri nets](image)

This model neither scales nor has a good development, internal, or dynamic quality. The extension to yellow colour is an intellectual challenge as well the extension to more flexible directing. This example is typically chosen due to everyday life experience of the students despite its complete inadequacy. This pitfall has already discussed in [8] who tried to find a better solution based on state change diagrams and failed due to complex integrity constraints. Implementations neglect this solution and implement a completely different solution.

The main reason for the poor quality and the conceptual and implementation inadequacy is its wrong attitude, wrong scope, wrong abstraction, and wrong granularity.

Explicit assumptions can also be derived for the traffic light control application. We first need to decide whether the analogy to real-life is based on the behaviour of the entire system or on the combined behaviour of the behaviour of components. This distinction directly implies a choice between a model that represents the entire application as one system and the components as its elements (local-as-view model) and a model that combines local models to a global one (global-as-view model). All conceptual solutions known in literature use the global-as-view model. In this case
state tables and (ASM) state transfer rules like the following ones are used:

\[
\begin{array}{c|c|c|c|c|c|c|c}
\text{Controller} & \text{location} & \text{state} & \text{clock} & \text{reset} & \text{switch} \\
\hline
\text{e} & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
\end{array}
\]

\[
\text{if Switch}(e) \text{ then UPDATE}(e, \text{collocated}(e)); \text{ CHANGESWITCH}(e).
\]

These states and rules may obey a number of rather complex integrity constraints.

We might prefer the local-as-view approach. States reflect the entire state of the crossroad, i.e. \text{NSredEWgreen}, \text{NSredEWred}, \text{NSgreenEWred}. The last state reflects that the north-south direction is open and the east-west direction is closed. We might add the state \text{NSredEWred} for representation of the exception state and the state \text{NSnothingEWnothing} for the start and the end state. The state \text{NSgreen-EWgreen} is a conflict state and thus not used for the model.

The other decisions discussed in this section can now made in a similar manner. We choose a full controller for all lights. We might however choose a local controller for each cross light. In this case, the local controller is nothing else than a view on the global schema. The model we propose supports simulation as well as understanding, reasoning, variation and extension, optimisation and technical artifacts. The workmanship includes also a collection of extensions that seems to be probable such as people calling a state change, exceptional situations, yellow lights, specific directions etc. The local schemata are based on views and on the master-slave principle. Update is central and display is local.

This model also allows to explicitly specify which states are never under consideration, which states are a ‘must’ and which states are used for later extensions. We further assume that reality can be mapped to discrete variables, clocks are based on linear time logics, control is restricted to vehicle and pedestrian direction gauge. This model extends also the real life application by adding a global, combined state. Its main advantage is however that the context conditions for correct traffic lights for all coexisting directions are directly coded into the model domain space and thus do not need any explicit support.

The local-as-view model is based on a two-layer architecture that uses a global schema and local view schemata. The extended ER model [18] provides a number of opportunities for the representation of hierarchies. A typical hierarchy in our traffic light application is the specialisation hierarchy for states. Since states can be multiply classified depending on the day time and the week day we might choose the bulk representation for the classification of types through a \text{StateKind} instead of explicit specialisation types. State changes may also be classified in a similar way. We might however prefer to separate calls for state change made by pedestrians and triggering of state changes through a times. Based on these choices we derive modelling activities for the database schemata and workflow rules. We explicitly specify properties and binding among the global and local schemata, e.g. master-slave binding.

The given application can be specified through different modelling concepts. These modelling concepts provide a number of alternatives and a number of opportunities. Therefore the ER schema in Figure 17.8 represents one of the possible schemata for the global schema. The state changes and the pedestrian calls are not recorded after they have been issued. The scheduler is based on this schema and
might use workflow diagrams, trigger rules or ASM rules [2] for specification of BPMN diagrams. We can use a generic pattern approach that supports extensions, e.g. for kinds of states and kinds of state changes. Typical examples are the following:

\[
\text{CHANGE ACTION} := \text{getState}; \text{choosePossibleStateChange(state)}; \text{apply(possibleStateChange(state))}
\]

\[
\text{ALARM ACTION} := \text{on alarm changeStateToErrorState}
\]

\[
\text{CLOCK} := \text{on tick observeWhetherChangeRequired}
\]

\[
\text{NORMAL ACTION} := \text{if change = true then CHANGE ACTION}
\]

\[
\text{PEDESTRIAN CALL} := \text{on callAtPoint(cp) CHANGE NEXT STEP ISSUED AT(cp)}.
\]

In a similar form we specify views for local display.

### 17.2.1.5 Pragmatics that cannot be Neglected

While syntax and semantics of language expressions has been well explored, its pragmatics apart from the use of metaphors has not. Pragmatics is part of semiotics, which is concerned with the relationship between signs, semantic concepts and things of reality. This relationship may be pictured by the so-called semiotics triangle. Main branches of semiotics are syntactics, which is concerned with the syntax, i.e. the construction of the language, semantics, which is concerned with the interpretation of the words of the language, and pragmatics, which is concerned with the current use of utterances by the user and context of words for the user. Pragmatics permits the use of a variety of semantics depending on the user, the application and the technical environment. Most languages defined in Computer Science have a well-defined syntax; some of them possess a well-defined semantics; few of them use pragmatics through which the meaning might be different for different users.

Syntactics (often called syntax) is often based on a constructive or generative approach: Given an alphabet and an set of constructors, the language is defined as the set of expressions that can be generated by the constructors. Constructions may be defined on the basis of grammatical rules.

Semantics of generative languages can be either defined by meta-linguistic semantics, e.g. used for defining the semantics of predicate logics, by procedural or referential semantics, e.g. operational semantics used for defining the semantics of
programming languages, or by convention-based semantics used in linguistics. Semantics is often defined on the basis of a set of relational structures that correspond to the signature of the language.

Pragmatics has to be distinguished from pragmatism. Pragmatism means a practical approach to problems or affairs. Pragmatism is the “balance between principles and practical usage”. Here we are concerned with pragmatics, which is based on the behaviour and demands of users, therefore depends on the understanding of users.

Let us consider an example for a well-known class of constraints in databases.

A similar observation can be made for multivalued, join, inclusion, exclusion and key dependencies. Functional dependencies are the most known class of database constraints and commonly accepted. They are one of the most important class of equality-generating constraints.

Given a type $R$ and substructures $X, Y$ of $R$.

The functional dependency $R : X \rightarrow Y$ is valid in $R^C$ if $o|_Y = o'|_Y$ whenever $o|_X = o'|_X$ for any two objects $o, o'$ from $R^C$.

Functional dependencies carry at least five different but interwoven meanings. The notion of the functional dependency is thus overloaded. It combines different properties that should be separated:

- **Explicit declaration of partial identification:** Functional dependencies are typically explicitly declaring a functional association among components of types.
  
  The left hand attribute uniquely identify right side attributes, i.e. $X^{\text{Ident}} \rightarrow Y$.
  
  Identification can either be based on surrogate or on natural attributes [1].

- **Tight functional coupling:** Functional dependencies may also be numerical constraints. We denote such constraints by $X^{\text{Num}} \rightarrow Y$. Another denotation is based on cardinality constraints [18].

- **Semantic constraint specific for the given application:** Constraints may be stronger than observed in usual life since the application has a limited scope and allows us to strengthen the constraint. In this case, constraints restrict the application only to those cases in which the left side has only one associated right side value despite that this restriction may not be valid for any application.
  
  We denote this case by $X^{\text{Sem}} \rightarrow Y$.

- **Semantical unit with functional coupling:** *Semantical units* are those reducts of a type that are essential in the given application. Their components cannot be separated without loosing their meaning. Semantical units may have their inner structure. This structure tightly couples dependent object parts to those that are determining them [18]. We denote this coupling by $X^{\text{Unit}} \rightarrow Y$.

- **Structural association among units:** Semantical units may allow a separation of concern for certain elements. Their separation supports a more flexible treatment while requiring that the dependent part cannot exist without the determining part. If this dependence is functional we may represent such by the constraint $X^{\text{Struct}} \rightarrow Y$. 

17.2.2 Concepts and Models

Concepts are the basis for conceptual models. They specify our knowledge what things are there and what properties things have. Concepts are used in everyday life as a communication vehicle and as a reasoning chunk. Concepts can be based on definitions of different kinds. Therefore our goal for the development of a theory of conceptual modelling and of conceptual models can only be achieved if the conceptual model definition covers any kind of conceptual model description and goes beyond the simple textual or narrative form.

A general description of concepts is considered to be one of the most difficult tasks. We analysed the definition pattern used for concept introduction in mathematics, chemistry, computer science, and economics. This analysis resulted in a number of discoveries:

- Any concept can be defined in a variety of ways. Sometimes some definitions are preferred over others, are time-dependent, have a level of rigidity, are usage-dependent, have levels of validity, and can only be used within certain restrictions.
- The typical definition frame we observed is based on definition items. These items can also be classified by the kind of definition. The main part of the definition is a tree-structured structural expression of the following form:
  \[
  \text{SpecOrderedTree} (\text{StructuralTreeExpression} (\text{DefinitionItem}, \text{Modality(Sufficiency, Necessity)}, \text{Fuzziness}, \text{Importance}, \text{Rigidity}, \text{Relevance}, \text{GraduationWithinExpression}, \text{Category})).
  \]

- Concepts typically also depend on the application context, i.e. the application area and the application schema. The association itself must be characterised by the kind of association.

Concepts are typically hierarchically ordered and can thus be layered. We assume that this ordering is strictly hierarchical and the concept space can be depicted by a set of concept trees. A concept is also dependent on the community that prefers this concept. A concept is typically given through an embedding into the knowledge space of users involved. The schema in Figure 17.9 displays the general structure for content definition. This schema also covers all aspects discussed in [13]. This schema extends the relationship between artifacts, representations and concepts introduced in Figure 17.5.

Concept gathering can be understood as a technique which combines concept representation [5, 13, 19] and (algorithmic) learning approaches. A concept gathering system is based on:

- a set of concepts and available experience \( C \),
- a set of domain knowledge \( D \),
- a set of representable meta knowledge \( M \),
- a set of learning goals \( G \), and
- a set of representable hypotheses \( H \).

The set of representable knowledge and concepts is denoted by \( R = C \cup D \cup M \cup G \cup H \).
The concept gathering system \((\gamma, \lambda, \nu, \mathcal{C}, \mathcal{R})\) consists of

- a concept generator \(\gamma : \mathcal{C} \times \mathcal{R} \rightarrow \mathcal{C}\),
- a learning function \(\lambda : \mathcal{C} \times \mathcal{R} \rightarrow \mathcal{H}\), and
- an evaluator \(\nu : \mathcal{C} \times \mathcal{R} \rightarrow \mathcal{Q}\) where \(\mathcal{Q}\) denotes set of quality characteristics.

A run of the concept gathering system results in

- a concept detection sequence \(C_1, C_2, \ldots, C_f\) with \(C_i \in \mathcal{C}\) and
- a learning sequence \(R_0, R_1, R_2, \ldots, R_f\) with \(R_i \in \mathcal{R}\) where \(R_0\) denotes the initial knowledge and \(R_f\) denotes the final knowledge.

The run is typically recorded and is dependent on the concepts gathered so far. Additionally, the concept gathering system records

- the background knowledge of the user \(\mathcal{B} \subseteq \mathcal{D} \cup \mathcal{M} \cup \mathcal{G}\) and
- the actual available knowledge \(\mathcal{B} \cup \mathcal{H}'\).

17.2.3 Information Exchange of Stakeholders based on Models

Stakeholders such as the author of a model and the addressee for a model use model in a variety of ways. The main use of models is information (or knowledge) exchange among stakeholders. There are several definitions for information.

- The first category of these definitions is based on the mathematical notion of entropy. This notion is independent of the user and thus inappropriate in our project context.
The second category of information definitions bases information on the data a user has currently in his data space and on the computational and reasoning abilities of the user. Information is any data that cannot be derived by the user. This definition is handy but has a very bad drawback. Reasoning and computation cannot be properly characterised. Therefore, the definition becomes fuzzy.

The third category is based on the general language understanding of information. Information is either the communication or reception of knowledge or intelligence.

Information can also be the act of informing against a person. Finally information is a formal accusation of a crime made by a prosecuting officer as distinguished from an indictment presented by a grand jury.

All these definitions are too broad.

We are thus interested in a definition that is more appropriate for the internet age.

Information as processed by humans,

- is carried by data
- that is perceived or noticed, selected and organized by its receiver,
- because of his subjective human interests, originating from his instincts, feelings, experience, intuition, common sense, values, beliefs, personal knowledge, or wisdom,
- simultaneously processed by his cognitive and mental processes, and
- seamlessly integrated in his recallable knowledge.

Therefore, information is directed towards pragmatics, whereas content may be considered to highlight the syntactical dimension. If content is enhanced by concepts and topics, then users are able to capture the meaning and the utilisation of the data they receive. In order to ease perception we use metaphors or simply names from a commonly used namespace. Metaphors and names may be separated into those that support perception of information and into those that support usage or functionality. Both carry some small fraction of (linguistic) semantics.

The information transfer from a user A to a user B depends on the users A and B, their abilities to send and to receive the data, to observe the data, and to interpret the data. Let us formalise this process. Let $s_X$ denote the function user by a user $X$ for data extraction, transformation, and sending of data. Let $r_X$ denote the corresponding function for data receival and transformation, and let $o_X$ denote the filtering or observation function. The data currently considered by $X$ is denoted by $D_X$. Finally, data filtered or observed must be interpreted by the user $X$ and integrated into the knowledge $K_X$ a user $X$ has. Let us denote by $i_X$ the binary function from data and knowledge to knowledge. By default, we extend the function $i_X$ by the time $t_{i_X}$ of the execution of the function.
Thus, the data transfer and information reception (or briefly information transfer) is formally expressed by

\[ I_B = i_B(o_B(r_B(s_A(D_A)))), K_B, t_{iB} \].

In addition, time of sending, receiving, observing, and interpreting can be taken into consideration. In this case we extend the above functions by a time argument. The function \( s_X \) is executed at moment \( t_{sX} \), \( r_X \) at \( t_{rX} \), and \( o_X \) at \( t_{oX} \). We assume \( t_{sA} \leq t_{rB} \leq t_{oB} \leq t_{iB} \) for the time of sending data from \( A \) to \( B \). The time of a computation \( f \) or data consideration \( D \) is denoted by \( t_f \) or \( t_D \), respectively. In this extended case the information transfer is formally expressed by

\[ I_B = i_B(o_B(r_B(s_A(D_A), t_{sA}, t_{rB}), t_{oB}, K_B, t_{iB})). \]

The notion of information considers senders, receivers, their knowledge and experience. Figure 17.10 displays the multi-layering of communication, the influence of explicit knowledge and experience on the interpretation.

The communication act is specified by

- the communication message with the content or content chunk, the characterisation of the relationship between sender and receiver, the data that are transferred and may lead to information or misinformation, and the presentation,
- the sender, the explicit knowledge the sender may use, and the experience the sender has, and
- the receiver, the explicit knowledge the receiver may use, and the experience the receiver has.

![Fig. 17.10 Dimensions of the communication act](image-url)
17.2.4 Mappings among Models and Originals

17.2.4.1 Modelling Supported by Mapping

So far two of the four main dimensions have been founded. Let us now consider the mapping between two worlds: source world and target world. Examples of source-target pairs are

- origins from the real real world are mapped to an artifact that is considered to be a model,
- elements of an artifact that serves as a model to a realisation of the artifact by an implementation, and
- elements of one model to elements of another model.

The first mapping is typically based on a description of the origins that is represented by a model about these origins. The second mapping is typically based on a prescription made by the model for a realisation of the model by a technical artifact. The third mapping has been used above for the association between the topographical model and the graph model for the Königsberg bridge problem on page 9.

We observe also other pairs of such mappings depending on the purpose. For instance, documentation uses an artifact to be documented and another artifact that documents essential elements of the first artifact. It typically extends the first artifact for pragmatic rules for exploitation of the first artifact and by behavioural scenario as examples of deployment. It bases the documentation also on an idealisation of the first artifact. A similar association may be developed for the other purposes: Perception support for understanding the application domain; explanation and demonstration for understanding; preparation to management and handling of the original; optimisation of the application domain operating; hypothesis verification through the model; control of parts of the application; simulation of behaviour in certain situations; substitution for a part of the application.

We may now combine these observations with the treatment of languages introduced in Figure 17.5. We add to this treatment the logical framework. It defines for a set of artifacts that are of interest a language and a theory that can be used for reasoning on properties of the artifacts and for explicit consideration of postulates. In this case, we need to consider two languages: the language of the origin of the mapping and the language of the target of the mapping. Figure 17.11 displays the mappings between the different artifacts.

Furthermore we observe another important property of the mapping:

**Principle of conservative stability of source properties:** The properties of the source are relatively stable. It results in some kind of target conservativeness: Any target artifact revision that cannot be reflected already in the current set of properties of the source is entirely based on explicit changes considered for the first artifact.

**Principle of consistency of mapping:** Main properties of the source artifact should be stable for the target artifact.

These principles can be extended by other principles for mappings that are often assumed but not necessary:
Conceptualization principle: Only aspects of the source artifact should be taken into account when constructing the target artifact.

95% principle: All the relevant aspects of the source artifact should be described in the target artifact. We notice that this principle is weaker than the classical 100% used in software engineering. It better reflects the engineering component of modelling.

Formalization principle: Target artifacts should be formalisable in order to be realisable.

Semiotic principle: Target artifacts should be easily interpretable and understandable.

Correspondence condition for knowledge representation: The target artifact should be such that the recognizable constituents of it have a one-to-one correspondence to the relevant constituents of source artifact.

Invariance principle: Target artifacts should be constructed on the basis of such entities detected for the source artifact that are invariant during certain time periods within world of the source artifact.

Construction principle: In order to construct a good target artifact it is important first to construct relevant sub-artifacts and then to search for connections between them.

The main postulate for the mapping is however the

Postulate of purpose invariance: The purpose of the modelling activity can be realised through the target artifact. It can both be considered for the source artifact as well as for the target artifact.

This postulate requires that the mapping must obey an invariance property for the purpose. It has a number of implications:

- The mapping is a realisation of an analogy property.
- It is possible to re-map properties observed for the target artifact to the source artifact if those are not caused by idealisation, distortion or amplification.
The target artifact can also be used for other mapping with different intentions and goals. We shall discuss specific forms of analogies below.

As an example, we may refine Figure 17.11 to classical ER modelling displayed in Figure 17.12. This picture allows also to reason on the advantages and on the disadvantages of the ER modelling approach.

### 17.2.4.2 Modelling with a Manifold of Models

Typical modelling follows a number of purposes. The UML is an example of model suites that are used at the same time. Class diagrams reflect the structuring of object sets and the functionality provided for the object sets. Object diagrams may be based on class diagrams. They may however reflect also things in the application domain as a combined set of class objects. Interaction diagrams reflect the message and control flow among objects in the first setting of object diagrams.

A similar picture is observed for models that are developed for different purposes. Consider, for instance, models that have been developed for construction of a technical artifact, for communication and discussion of properties among stakeholders, for documentation, and for analysis. Figure 17.13 display the manifold of models developed for different purposes for an origin.

Fig. 17.12 The relationship between application domain world and Entity-Relationship modelling language world

![Diagram](image-url)
This picture displays a pitfall of multi-language modelling. The models may consider different aspects of the origin, they may contradict and they may not be integratable. For instance, if we use class diagrams, statecharts, activity diagrams, time diagrams, component diagrams, interaction diagrams and other s within a software development team then integration of different aspects might become infeasible. Therefore we may apply two rather rigid modelling restrictions: The main principles for the multi-language modelling are therefore the following the following ones:

**Principle of coherence of models:** Models are coherent if their common reflection of the origin is consistent, i.e. sub-models that reflect the same properties of the origin can be injective mapped to each other.

**Principle of origin property completeness:** Models partially reflect the same set of properties of the origin. None of the model uses properties that are different from properties that can potentially be used for another model.

Chapter ?? already considered co-evolution of models and introduced a formalism to handle coherence. Coherence describes a fixed relationship between the models in a model suite. Two models are coherent when each change in one of the models is propagated to the other model. This change transfer implicitly assumes that the integrity constraints of the corresponding model types remain to be valid. They are non-coherent if there is a random or changing relationship. We aim in an explicit specification of the association schema and use an explicit specification of the collaboration among models. For instance, the master-slave association or collaboration propagates any change of the master to its slaves. Slaves do not have any right the change the master without consensus with the master.

If we enforce these principles then the model variety can be handled in a simpler and feasible way. Figure 17.14 displays the advantages of a coherent set of models which is based on a complete set of properties of the origin. It allows to introduce a binding between these models. This binding can be mapped to a contract in the sense of Chapter ??.

**Fig. 17.14** Coherent models reflecting different purposes on a complete set of origin properties
This approach directly results in a *coordination of models* on the basis of *separation of aspects*.

### 17.2.5 Development Phases that use Models

#### 17.2.5.1 Description through Models and Prescription by Models

One of the main combined purposes of models is the description of an application domain that is subsequently uses the developed model as a prescription of realisation of a technical artifact. Conceptual modelling adds to the model a number of

---

**Fig. 17.15** Modelling for description of the origin and as the basis of realisation by a technical artifact
concepts that are the basis for an understanding of the model and for the explanation of the model to the user.

This two-phase development cycle of technical artifacts is the kernel of conceptual modelling of information systems and database systems. There are different other forms of this two-phase database system development. We may use the association between the model and the application for model refinement and model evolution. Models are typically parameterised. Therefore, these parameters may be adopted to the actual or intended situation. Models are integrated during bottom-up modelling. They can be refined, optimised, validated, or improved before the realisation phase starts. Verification is typically checking the properties of a model and the the properties of a realisation. Testing checks the relationship between properties in the application domain and properties of the realisation.

17.2.5.2 Reasoning Support for Modelling

Design science [7] has been targeting on an explicit support for the modelling process. This support includes an explicit consideration of the quality of the model, of the quality of the modelling process, and of the quality of supporting theories. We may combine the informal discussions with our approach and separate the modelling acts by the things that are under consideration. Figure 17.16 displays the different ways of working during a database systems development. We use here the two-phase model: Description followed by prescription.

Fig. 17.16  Reasoning processes and reasoning support for description followed by prescription
These different “ways of working” characterise
- the modelling acts with its specifics; [20]
- the foundation for the modelling acts with the theory that is going to support this act, the technics that can be used for the start, completion and for the support of the modelling act, and the reasoning techniques that can be applied for each step;
- the partner involved with their obligations, permissions, and restrictions, with their roles and rights, and with their play;
- the aspects that are under consideration for the current modelling acts;
- the consumed and produced elements of the artifact that are under consideration during work;
- the resources that must be obtained, that can be used or that are going to be modified during a modelling act.

Consider, for instance, the way or requiring. It includes specific facets such as
- to command, to require, to compel, and to make someone do something with supporting acts such as communicating, requesting, bespeaking, ordering, forbidding, prohibiting, interdicting, proscribing;
- to ask, to expect, to consider obligatory, to request and expect with specific supporting acts such as transmitting, communicating, calling for, demanding;
- to want, to need, to require, to have need of with supporting acts of wanting, needing, requiring;
- to necessitate, to ask, to postulate, to need, to take, to involve, to call for, to demand to require as useful, to just, or to proper.

The ways of functioning, understanding, elicitation, modelling, reasoning, assessment, and construction can be characterised in a similar form.

The rigor stage may be replaced by other stage that support different purposes. We concentrated on prescription and construction of new systems. Another application is model refinement similar to two-model representation of the Königsberg bridge problem on page 9.

Design science aims at another kind of model refinement by adding more rigor after evaluation of a model. This refinement is essentially model evolution. Another refinement is the enhancement of models by concepts. This refinement is essentially a ‘semantification’ or conceptualisation of the model. Experimentation and justification of models is a third kind of adding rigor to (conceptual) models.

17.2.6 Properties of the Models-Origin and the Models-Reflections

Analogies

Figure 17.1 bases modelling on a quadruple of origin, model, author and addressee. The origin-model association as well as the the experimentation, construction or reasoning with models is based on an explicit consideration of the notion of an analogy between the model and the origin or the model and its reflection in theories,
constructions, hypotheses, or illustrations. Therefore we need a characterisation of analogies.

Analogies are statements on similarity, statements of adjustment, statements of emphases. They characterise the approximation made by the model. These characterisations can be given by:

- **Degree of structural analogy:** This grad characterises the degree of similarity of either the original with the model or of the model with its reflection.
- **Degree of qualitative analogy:** This grad characterises to which degree the character and constitution is reflected.
- **Degree of structural adjustment:** This degree characterises to which extent the structure is considered despite from the later use.
- **Degree of qualitative adjustment:** This degree characterises what is going to be used for the later exploitation and what part is not going to be used.
- **Degree of functional adjustment:** This degree characterises the functions that are considered and the functions that are not considered.
- **Degree of contrast and emphasis:** This degree provides a means to specifically consider the distortion, amplification and idealisation made by the model.

Degree measurement is based on the ratio between the good or bad cases against all possible cases. We may consider a number of ratio measurements. **Recall evaluation** relates the number of positive observations to the number of all possible observations. **Fallout evaluation** measures the negative observations against the number of all possible observations. **Precision evaluation** typically measures the relevant observations similar to recall observations. Measurement functions often use metrics. Another kind of measurement uses **model-checking** functions that are based on predicates that evaluate certain properties. These properties can be used to decide whether a work product is consistent and can be refined for work products at the implementation layer.

Additionally, we need an approach to provide **tolerance** of the results and deviations from the either the origin or the realisations.

Additionally we need a logics that provides us with a means for reasoning on analogy and for using analogy for transfer of derived statements and properties into the other domain. This directly results in a **logics of analogical reasoning**. Such logics have been developed in artificial intelligence and logics research. We may use, for instance, derivation rules for a source object $s$ and a target object $t$ of the following form

$$ t \approx_{\alpha} s \quad \alpha \models \beta $$

$$ \beta(t) := \beta(s) $$

This rules allows us to conclude that whenever the source and the target object are analogue based on a certain predicate $\alpha$ and the predicate $\alpha$ entails another predicate $\beta$ then we may transfer the value for $\beta$ for the source to the target.

Another such rule is the following one:

$$ t \approx_{\alpha} s \quad \alpha(s) $$

$$ \diamond \alpha(t) $$
If we know that \( s \) and \( t \) are \( \alpha \)-analog and we observe the value \( \alpha(s) \) then it is plausible to assume \( \alpha(t) \).

We also may incorporate lifting relations or bridge rules between an origin and the model or the model and its reflections. These rules must consider a certain context between for both the model and the origin or both the model and its reflection. Therefore we use mappings between two languages with an additional context parameter for a context \( \mathcal{C} \):

\[
\mathcal{F} : \mathcal{L}_1 \times \mathcal{C} \rightarrow \mathcal{L}_2.
\]

If we consider formulas \( \alpha \) in context \( C_i \) then rules need to be extended:

\[
\begin{array}{l}
(\alpha_1, i_1) \ldots (\alpha_n, i_n) \\
\hline
(\alpha, i)
\end{array}
\]

Such rules state that \( (\alpha_1 \ldots \alpha_n) \) in their contexts \( (C_{i_1} \ldots C_{i_n}) \) imply \( \alpha \) in the context \( C_i \) is the applicability condition \( i \varphi \) is valid.

Such rules are considered in calculi of plausible reasoning that incorporate abduction and induction. Plausible reasoning uses inference patterns which can yield uncertain conclusions even if the premises are certain. It is typical for situations in which the knowledge is incomplete. The modelling situation is based on incomplete information or incomplete knowledge.

The most important property for the analogy relationship is adequacy. Adequacy requires the satisfaction of the following four properties.

1. **Similarity** between origin and model or between model and reflection in dependence on the purpose of the model is based on an explicitly given similarity relation that allows also to reason on the restrictions of similarity, i.e. in the case of origin and model we may base similarity in subsets of properties \( \Phi(O) \) and \( \Phi(M) \) that are defining the similarity. Similarity support the deployment of the model instead of the origin or the reflection instead of the model in all situations in which there is a similarity between the two sides.

2. **Regulative factors** form a standardisation on the basis of exact rules which are given within a well-defined system. These rules allow to derive the properties and do not result in exceptions that cover specific properties of the target that are not observed for the source.

3. **Copiousness** is based on the capacity of the model. The model is a far better medium for reasoning about the origin or the reflection. It allows to simpler draw conclusions, to simpler reason about properties and to simpler state postulates.

4. **Simplicity** of the model is based on its concentration on the essential and relevant properties in dependence on the models purpose.

**17.3 Conclusion**

The aim of the chapter has not been to develop an entire theory of conceptual modelling. Instead we aimed in the development of a programme for the theory. We
described the general purpose of this theory, demonstrated how different paradigms can be selected, and showed which scope, modelling acts, modelling methods, modelling goals and modelling properties might be chosen for this theory.

The programme requires far more work. The theory needs a variable taxonomy that allows a specialisation to languages chosen for a given application domain, must be based on a mathematical framework that allows to prove properties, must be flexible for coping with various modelling methodologies, must provide an understanding of the engineering of modelling, and finally should be supported by a meta-CASE tool that combines existing CASE to a supporting workbench.

The findings of this chapter are:

- A model is a representation of something for a purpose of somebody developed by somebody else.
- Each model is author-driven and addressee-oriented, is aspect-related, is purpose-specific, is limited in space, context and time, and is perspective.
- The model quality is also given by those elements that are not observed for the origin or not realised in the reflection or realisation.
- Due to amplification, distortion and idealisation, models cannot be used outside their purpose. If the purpose changes then the model should change as well.
- Models are similar to concepts; they are abstract and concrete; they associate worlds, e.g., the world of origins and models.
- Conceptual models are similar to other systems context- and utilisation-dependent. They have their value within the purpose range.

Models are imperfect and diverge from the real world. They are incomplete, have a different behaviour, and also use other kinds of errors. Imperfectness is based on exceptional states (events, time lags), on incompleteness to limitations of the language and consideration, and errors either based on real errors and exceptional states or based on biases.

A theory of conceptual modelling can be based on a system of guiding principles. This paper shows that at least three guiding principles must be explored in detail:

- **Internal principles** are based on a set of ground entities and ground processes.
- **Bridge principles** explain the results of conceptual modelling in the context of their usage, for instance for explanation, verification/validation, and prognosis.
- **Engineering principles** provide a framework for mastering the modelling process, for reasoning on the quality of a model, and for termination of a modelling process within a certain level of disturbance tolerance (error, incompleteness, open issues to be settled later, evolution).

**References**


The Science and Art of Conceptual Modelling

Bernhard Thalheim

Christian-Albrechts-University Kiel, Computer Science Institute, 24098 Kiel, Germany, thalheim@is.informatik.uni-kiel.de, http://www.is.informatik.uni-kiel.de/~thalheim

Abstract. Conceptual modelling is one of the central activities in Computer Science. Conceptual models are mainly used as intermediate artifact for system construction. They are schematic descriptions of a system, a theory, or a phenomenon of an origin thus forming a model. A conceptual model is a model enhanced by concepts. The process of conceptual modelling is ruled by the purpose of modelling and the models. It is based on a number of modelling acts, on a number of correctness conditions, on modelling principles and postulates, and on paradigms of the background or substance theories. Purposes determine the (surplus) value of a model. Conceptual modelling is performed by a modeller that directs the process based on his/her experience, education, understanding, intention and attitude. Conceptual models are products that are used by other stakeholders such as programmers, learners, business users, and evaluators. Conceptual models use a language as a carrier for the modelling artifact and are restricted by the expressiveness of this carrier.

This paper aims at a discussion of a general theory of modelling as a culture and an art. A general theory of modelling also considers modelling as an apprenticeship and as a technology. It is thus an art. Modelling is one of the main elements of Computer Science culture that consists of commonly accepted behaviour patterns, arts, consensus, institutions, and all other supporting means and thoughts.

Keywords conceptual modelling, modelling workflow, foundations of modelling.

1 The Triptychon of Model as an Artifact, Modelling as an Activity and Modelling as an Art and Science, thus as a Culture

Conceptual modelling is a widely applied practice in Computer Science and has led to a large body of knowledge on constructs that might be used for modelling and on methods that might be useful for modelling. It is commonly accepted that database application development is based on conceptual modelling. It is however surprising that only very few publications have been published on a theory of conceptual modelling. We continue the approach [36, 37] and aim in a theory of modelling within this paper. An approach to a theory of models has been developed in [37]. An approach to a theory of model activities is discussed in [36].
1.1 Three Guiding Concerns during Conceptual Modelling

Conceptual modelling is often only discussed on the basis of modelling constructs and illustrated by some small examples. It has however three guiding concerns:

1. **Modelling language constructs** are applied during conceptual modelling. Their syntax, semantics and pragmatics must be well understood.
2. **Application domain gathering** allows to understand the problems to be solved, the opportunities of solutions for a system, and the requirements and architecture that might be prescribed for the solution that has been chosen.
3. **Engineering** is oriented towards encapsulation of experiences with design problems pared down to a manageable scale.

The first concern is handled and well understood in the literature. Except few publications, e.g. [2], the second concern has not yet got a sophisticated and well understood support. The third concern has received much attention by data modelers [29] but did not get through to research literature. It must therefore be our goal to combine the three concerns into a holistic framework.

A model is nothing else than a material or virtual artifact that is used or can be used in dependence on some objectives, purposes and functions by somebody. As an artifact, somebody else was acting as a developer with some intentions and goals. The objective of models, the purpose of models and the function of models are often considered to be synonymous. There are however differences that should be taken into account [13, 18]. The objective of a model is the change of reality through the model that can be reached by activities and is a goal of stakeholders, i.e., it is a ternary relation between two states of reality and humans. The purpose is based on the objective and presupposes the existence of instruments through which the state change can be reached. The purpose is bound to intentions of stakeholders to reach this objective by activities. The function of a model is based on the use or deployment of a model in a practice. It is thus bound to processes in which the model has its applications (‘Sprachspiel’ (language game) [39] or deployment game).

Objectives can be abstract. Purposes need however a meaningful description. The purpose thus includes the intention, the meaning, the function, and the tasks. Typical functions of a model are deployment for illustration or explanation, for verification or validation, as a deputy for another artifact or surrogate within an investigation or experimentation process, for tests of theories, as basis for simulation, within a learning process, and last but not least for construction of systems augmenting current reality. The model plays a part within these processes. These parts are typically categorised by roles.

1.2 Implications for a Theory of Conceptual Modelling

The three concerns of conceptual modelling must be integrated into a framework that supports the relevant concern depending on the modelling work progress. The currently most difficult concern is the engineering concern. Engineering is inherently concerned with failures of construction, with incompleteness both in specification and in coverage of the application domain, with compromises for all quality dimensions, and with problems of technologies currently at hand.
At the same time, there is no universal approach and no universal language that cover all aspects of an application, that have a well-founded semantics for all constructions, that reflect any relevant facet in applications, and that support engineering. The choice of modelling languages is often a matter of preferences and case, empirical usage, evolution history, and supporting technology.

1.3 Differences between ‘Model’, ‘To Model’ and ‘Modelling’

The conceptions of model, of the activity ‘to model’ and of modelling are often used as synonyms. We must however distinguish these conceptions for a theory of models, a theory of model activities and a theory of the modelling process.

Based on the notions in the Encyclopedia Britannica [23] we distinguish between the conception of a model, the conception of a model activity, and the conception of modelling processes.

The model as an artifact: The model is something set or held for guidance or imitation of an origin and is a product at the same time. Models are enduring, justified and adequate artifacts from one side. From the other side, models represent the state of comprehension or knowledge of a user.

To model as an activity: ‘To model’ is a scientific or engineering activity beside theoretical or experimental investigation. The activity is an additive process. Corrections are possible during this activity. Modelled work may be used for construction of systems, for exploration of a system, for definition and negotiation, for communication, for understanding and for problem solving.

Modelling as a systematically performed technological process: Modelling is a technique of systematically using knowledge from computer science and engineering to introduce technological innovations into the planning and development stages of a system. At each stage the modeller is likely to ask both why and how, rather than merely how. Modelling is thus based on paradigms and principles.

Additionally, the notion of model may be used in an adjective sense as serving as or capable of serving as a pattern or being a usually miniature representation of something. This notion is often used for sample representations such as a ‘model chair’. Another notion of the model that is not of interest within this paper is the miniature representation of something.

1.4 The Simultaneity of Art, Culture, Technology and Techniques in Modelling

Modelling can be understood as a technique\(^1\) or as a technology\(^2\). [23] distinguishes between science and technology: Technology is the systematic study of techniques for making and doing things; science is the systematic attempt to understand and interpret

\(^1\) I.e., the fashion, manner, mode, modus, system, way, wise in which a system etc. is mastered. Techniques consist of methods of accomplishing a desired aim.

\(^2\) Technology is an element of engineering. It consists of the practical application of knowledge especially in a particular area. It provides a capability given by the practical application of knowledge. Therefore, it is a manner of accomplishing a task especially using technical processes, methods, or knowledge.
the world. While technology is concerned with the fabrication and use of artifacts, science is devoted to the more conceptual enterprise of understanding the environment, and it depends upon the comparatively sophisticated skills of literacy and numeracy.

At the same time, modelling is an art. Modelling is a highly creative process. It requires skills in planning, making, or executing. It is often claimed that it is not to be formalisable. It requires deep insight into the background as well as skills, careful simplification, experience and ingenuity. Due to the variety of viewpoints, modelling is also based on judgement and clever selection with different alternatives.

Modelling in one of the main activities in Computer Science. It consists of commonly accepted and practised behavior patterns, arts, consensus, institutions, and all other products of human work and thought. Turning to culture, culture is based on the capacity for rational or abstract thought. The meaning of abstraction is not sufficiently explicit or precise. The term symboling has been proposed as a more suitable name for assigning to things and events certain meanings that cannot be grasped with the senses alone.

This culture is learned and shared within communities which have their own behaviour pattern and approaches. It is not yet a science since it heuristically uses operational and/or scientific terms.

1.5 Orientation of this Paper

This paper explores modelling as an art and culture. We base the discussion on a theory of models and of modelling activities. We abstract therefore in this paper from micro-, meso-, macro-models or model suites used in many natural sciences or model suites, e.g., model ensembles used in UML or OWL. We do not yet consider modelling competency or MDA/D. We do not yet consider modelling competency. All notions used in this paper are based on culture. The main goal of this paper is to show that modelling requires apprenticeship and technology. The orientation towards an expert mode can be reached if modelling is based on systematic development and if modelling is considered to be a craft of modelling activities. This approach shows that modelling incorporates design science in a wider sense as it has been considered in the literature.

Art requires capability, competence, handiness, and proficiency. Art is based on finesse, i.e. on refinement or delicacy of workmanship. Models and art share a Janus head evaluation: The judgement of beauty evaluates the model within a community of business users. The judgement of the sublime evaluates the model against its technical realisation. A model has thus both an extrinsic and intrinsic value.

The notion of culture combines at least eight facets: (1) cultivation, tillage; (2) the act of developing the intellectual and moral faculties especially by education; (3) expert care and training; (4) enlightenment and excellence of taste acquired by intellectual and aesthetic training; (5) acquaintance with and taste as distinguished from vocational and technical skills; (6) integrated pattern of human knowledge, belief, and behavior that depends upon man’s capacity for learning and transmitting knowledge to succeeding generations; (7) the customary beliefs, social forms, and material traits of a group; and (8) the set of shared attitudes, values, goals, and practices that characterizes a company or corporation. Culture requires different practices: education, enlightenment, erudition, learning from one side, gentility, manners, discrimination, taste from the other side, and sophistication, class, and elegance from a third side.
We base our ideas on our observations on model developments for very large database schemata and very large database systems\(^5\). Such systems require a well organised modelling process. They must be evolution-prone and revision-prone. The paper concentrates thus one of the main workflows: description of application worlds followed by prescription for system worlds and specification of systems.

2 The World of Models

2.1 The Conception of the Model

Models are artifacts selected by a stakeholder based on some stakeholder judgement or perception and governed by the purpose. Models can thus be characterised by main dimensions:

- **purpose** ("wherefore") of models and modelling with the intentions, goals, aims, and tasks that are going to be solved by the model,
- **result of mapping** ("whereof") with a description of the solution provided by the model, the characterisation of the problem, phenomena, construction or application domain through the model,
- **language** ("wherewith") with a careful selection of the carrier or cargo\(^{[15]}\) that allows to express the solution, the specification of the world or the construction, and
- **value** ("worthiness") of a model by explicit statement of the internal and external qualities, and the quality of use, e.g. explicit statement of invariance properties relating the model to its associated worlds or by preservation properties that are satisfied by the model in dependence on the associated worlds.

These four dimensions are driven by two context dimensions: the application domain dimension rules the scope and (explicit and implicit) disregard of the model; the user or stakeholder dimension governs the viewpoint, orientation and background of users involved. The mapping associates the origin and the artifact. As far as we are interested in modelling of information systems, we may use a (semi-)formal language for the artifact.

\(^5\) Due to our involvement into the development and the service for the CASE workbenches (DB)\(^2\) and ID\(^2\) we have collected a large number of real life applications. Some of them have been really large or very large, i.e., consisting of more than 1,000 attribute, entity and relationship types. The largest schema in our database schema library contains of more than 19,000 entity and relationship types and more than 60,000 attribute types that needs to be considered as different. Another large database schema is the SAP R/3 schema. It has been analyzed in 1999 by a SAP group headed by the author during his sabbatical at SAP. At that time, the R/3 database used more than 16,500 relation types, more than 35,000 views and more than 150,000 functions. The number of attributes has been estimated by 40,000. Meanwhile, more than 21,000 relation types are used. The schema has a large number of redundant types which redundancy is only partially maintained. The SAP R/3 is a very typical example of a poorly documented system. Most of the design decisions are now forgotten. The high type redundancy is mainly caused by the incomplete knowledge on the schema that has been developed in different departments of SAP.
These main dimensions of models and modelling govern the model and the modelling acts. They are extended by secondary dimensions that are used to shape and to adapt the model: the artifact, the user, the context and the application domain dimensions. The mapping dimension is discussed in [36]. The value dimension can be described based on [10]. The purpose dimension is ruling and governing both the development of models and the application of models. This tight governance is caused by the main aim of a model: to provide a solution to a problem.

2.2 The Model as a Physical or Virtual Artifact

The main product of modelling and model activities is the model, i.e. an artifact that is considered to be worth for its purpose by the author. The model can, for instance, be used for the description of the world of origins or for the prescription of constructions. There are a number of explicit choices an author makes and that rule applications of models. Modelling of information systems depends on the abstraction layer, e.g. requirements, specification, realisation or implementation layer, depends on chosen granularity and precision of the work product itself, depends on resources used for development of a model such as the language, depends on level of separation of concern such as static/dynamic properties, local/global scope, facets, depends on quality properties of the input, e.g. requirements, completeness, conciseness, coherence, understandability, depends on decomposition of the work products in ensembles of sub-products, and satisfies quality characteristics such as quality in use, internal quality, and external quality.

The task of model development is never completed (τα πάντα ρεί, ‘the rivers flow’; narrative: everything flows). Models are changing artifacts due to changes imposed by scope insight for conscious handling of restriction, capabilities, opportunities, guiding rules for convenience, for completion, refinement, and extension, development plans for partial delivery of models, partial usage and deployment, theories supporting development of models, quality characteristics for model completion, model evolution, model engineering, and mapping styles for mapping models among abstraction layers.

2.3 The Purpose Dimension

The purpose dimension is ruling and governing the model, the development process and the application process because of the main reason for using a model is to provide a solution to a problem. Therefore the purpose is characterised by the solution to the problem provided by the model. We may distinguish a number of concerns such as the impact of the model ("whereto") for a solution to a problem, the insight into the origin’s properties ("how") by giving details how the world is structured or should be structured and how the functionality can be described,
restrictions on applicability and validity ("when") of a model for some specific solutions, for the validity interval, and the lifespan of a model, providing reasons for model value ("why") such as correctness, generality, usefulness, comprehensibility, and novelty, and the description of functioning of a model ("for which reason") based on the model capacity.

The task of model development is never completed (τα παντα ρέι, ‘the rivers flow’, narrative: everything flows). Models are evolving artifacts due to changes imposed by

- scope insight for conscious handling of restriction, capabilities, opportunities,
- guiding rules for convenience, for completion, refinement, and extension,
- development plans for partial delivery of models, partial usage and deployment,
- theories supporting development of models,
- quality characteristics for model completion, evolution and engineering, and
- mappings styles for mapping models among abstraction layers.

2.4 The Language Dimension

Models are represented by artifacts that satisfy the pragmatic purposes of users. In this case, artifacts are linguistic expressions that describe the model. Linguistic expressions are built within a language with some understanding. Therefore, artifacts use syntax, semantics and pragmatics built within the chosen language.

Models are often expressed through expressions in a formal language $\mathcal{L}_M$. A model should support its objectives. Optimally, these objectives $\Psi(M)$ can be expressed in the same language $\mathcal{L}_M$ that is also used for the model $M$. A model has a number of properties. Some of them are of interest and used for characterisation of the model, e.g., $\Phi(M)$. This characterisation depends on the model and its purpose.

Fig. 1. Artifacts with a language, their properties and objectives within a given language for the artifact

Constructive languages are a special case and support

- the prescription of the objectives or postulates that restrict the judgement that an artifact can be accepted as a model,
- the scope of our attention to those artifacts that can be considered for a model or for parts of a model, and
- the orientation of the user on certain properties that are of interest for the purpose of modelling.
Natural languages have a high potential for deployment of deep semantics and cause a threat to everybody who does not use the language within the same semantical culture. Culture depends on participating stakeholders, their profile (educational, employment, psychological) and includes language, styles of communication, practices, customs, and views on roles and relationships. Deployment of natural language expressions may thus result in misunderstandings. There are two ways to avoid such: the development of a sophisticated ontology that includes all namespaces a user might use or the development of an orthonormalised language [20] that is restricted in expressivity and does not allow misinterpretations.

2.5 The Context Dimensions

The User Dimension. A number of users are involved into the development of models. The user dimension thus reflects intentions, the understanding, the comprehension and other characteristics of users in a variety of roles, e.g.,
the role of an author (“by whom”) that results in reflections of the educational level,
application of templates, pattern or reference models,
the role of an addressee (“to whom”) that restricts the utilisation of the model or that supports the extended application beyond the purpose originally intended, and
the role of broad public (“whichever”) that develops a common understanding of the model depending on the group or the culture of the public.

The Application Domain Dimension and the World of Origins. The application domain consists of people, organisational systems, and technical systems that interact to work towards a goal. This dimension clarifies
the domain depending on models purpose (“for what”) such as an application domain,
properties reflected or neglected,
the scope to specific elements (“what”) that are considered to be typical and whose properties should be reflected,
the attention within the domain depending on models purpose (“where”) that limits the model to the ‘normal’ aspects,
the orientation of the domain (“wherefrom”) that restricts the attention and the issues for the current activities supported by the model,
the sources for origins or the infrastructure (“whence”) considered for the model, and
the restrictions of the world (“wherein”) associated with the model.

3 The World of Modelling Activities

3.1 Workflows Applied in the Model Development and Deployment Process

The purpose dimension governs the workflows applied in conceptual modelling. It also governs the kind of model application. We may distinguish a number of workflows in conceptual modelling such as the following ones:
Construction workflows are based on creation of models (as images, representations or portraits of the origin) that are used for production of systems (using as models as groundwork, background, pattern, standards, prototypes for the system). This kind of model exploitation uses the dichotomy of models as image of an origin and groundwork for a system.

Explanation workflows result in new insights into the world of the origins.

Optimisation-variation workflows result in an improvement and adaptation of the origins.

Verification-validation-testing workflows result in an improvement of the one of the subject, in most cases in an improvement of models.

Reflection-optimisation workflows are typical for mathematical modelling of the world of origins.

Explorative workflows are using models for learning about origins.

Hypothetical workflows are typical for discovery sciences, e.g., sciences used for climate research.

Documentation-visualisation workflows target on better understanding and comprehension of models.

These workflows can be intertwined or shuffled with each other. They may be performed one after another. In this paper we concentrate on the construction (or creation-production) workflow which seems to be central for information systems.

3.2 Conceptual Modelling Activities Governed by its Purpose

Models are developed with different goals, different scope, within different context, with different appeal to the receiver of the model, with different granularity, with different background, and with different realisation forms. Therefore we have to explicitly handle modelling purpose properties.

The mission of modelling is described by scope of the model, the users community, the tasks the model might support, the major and minor purposes satisfied by the model and the benefits obtained from the model for the given user community. The goals of a model are based on the impact of the model, restricted by the relationships among users and their roles they are playing. The brand of the model is given by the who-what-whom-action pattern. The meta-model can be used to provide information about the model such as the context of the model, the context in which the model might be useful to the auditory, the usage of the model, and the restrictions of the model.

3.3 Modelling Acts

It surprises that these model activities are not explicitly handled in most modelling approaches. The same observation can be observed for a declaration of the main goals of the modelling act. Main modelling acts which are the following ones: construct a model, a part of the model, a concept or a judgement, etc. (describe, delineate, fabricate, master), communicate the judgements, the observations, the concepts, etc. (explain, express, verbalise or display),
understand the application domain, the system opportunities, etc. (cognise, identify, recognise, percep),
discover the problems, the potential, the solutions, etc. (interact, identify),
indicate properties of importance, relevance, significance, etc. (visualise, measure, suggest, inform),
virate and optimise a solution, a judgement, a concept, a representation depending on some criteria,
verify or validate or test a model, a solution, a judgement, a representation or parts of those,
control the scope of modelling, the styles or pattern, parts of a model, judgements, etc. (rule, govern, proofread, confirm, restrain, administer, arrange, stratify, standardise),
alternate or compensate or replace or substitute or surrogate models or parts of them, judgements, concepts, etc. (transfer, reassign, evolve, migrate, balance, correct, novate, truncate, ersatz).

The first and last four goals lead to a datalogical model that is structured according to technology. The other goals result in an infological model that is delivered to the needs of the user. We thus use a different frame of reference. The application of the results may thus be descriptive or prescriptive, constitutive or prognosticating, categorical or exegetic or contemplative or formulaic.

4 The Theory of Conceptual Models and Conceptual Modelling

4.1 Conceptual Modelling: Modelling Enhanced by Concepts

An information systems model is typically a schematic description of a system, theory, or phenomenon of an origin that accounts for known or inferred properties of the origin and may be used for further study of characteristics of the origin. Conceptual modelling aims to create an abstract representation of the situation under investigation, or more precisely, the way users think about it. Conceptual models enhance models with concepts that are commonly shared within a community or at least between the stakeholders involved in the modelling process. A general definition of concepts is given in [37]. Concepts specify our knowledge what things are there and what properties things have. Concepts are used in everyday life as a communication vehicle and as a reasoning chunk. Concept definition can be given in a narrative informal form, in a formal way, by reference to some other definitions etc. We may use a large variety of semantics [28], e.g., lexical or ontological, logical, or reflective. [37] introduces a general theory of concepts that can be used for conceptualisation.

Conceptualisation aims at collection of objects, concepts and other entities that are assumed to exist in some area of interest and the relationships that hold among them. It is thus an abstract, simplified view or description of the world that we wish to represent. Conceptualization extends the model by a number of concepts that are the basis for an understanding of the model and for the explanation of the model to the user.
4.2 Conceptual Models

The theory of conceptual models [37] extends the framework [11, 31, 32]. Models can be characterised by four main dimensions: (1) Models and conceptual models are governed by the purpose. The model preserves the purpose. (2) The model is a mapping of an origin. It reflects some of the properties observed or envisioned for the origin. (3) Models use languages and are thus restricted by the expressive power of these languages. (4) Models provide a value or benefit based on their utility, capability and quality characteristics.

The purpose of a model covers a variety of different intentions and aims such as perception support for understanding the application domain, explanation and demonstration for understanding an origin, preparation to management and handling of the origin, optimisation of the origin, hypothesis verification through the model, construction of an artifact or of a program, control of parts of the application, simulation of behaviour in certain situations, and substitution for a part of the application. Depending of the purpose we shall use different models.

Models are author-driven and addressee-oriented. They depend therefore on the culture, attitude, perceptions, education, viewpoints etc. of the stakeholders involved into the modelling process. Models are purposeful/situated/easily-modifiable/sharable/reusable/multi-disciplinary/multi-media chunks of knowledge about the application domain. They are both bigger and smaller than theories, i.e., bigger since they integrate ideas from different theories, since they use different representations, and since they are directed by their purpose; smaller since they are created for their purpose in a specific situation and since they are developed to be sharable and reusable. One of the most important quality characteristics of a model is that it should be easy to modify and to adapt.

4.3 Towards a Theory of Model Activities

Modelling activities are based on modelling acts. Modelling is a specific form and we may thus develop workflows of modelling activities. These workflows are based on work steps [33] such as ‘decompose’ or ‘extend’, abstraction and refinement acts, validation and verification, equivalences of concepts, transformation techniques, pragmatic solutions and last but not least the domain-specific solutions and languages given by the application and implementation domains.

The act of modelling is based on an activity that is characterised by the work products, the aspects under consideration (scope), the resources used in an activity, and the partners involved into the activity. Additionally we might extend this characterisation by activity goals and intentions (for what), time span (when), and restrictions (normal, exception and forbidden cases) or obligations for later activities. We may distinguish a number of activities and acts, e.g., understand, conceptualise, abstract, define, construct, refine, document and evaluate. Model activities should be governed by good practices which can be partially derived from modelling as an apprenticeship or technology. A theory of modelling activities has been developed in [36] and therefore not within the scope of this paper.
5 Modelling of Information Systems as Engineering

5.1 Models Serving Both as a Description of an Application (Domain and Problem) and as a Prescription for Construction (of Systems)

By taking a leaf out of D. Bjorner [1] book we divide information systems engineering into five main phases: (1) application domain description with properties that are of interest and that are of relevance, (2) requirements or objectives prescription for a model, (3) model development with a statement of properties that are obeyed by the model, (4) requirements of objectives prescription for the construction of an information system, and (5) information systems construction and coding with properties that are obeyed by the information system. Therefore, a model is used as a mediator between the application world and the systems world. The (application, model, system)-triple is reflected by the information system development triptych consisting of description of application world, prescription for construction and specification of systems.

Conceptualisation is an orthogonal phase that aims at a theoretical underpinning of models. It is used for semantification of models and for improvement of comprehensibility of models and explicit reasoning on elements used in models.

The application domain description is mapped to a model describing the application domain, its entities, functions, events, and behaviour. It is based on a formal, semiformal or natural language which allows to formulate a set of theorems or postulates or properties that are claimed to behold of the domain model. The information system itself is an artifact too. The model mediates between this final artifact and the application. Models describe the problem to be solved for the application and which are used as starting point for implementation. They are also used for documentation of the system, for migration and evolution processes, for optimisation of systems, for control of parts of systems, and for simulation of systems. Models must reflect the structure of a system, the functionality of a system, the support facilities of a system and the collaboration environment of a system.

Therefore we concentrate on one of the workflows: the prescription of systems imposed by the description of an application domain and of the problems under solution. This workflow is often considered to be one of the main workflows. We may also use other workflows. The construction workflow is however a typical example of an engineering workflow. Engineering is nowadays performed in a systematic and well-understood form.

5.2 The Construction Workflow Based on Information Systems Models

Modelling is based on an evolutionary process and thus consists of at least three subprocesses:

6 The difference between scientific exploration and engineering is characterised by [24] as follows: “Scientists look at things that are and ask 'why'; engineers dream of things that never were and ask 'why not'. Engineers use materials, whose properties they do not properly understand, to form them into shapes, whose geometries they cannot properly analyse, to resist forces they cannot properly assess, in such a way that the public at large has no reason to suspect the extent of their ignorance.” Modelling incorporates both engineering and science. It is thus considered to be an engineering science.
• *selection* including rigorous testing against the origin,
• *communication* for generation of a common understanding and a productive way of thinking within a community, and
• *accumulation* of results and integration of these results into future developments.

The construction workflow is one of the most prominent workflows in information systems modelling. Methodologies developed for software engineering can be directly applied to this workflow. They are however mainly oriented towards system construction. The systems description dimension is not as well explored. The combination of these two sub-workflows is shown in Figure 2. We need to include into this combination also the quality dimension. The body of knowledge of software engineering includes also a large set of quality characteristics. [10] develops an approach to systematic quality development. We integrate this systematic quality management.

We can also develop other workflows such as agile modelling, spiral modelling and incremental modelling workflows. We restrict our attention in the sequel to the workflow in Figure 2 due to the length of this paper. This workflow separates three different worlds: the world of applications, e.g., the application domain in dependence on the purpose; the world of models, e.g. conceptual models used for information systems development; the world of systems, e.g., information systems combined with presentation.

![Diagram of workflow](image-url)
Based on this separation we can distinguish three stages: the relevance, modelling and realisation stages. This workflow reflects the separation into objectives, the artifact and the properties within the language dimension.

5.3 Modelling by Generalising Engineering Approaches

The development of models results in many challenges. Modelling is essentially synthetic rather than analytic in substance. Identifying the real task of the modelling problem is probably the greatest challenge. Models can be based on building blocks. Another challenge is to find the right modelling method. Engineering targets at capacity of products to withstand service load both effectively and efficiently during their service life [24]. Efficiency also considers performance of the system. Engineering is also concerned with avoidance of technical, operational or unpredictable failures, i.e. to develop a system that deflect all service loads.

Engineering science for modelling is based on many different supporting sciences and technologies: industrial design, ergonomics, aesthetics, environments, life sciences, economics, mathematics, marketing, and manufacturing and forming processes. Engineering design includes five facets: design for effective function, design for manufacture, design for human users, socially responsible design, and economically responsible design.

Engineering distinguishes three dimensions: the stakeholder dimension, the procedure or process dimension, and the product dimension. It uses many techniques such as enformulation for structuring the purposes and objectives, problem decomposition together with component engineering, problem evolution, organising the engineering process, result evaluation, and result management. It also considers economic, social and environmental issues.

Therefore, it seems to be natural to use achievements of engineering for understanding modelling. This similarity is not only applicable to the description-prescription workflow but also for all the other workflows.

6 Modelling of Information Systems as a Science

6.1 Properties of Activities To Model

Activities to model form a process and can be characterised by a number of (ideal) properties:

**Monotonicity:** Activities are monotone if any change to be applied to one specification leads to a refinement. It thus reflects requirements in a better form.

**Incrementality:** Activities are iterative or incremental if any step applied to a specification is only based on new requirements or obligations and on the current specification.

**Finiteness:** Activities are finite if any quality criteria can be checked in finite time applying a finite number of checks.

**Application domain consistency:** Any specification developed corresponds to the requirements and the obligations of the application domain. The appropriateness can be validated in the application domain.
**Conservativeness:** Activities are conservative if any model revision that cannot be reflected already in the current specification is entirely based on changes in the requirements. Typical matured activities to model are at least conservative and application domain consistent. Any finite sequence of activities can be transformed into a process that is application domain consistent. The inversion is not valid but depends on quality criteria we apply additionally. If the modelling process is application domain consistent then it can be transformed in an incremental one if we can extract such area of change in which consistency must be enforced.

### 6.2 Towards Modelling Principles

A theory of conceptual modelling can be based on a system of guiding principles. We conclude that at least three guiding principles must be explored in detail:

- **Internal principles** are based on a set of ground entities and ground processes.
- **Bridge principles** explain the results of conceptual modelling in the context of their usage, for instance for explanation, verification/validation, and prognosis.
- **Engineering principles** provide a framework for mastering the modelling process, for reasoning on the quality of a model, and for termination of a modelling process within a certain level of disturbance tolerance (error, incompleteness, open issues to be settled later, evolution).

Information systems modelling principles are specialisations of design principles [3]. They are conventions for planning and building correct, fast or high performance, fault tolerant, and fit information systems. Conceptual modelling is based on architecture of a system into components, uses their interactions, and pictures their layout. Modelling is the process of producing models. It is thus adapted from engineering and may thus use the separation of activities into requirements, specification, development, and testing.

Depending on the purpose of the model several quality criteria may be preferred. For instance, construction models should fulfill criteria for good models such as correctness of models, refinement to highly effective systems, fault tolerance of systems, ubiquity of systems, and fitness of systems.

Modelling principles are not laws of nature, but rather conventions that have been developed by modellers to the most success when pursuing quality properties. Therefore, various sets of principles might be developed depending on the community. For instance, modelling based on extended ER models is based on compositionality, incrementality, structure-orientation, and conservativeness. Modelling principles for sets of models such as UML are far more difficult to develop and to maintain.

### 6.3 The Design Science Approach

MIS design science aims at the development of a general theory for models, model activities and modelling. We shall use the approach for a deeper insight into modelling. Models are called ‘design’ in [7].

The management information system community characterises the modelling process by seven guidelines [7]:

1. **Define the problem**: Clearly define the problem to be solved by the information system.
2. **Understand the problem**: Gather and analyse information about the problem and its context.
3. **Develop a solution**: Create a conceptual model that captures the essence of the problem and its solution.
4. **Implement the solution**: Translate the conceptual model into a design for the information system.
5. **Test the solution**: Verify the model and its design against the requirements.
6. **Prototyping**: Build a prototype to test the feasibility and usability of the solution.
7. **Maintenance**: Continuously update and maintain the system to adapt to changes in the problem context.
(1) model are purposeful IT artifacts created to address a problem;
(2) models are solutions to relevant and important problems;
(3) the utility, quality, and efficacy of models must be evaluated by quality assessment;
(4) modelling research must contribute to the state of the art;
(5) modelling research relies upon the application of rigorous methods;
(6) modelling is a search process and use termination conditions;
(7) models must be communicated both to technology-oriented as well as to management audiences.

We observe that guidelines (1), (2), and (7) are characterising the model. Guidelines (3), (6) characterise model activities. Guideline (3), (5) is related to modelling as a technology. Guideline (4) is a general statement that relates modelling to a science.

Main ingredients of modelling can be derived from these guidelines [4, 16]. Core components are purpose and scope (causa finalis), artifacts (causa materialis), the one-ness of form and function (causa formalis), artifacts mutability, testable propositions about the model, and theoretical underpinning. Additional requests are the potential implementation (causa efficiens) and utility for exposition and testing [4].

Design science separates three cycles [38]: the relevance (or description) cycle, the design (or modelling) cycle, and the rigor (or conceptualisation) cycle.

6.4 Reasoning Support for Modelling

Properties and objectives are used as a glue between the three sections or panels of the triptych. We distinguish between properties of the application, properties of the model as an artifact, and properties of the information system as a final artifact. Since we typically use different languages the (property-objectives)-pair is used as an hinge in the triptych. Design science separates three cycles [38] within modelling workflows. It distinguishes the relevance cycle as the iterative process that re-inspects the application and the model, the design cycle as the iterative model development process, and the rigor cycle that aims in grounding and adding concepts developed to the knowledge base. This separation of concern into requirements engineering, model development and conceptualization is the starting point for the development of a reasoning support for modelling.

This reasoning support includes an explicit consideration of the quality of the model, of the quality of the modelling process, and of the quality of supporting theories. We may combine the informal discussions with our approach and separate the modelling acts by the things that are under consideration. Figure 3 displays the different ways of working during a database systems development. We may distinguish the description-prescription-specification triptych and the engineering diptych. The first consists of three phases:

- **Description** of problems, phenomena and demands for system support in the application domain is based on the actual/goal analysis. It starts with the description of origins $O$ and targets at an understanding of properties $\Phi(O)$ that are of relevance. It may be used for derivation of objectives $\Psi(M)$ for model development.
- **Prescription** uses the objectives $\Psi(M)$ derived during description for model $M$ development and derivation of properties $\Phi(M)$ that should reflect properties $\Phi(O)$ of the origin.
Fig. 3. Reasoning processes and reasoning support for description followed by prescription

Specification is based on a model $M$ and its properties $\Phi(M)$. It may be used for derivation of objectives $\Psi(S)$ for system development.

The engineering diptych consists of specification of the system that should solve problems in the application domain and coding of the systems that augment the reality.

The relevance stage consists of $O$ and $\Phi(O)$. The modelling stage consists of $\Psi(M)$, $M$ and $\Phi(M)$. The realisation stage consists of $\Psi(S)$ and $Y$. It could consider $\Phi(S)$.

We observe that support for modelling results in a wide variety of reasoning. For instance, reasoning about properties of a model is also based on an explicit consideration of the notion of an analogy between the model and the application domain things, or the model and its reflection in theories and constructions. Reasoning on objectives of realisations includes detection of requirements a system must satisfy.

A general theory of reasoning must therefore cover many different aspects. We may structure these aspects by a pattern for specification of reasoning support for modelling acts or steps as follows:

- the modelling acts with its specifics; [36]
- the foundation for the modelling acts with the theory that is going to support this act, the technics that can be used for the start, completion and for the support of the modelling act, and the reasoning techniques that can be applied for each step;
- the partner involved with their obligations, permissions, and restrictions, with their roles and rights, and with their play;
- the aspects that are under consideration for the current modelling acts;
- the consumed and produced elements of the artifact that are under consideration during work;
• the resources that must be obtained, that can be used or that are going to be modified during a modelling act.

Consider, for instance, the reasoning that targets on realisation objectives. It includes specific facets such as

• to command, to require, to compel, and to make someone do something with supporting acts such as communicating, requesting, bespeaking, ordering, forbidding, prohibiting, interdicting, proscribing;
• to ask, to expect, to consider obligatory, to request and expect with specific supporting acts such as transmitting, communicating, calling for, demanding;
• to want, to need, to require, to have need of with supporting acts of wanting, needing, requiring;
• to necessitate, to ask, to postulate, to need, to take, to involve, to call for, to demand to require as useful, to just, or to proper.

The reasoning on operating, on relevant properties, on model objectives, on the model itself, on construction and assessment and guarantees can be characterised in a similar form.

The realisation stage may be replaced by other stage that support different purposes. We concentrated on prescription and construction of new systems. Another application is model refinement.

Design science aims at another kind of model refinement by adding more rigor after evaluation of a model. This refinement is essentially model evolution and model evaluation. Another refinement is the enhancement of models by concepts. This refinement is essentially a ‘semantification’ or model conceptualisation of the model.

Experimentation and justification of models is a third kind of adding rigor to (conceptual) models.

6.5 The Model of a Model

Models are often only considered with their intext, i.e., their structures und behaviour. Context is either neglected or taken for granted. We must however relate a model to the context dimension if we want to understand, to deploy or to modify the model.

Models follow typically some modelling schemata or pattern [5]. They are based on conceptions (concepts, theoretical statements (axioms, laws, theorems, definitions), models, theories, and tools). Conceptual processes include procedures, conceptual (knowledge) tools and associated norms and rules. Conceptions and conceptual processes are based on paradigms which are corroborated.

Models support interaction, understanding, sharing, and collaboration among people. They depend on existing knowledge, the actual (ontological) state of the reality, the condition of the person’s senses and state of mind, and the state of employed instruments. Therefore, models depend on the background concepts that are accepted in a community.

We can summarise the considerations so far and develop a general model frame, i.e., a model of the model. It consists of four main components
Founding concepts: A model is based on paradigms, background theories, assumptions and guiding principles. It is composed of base conceptions/concepts with a certain scope, expressions, concept space organisation, and some quantification/measurement. Language-based models use a namespace or ontology as a carrier. This namespace is based on definitions made, i.e., the cargo in the sense of [15].

Structure and behaviour: A model is often build incrementally. Models can be multi-faceted (with a specific topology/geometry, with states, with interactions, with causal associations) or monolithic.

Application domain context: A model corresponds to a part of the reality, i.e., the application domain. The domain forms the empirical scope of the model. General or application-specific correspondence rules guide the association between the origin and the model. Each application domain is based on general laws one might have to consider for the model as well.

Meta-model: Models are developed within a theory and have a status within it. These theories provide the content of paradigms. Concepts are the most elementary building blocks. The construction process of a model is guided by the laws applicable to such theory. We may use basic models, emergent models, and subsidiary models.

Reusing the theories of concepts, content and topics [34], we shape the general concept frame. A concept is given by the scope, by at least one expression, by its association to other concepts and its media type [27] for the content. The application domain and potential functions constitute the scope of a concept. A concept can be defined by one or more partially synonymous expressions in a definition frame [36]. The concept space must follow some internal organisation. Concepts are interdependent and associated with each other. A concept must be underpinned and quantified by some data which use a certain format. We assume that the formatting can be given by a media type.

6.6 Model Fitness

[5] introduces model viability. We extend this approach and consider fitness of a model. Fitness (or superior quality) of a model is given by

(a) usability of the model for its purpose, i.e., for resolving the questions, e.g., validity of the model;
(b) potential of the model for the purpose, i.e., for the goals that are satisfied by the model, e.g., reliability and degree of precision of the the model;
(c) efficiency of the model for the function of the model within the application, i.e., the practice [39] of deployment of the model;
(d) generality of the model beside its direct intention of construction of the model, i.e., for applying the model to other goals or purposes, within another function or with some modification or extension, e.g., the extend of coverage in the real world.

These four criteria form main quality characterisations of a model. Viability is defined through validity, reliability for the model purpose and function, extent of coverage in dependence on context such as space and time, and efficiency of the model. Viability
thus can be used to evaluate how well the model represents the reality for a given scope and how suitable or instrumental is the model for its purpose and function.

The potential of a model is defined by its strengths, weaknesses, opportunities and threats. The potential can be assessed within a SWOT analysis. A model must be empirically corroborated in dependence on the objective. The abstraction property [31] determines the degree of corroboration. A model must be consistent with the context and the background and coherent in its construction. Models are parsimonious reductions of their origins. Due to this reduction models must be revisable for changes in reality. At the same time models must be relatively stable and robust against minor changes.

6.7 Observations for Information Systems Model Engineering

Engineering of conceptual models inherits both facets of didactically ruled learning [30] and of engineering [24]. The following characteristics of engineering sciences are observed also for conceptual modelling:

(α) The origin of a model is partly a product of creativity. Systems developed in our field are a product of developers and thus dependent on these stakeholders. They must be understood, well-explained and used with a purpose.

(β) The origin of a model is a complex systems. The attention focuses both on the creation of complex artifacts and on conceptualisations of the application world. They are typically modularly constructed. Modularisation is only one of the underlying design principles. Conceptual modelling targets at useful concepts. It goes through a series of iterative design cycles in dependence on its purpose.

(γ) Models satisfy the purpose, are sharable, useful and reusable. Models are not developed just as an intermediate result of the implementation process and for satisfaction of purpose. They are shared within a community and are reusable in other situations. Moreover, models support a better understanding of the origins.

(δ) The origins are continuously changing and thus the models too. The application domain is continuously evolving. Models must correspondingly evolve too. Significant changes tend to be applied to the starting model so that the original concepts become unrecognizable after model evolution. Models are also used for changing the application world. This change must again be reflected within the model.

(ε) Origins being modelled are influenced by social constraints and affordances. Models are influenced as much by purposes as by physical and economical aspects of the contexts in which they are used. These influences are changing and evolving as well. Therefore, models are going to be used in ways their stakeholders did not imagine. Models are influenced as much by socially generated capital, constraints, and affordances as by the capabilities of stakeholders who created them.

(ζ) No single “grand theory” is likely to provide realistic solutions to realistic complex application problems. In realistic modelling situations that involve information systems, there almost never exist unlimited resources. Relevant stakeholders have typically conflicting goals. Therefore, ways of working displayed in Figure 3 usually need to integrate approaches drawn from different disciplines.
(η) Development of a model usually involves a series of iterative modelling cycles. Artifacts that serve as models are developed through a series of model activities and are iteratively tested and revised in dependence on the purpose.

Consequences for model engineering: The modelling process itself also changes the application domain and the understanding of the origin. Therefore, modelling is not reducible to condition-action rules. Modelling is a matter of engineering. Experienced modellers not only right develop a model but they also develop the right model - by developing models at the right time, with the right background and context, and for the right purpose. Model engineering is therefore based on advanced skills of handcrafting, i.e., making substitutions and adaptations depending on purpose and application situation, understanding which compositions perform best, continuously adapt the result of the process, and understand difficult-to-control things in their handcraft environment. The same situation is valid for information systems development if performance of the system becomes crucial and heavily depends on the DBMS.

7 Techniques for Modelling of Information Systems

7.1 Separation and Decomposition of the Workflows

Modelling integrates the classical problem solving four-phase cycle [22]:

1. Developing an understanding of the task: The task is analysed within its context and is compared with the goal for completion of the task. The initial situation is characterised. in the case of modelling processes the understanding is based on objectives derived from the purpose or from previous steps.
2. Development of a plan for the solution of a task: The instruments and tools for the solution of the task are reconsidered. The plan consists of heuristic forward and backtracking steps, steps for problem restructuring and for quality control.
3. Application of the plan for the development of the solution: The plan is consecutively applied for the generation of the solution. If certain steps are considered to be inappropriate then the plan is revised as well.
4. Development of an understanding of the solution: The result is evaluated based on criteria that either follow from the purpose or from the problem. Properties of the solution are derived.

This approach is based on four components:

The state space consists of the collection of all those states that are reachable from the initial state. Some of the states are considered to be desirable, i.e. are goal states. States can be modelled through languages such as ER. State may have properties such as suitability for certain purposes.

The actions allow to move from one state to another state under certain conditions. We may assume that the effect of the actions is observable to a certain extent by the user. User may use several actions in parallel. Actions may be blocked or enabled depending on conditions. Actions may be used at some cost.
The goal test determines whether a given state or state set satisfies the goals. The goal test may be defined through a set of states or through properties. The goal test may also allow to state which quality has the state set for the problem solution.

The controller evaluates the actions undertaken by the stakeholder. Some actions may be preferred over other, e.g. have less costs, or are optimal according to some optimality criterion. Controllers can be based on evaluators of the paths from the initial state to the current state.

Creation steps are the most complex steps in modelling. They typically consist of an orientation or review substep, of a development step performed in teams, and of a finalisation substep. Creation steps are composed of a number of substeps that can be classified into:

- Review of the state-of-affairs: The state of the development is reviewed, evaluated, and analysed. Obligations are derived. Open development tasks can be closed, rephrased or prioritised.
- Study of documents and resources: Available documents and resources are checked whether they are available, adequate and relevant for the current step, and form a basis for the successful completion of the step.
- Discussions and elicitation with other partners: Discussions may be informal, interview-based, or systematic. The result of such discussions is measured by some quality criteria such as trust or confidence. They are spread among the partners with some intention such as asking for revision, confirmation, or extension of the discussion.
- Recording and documentation of concepts: The result of the step is usually recorded in one work product or consistently recorded in number of work products.
- Classification of concepts, requirements, results: Each result developed is briefly examined individually and in dependence of other results from which it depends and to which it has an impact.
- Review of the development process: Once the result to be achieved is going to be recorded the work product is examined whether it has the necessary and sufficient quality, whether it must be revised, updated or rejected, whether there are conflicts, inconsistencies or incompleteness or whether more may be needed. If the evaluation results in requiring additional steps or substeps then the step or the substep is going to be extended by them.

7.2 Maieutics for Mastering Iterations

Modelling of information systems is not only aiming at achieving a nominal system but aims too at satisfaction of real interests of all stakeholders involved into modelling. It must consider all relevant aspects of an application and thus results in co-design of structuring, functionality, and supporting systems such as view and interaction support [33]. Stakeholders (or users) iteratively obtain a deeper insight and understanding about the necessities and conditions of the problem and the strengths, weaknesses, opportunities and threats of the solution depending on the purpose of the modelling within a modelling process. Therefore, modelling integrates ideas developed for maieutics [14, 17].

The maieutics frame [19] is essentially a specific form of a dialogue. In conceptual modelling, it consists (1) of an open-ended process, (2) of the elaboration of ideas that
are grounded in references to the application domain, to the users, prior knowledge and experience, and to the languages as carriers, and (3) of the discussion (in form of conceptualisation, interpretation, explanation, diverging ideas, and new understandings) that is inductive and exploratory rather than deductive and conclusive.

Modelling requires to utilise the knowledge in dependence on the purpose of the model. Answers found during modelling may not be evident in the material on hand; modellers may have to delve into subtleties or ambiguities they had not thought of. Information systems modelling is based on elaboration and conceptualisation of model elements. The inductive and exploratory discussion facilitates the development of argumentation by fostering the (re)consideration of alternatives and versions.

Conceptual modelling is based on references to the application domain, connections across the model, elaboration based on prior knowledge and/or experience, interpretations, explanations and conceptualisations, diverging ideas, and new understandings. Therefore the modelling process is highly iterative and revising/remastering decisions that have already been made.

7.3 Management and Support for Sub-workflows

For the modelling stage we derive the following general approach based on problem solving cycles:

<table>
<thead>
<tr>
<th>Initiation</th>
<th>Differentiation and Understanding</th>
<th>Evaluation and Selection by Relevancy</th>
<th>Model</th>
<th>Justification and Consensus</th>
<th>Experimentation and Field Exploration</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>$O$</td>
<td>$\Phi(O)$</td>
<td>$\Psi(M)$</td>
<td>$M$</td>
<td>$\Phi(M)$</td>
<td>$\Psi(Y)$</td>
<td>$Y$</td>
</tr>
<tr>
<td>Origin</td>
<td>Origin properties</td>
<td>Modelling Objectives</td>
<td>Artifact</td>
<td>Properties</td>
<td>Objectives</td>
<td>Implementation</td>
</tr>
</tbody>
</table>

We therefore arrive at a modelling process in Figure 4 that refines the general workflow in Figure 2. We may zoom-in into these sub-workflows. For instance, one of the most interesting step is step (3) in the modelling activities. This step consists of a number of substeps. The conceptualisation stage is orthogonal within the database design framework in Figure 5.

Conceptualisation is based on the notion of concepts introduced in [12, 37]. Design science [4, 7, 16, 38] uses the rigor cycle as one of its three cycles aiming at model development. The rigor cycle has not yet been defined. We also arrive at a sub-workflow for conceptualisation in Figure 6.

8 Duties and the Task Spectrum in Conceptual Modelling

8.1 CMM and SPICE for Conceptual Modelling

A software process is considered to be the set of activities, methods, and practices used in the production and evolution of software [8] and the associated products [21]. For improving of a software process there are four main approaches: modelling, assessment, measurement, and technology adoption [25]. The approaches supplement each other, but one of them is usually in a dominating position. CMMI and SPICE (Software Process Improvement and Capability dEtermination) [9] are the two most widely
Fig. 4. The sub-workflows for construction modelling processes

Fig. 5. Sub-steps of the modelling and concept derivation step in the modelling sub-workflow

Fig. 6. The sub-workflow for conceptualisation steps within the modelling step
used software assessment models in software process improvement work today. The capability dimension consists of six capability levels: incomplete, performed, managed, established, predictable and optimizing.

Information system development is a specific software development process. Therefore, the SPICE characterisations are applicable as well. We may therefore distinguish different levels of the IS development capability:

1. **Performed and executed**: The goals of the application domain are satisfied. The information system development process is set out.
2. **Managed and defined**: Additionally, the application domain and scope are imaged by a model that allows to derive components of the system by means of model elements.
3. **Established and controlled**: The model is well-documented and allows to understand its design decisions. The model is used as a background and groundwork for the system.
4. **Understood, predictable and performed with sense**: The elements of the model are based on concepts that describe their semantics and meaning. The impact of languages as a model carrier, the assumptions made during design, the paradigms used during and the scope of the model are given in an explicit form.
5. **Optimised**: The model is developed with a number of alternatives. There are quantitative methods that support reasoning on quality of the model. Model alternatives can be given in a form that is the most adequate for the auditory. They can be used for deriving the best realisation.

### 8.2 The Duty Portfolio of Modelling

Following [6] we distinguish four main duties in conceptual modelling:

1. **Description**: The application domain is described in a way that allows to comprehend the actual state, the necessities for system development and deployment, and the specifics and phenomena of the application.
2. **Explanation**: The understanding of reality, of the processes and data in the application world and of the context of the application supports the creation of systems that effectively and efficiently support users. This understanding can be based on explication of concepts behind the application. It can also be based on behavioural pattern, on general laws and regulations and on user profiles and stories [26, 27].
3. **Creation**: The system creation includes coding of the system, embedding the system into a systems context, developing supporting means for users, and supporting a new behaviour of users of a system. It uses the demands stated for the application, the analysis of the current state, and the requests for change by the system. Creation includes elements of SWOT analysis (strengths, weaknesses, opportunities, threats) and evaluation of the quality of the system.
4. **Prognosis**: The behaviour of the augmented system, the opportunities of changes and evolution and the restrictions of the augmented reality are predicted. The user expectations and the reality of system exploitation are compared on the basis of main storyboards observed for the applications.
We may now combine these approaches into a process survey in Figure 7. The relevance cycle is based on observation of the state of affairs, scoping of the demands for system development, and describing a view of the application domain. These cycles form the y-dimension. We also use the x-dimension for explicit display of the changes imposed to the reality. Typically, information systems augment the reality. Figure 7 combines the approaches of design science (research) [7, 16] with those based on main duties for system development [6] and those typically used for conceptual modelling [33].

The design or modelling cycle uses the scoped application domain description for the development of a model. The rigor cycle adds semantics, meaning and context to the model. The description of the scoped application domain may directly be used for system development. For instance, agile development is typically following this direct approach. The model may also be directly used for system development. The advantage of such approach is that all relevant elements are supported by a model and that the model may be used for understanding the system. The system is therefore defined. We may also use the model for development of a behaviour description, guidelines (e.g., for system deployment) and documentation. In this case modelling is established.

Furthermore, we might background the model by concepts. In this case, users of the model may perform system construction with a sense of groundwork behind the model and the description of the application domain. Models may also be a part of a knowledge base. In this case we integrate, generalise and found the model through concepts in the knowledge base.

The relevance, design and rigor cycles are based on comprehension of the application domain, perception of the relevant elements and knowledge or understanding development for those elements. During system development models are used as a mediating artifact. They describe and image the problems, phenomena and demand form one side and serve as a prescription for systems development from the other side. Models may also serve as a background and foundation of the system if they are integrated with concepts.

9 Conclusion

Models are artifacts that can be specified within a \( (W^4 + W^{17} H) \)-frame based on the classical rhetorical frame introduced by Hermagoras of Temnos\(^7\).

1. They are primarily characterised by \( W^4 \): wherefore (purpose), whereof (origin), wherewith (carrier, e.g., language), and worthiness ((surplus) value).
2. Secondary characterisation \( W^{17} H \) is given by:
   - user or stakeholder characteristics: by whom, to whom, whichever;
   - characteristics imposed by the application domain: wherein, where, for what, wherefrom, whence, what;

\(^7\) Quis, quid, quando, ubi, cur, quem ad modum, quibus adminiculis (\( W^7 \): Who, what, when, where, why, in what way, by what means). The Zachman frame uses a simplification of this frame.
• purpose characteristics characterising the solution: how, why, whereto, when, for which reason; and
• additional context characteristics: wheretowards, whereabouts, whither, when.

Modelling combines at the same time and systematically different aspects: culture, art, systematics and technology of model (re)development and model application. It uses modelling activities and techniques.

Conceptual modelling is biased through a pragmatical culture. It uses languages as a sophisticated medium of expression. It defines its specific arts and sciences. It reflects thoughts, i.e. perception, interpretation and understanding of people involved. It
is implicitly based on value systems transmitted through communities of practice based on some commonsense and consensus. Conceptual modelling is also at the same time a social activity, i.e. a shared pursuit within a community, demonstrated in a variety of textbooks, publications and conferences. Conceptual models are used for social aspects, i.e. include the give-and-take of socialisation, negotiation, protocol, and conventions within the community of their users. These aspects of models and of modelling activities collectively redefine conceptual modelling as a culture.

A general theory of models and modelling contains also other considerations: models as constructs that use signs, practices of model deployment, conditions for model functioning, cognitive and epistemic functions, normative functions, status and role of modelling, boundaries and reach of models, reflections on model deployment, etc. In Computer Science more specific questions must be taken into account such as the following ones: elements of model purposes, functioning of models, genesis, model capacity, added value of models, analysis of the general and epistemic role of a model, dependence of models in the context of other artifacts, functional reach of a model, etc. Models are elements of the Computer Science culture. As such we need to consider problems of model quality characteristics, of difference development and discovery, of integration into knowledge, of capability for system performance and prognosis, of the social impact of models within communities, of integration of intuition and vision, of parallelization and coexistence of models, of importance for feedback, of dependence of models on the carrier (language), of approximation and reduction, of abstraction, etc. These questions are problems for future research.

References


Syntax, Semantics and Pragmatics of Conceptual Modelling

Bernhard Thalheim
Computer Science Institute, Christian-Albrechts-University Kiel,
Olshausenstrasse 40, 24098 Kiel, Germany
thalheim@is.informatik.uni-kiel.de

Abstract
Models, modelling languages, modelling frameworks and their background have dominated conceptual modelling research and information systems engineering for last four decades. Conceptual models are mediators between the application world and the implementation or system world. Currently conceptual modelling is rather a craft and at the best an art. We target on a science and culture of conceptual modelling.

Models are governed by their purpose. They are used by a community of practice and have a function within application cases. Language-based models use a language as a carrier. Therefore, semiotics of models must be systematically developed. This paper thus concentrates on the linguistic foundation of conceptual modelling.

1 Introductory Notions for Conceptual Modelling

Conceptual modelling is a widely applied practice in Computer Science and has led to a large body of knowledge on constructs that might be used for modelling and on methods that might be useful for modelling. It is commonly accepted that database application development is based on conceptual modelling. It is however surprising that only very few publications have been published on a theory of conceptual modelling. We continue our research [2, 3, 15, 16, 17, 18] and aim in a theory of linguistic foundations for modelling within this paper.

1.1 Goals, Purposes and Deployment Functions of Conceptual Models

Purpose is often defined via intention and mixed with function. Goal (or intention or target or aim) is a ternary relation between a current state, envisioned states, and people (community of practice). Typical - sometimes rather abstract - intentions
are perception support, explanation and demonstration, preparation to an activity, optimisation, hypothesis verification, construction, control, and substitution.

The *purpose* is a binary relation between intentions and means or instruments for realisation of the intention. The main mean we use is the language. Semiotics is widely intentionally used for modelling; however without paying attention to it.

The *deployment function* of a model relates the model purpose to a practice or application cases or application ‘game’ similar to Wittgenstein’s language game (we call it better *deployment case* and is characterised by answering the classical W-questions: how, when, for which/what or why, at what/which (business use case), etc. We add to purpose: application, conventions, custom, exertion, habit, handling, deployment, service, usage, use, and way of using. The model has a role and plays its behaviour within this application game.

![Figure 1: Distinction of Intention, Purpose, and Deployment Function of a Model](image)

### 1.2 Abstraction and Conceptual Modelling

Abstraction is one of the most overloaded conceptions in Computer Science and at the same time one of the most under-specified. Abstraction means development of general concepts by abstracting common properties of specific concepts. Using the approach in [14] we develop three dimensions of abstraction:

**Structural abstraction** is used for highlighting essential, necessary, and general structural elements of the origin. Structural abstraction has three main constituents: *combining structural abstraction* (often called) *classifying structural abstraction* (often called meta(-meta(-meta)) abstraction) combines things of interest into collections that contain these things as elements; *generalising structural abstraction* (often called pattern or templates).

**Context abstraction** “factors out” repeating, shared or local patterns of things and functionality from individual things. Context abstraction assumes that the surroundings of a things under consideration are commonly assumed by a community of practice or within a culture and focuses on the concept, turning away attention from its surroundings such as the environment and setting. Models use ambiguities, ellipses, metaphors and commonly assumed conceptions.
**Behavioural abstraction** is used for concentrating of essential and general behavioural elements. We may distinguish between combining, classifying, and generalising behavioural abstraction. Aspect separation, encapsulation, and modularisation are specific techniques.

The opposite of abstracting is detailing. **Refinement** is a specific kind of faithful detailing. Refinement uses the principle of modularisation and information hiding. Developers typically use conceptual models or languages for representing and conceptualising abstractions. Classical forms of abstraction are generalisation, isolation, and idealisation.

We thus may concentrate on three main tasks for abstraction within a community of practice:

- Choose the right scope within the application area in dependence on goals.
- Choose the right focus at the right level and in the right granularity.
- Choose the right observation with the right behaviour and right properties.

## 2 The Notion of the Model

It is noted (e.g. [1]) (it can be observed indeed also in almost all textbooks on Computer Science) that there is no commonly accepted adequate definition of the concept of the model. This claim is considered as one of the big lacuna of the science and art of modelling.

A **model** is simply a material or virtual artifact (1) which is called model within a community of practice (2) based on a judgement (3) [7] of appropriateness for representation of other artifacts (things in reality, systems, ...) and serving a purpose (4) within this community. We distill thus criteria for artifacts to become a model. We can use on two approaches: abstract properties or we criteria for artifacts.

### 2.1 Stachowiak, Aristoteles, Galilei and Mahr Properties of Models

Models are often defined through abstract properties they must satisfy [15, 18].

(1) **Mapping** property: Each model has an origin and is based on a mapping from the origin to the artifact.
(2) **Truncation** property: The model lacks some of the ascriptions made to the original and thus functions as an Aristolean model by abstraction of irrelevant.
(3) **Pragmatic** property: The model use is only justified for particular model users, tools of investigation, and period of time.
(4) **Amplification** property: Models use specific extensions which are not observed for the original,
(5) *Distortion* property: Models are developed for improving the physical world or for inclusion of visions of better reality, e.g. for construction via transformation or in Galilean models.

(6) *Idealisation* property: Modelling abstracts from reality by scoping the model to the ideal state of affairs.

(7) *Carrier* property: Models use languages and are thus restricted by the expressive power of these languages.

(8) *Added value* property: Models provide a value or benefit based on their utility, capability and quality characteristics.

(9) *Purpose* property: Models and conceptual models are governed by the purpose. The model preserves the purpose.

The first three properties have been introduced by Stachowiaks [12]. The fourth and fifth property have been introduced by Steinmüller [13]. The seventh property is discussed by Mahr [10]. The sixth, eight and nine properties [18] are however equally if not the most important ones.

### 2.2 Criteria for Appropriateness of an Artifact to Become a Model

The separation into goal, purpose and deployment function for models provides three main appropriateness criteria:

(1) The adequacy of a model defines its potential for the goals. Adequacy is given by the similarity of the model with its origin in dependence on its goal, the regularity for the application (within a well-founded system that uses rules for derivation of conclusions), the fruitfulness (or capacity) for goals, and the simplicity of the model through the reduction to the essential and relevant properties in dependence on the goal.

(2) A model is fit for its purpose if it is usable for the purpose, suitable within the given context and for the prescribed purposes, robust against small changes in the parameters, accurate to the level of precision that is necessary for the purpose, and compliant with the funding concepts, application context and meta-model.

The model must be testable and, if false, it can be disconfirmed by a finite set of observations (finitely testable) and by any of superset of these observation (irreversibly testable).

(3) The usefulness for deploying is given by effectiveness for complete and accurate satisfaction of the goal, understandability for purposeful deployment of the model by users, learnability of the model within the deployment stories, reliability and a high degree of precision of the the model, and efficiency of the model for the function of the model within the application. testability

Additional criteria are generality of the model beside its direct goals and intentions and the extend of coverage in the real world for other goals.
2.3 The Hidden Background of Models

The structure and function of a model are based on a correspondence relation between the reality or the augmented and the model. The model can be constructed incrementally. It represents a number of facets of the origin (topology/geometry, state, interaction, causal). The model pragmatism is however hidden. It consists of at least three background dimensions:

Founding concepts: A model uses the cultural background within the application area and within a community of practice. Base conceptions (scope, expressions, concept space organisation, quantification/measurement), a namespaceuticalontology/carrier, a number of definitions (state, intrinsic, object, interaction descriptors and depictors), and a language as cargo [10] characterise these founding concepts.

Application context: The application domain binds the model to some common understanding that is not explicitly defined in the model. Each model has an empirical scope of the model, has specific application-domain driven correspondences, and must satisfy a number of laws and regulations.

Meta-model: Each model has a basement, is restricted by paradigms and theories, has a status in the application; context, displays elements on certain abstraction level and granularity, and uses a scale. It is also prone for paradigmatic evolution within the epistemological profile of community of practice.

These dimensions are partially known in didactics (of modelling) [5]. The three background dimensions drive however the model deployment and development.

3 Language-Backed Modelling

3.1 Language Selection Matters

Languages may however also restrict modelling. This restriction may either be compensated by over-development of language components or by multi-models. The relational database modelling language uses integrity constraint as compensation component for the inadequate expressibility of the language. The Sapir-Whorf hypothesis [19] results in the following principle:

Principle of linguistic relativity: Actors skilled in a language may not have a (deep) understanding of some concepts of other languages. This restriction leads to problematic or inadequate models or limits the representation of things and is not well understood.
The principle of linguistic relativity is not well understood. In [15] we demonstrated via a crossroad example that Petri nets are often not the right tool for representation of behaviour. A similar observation on UML is made by Krogstie [9].

3.2 The Cognitive Insufficiency of the Entity-Relationship Modelling Language

Lakoff introduces six basic schemata of cognitive semantics without stating that this list of schemata is complete.

- The container schema define the distinction between in and out. They have an interior, a boundary and an exterior.
- The part-whole schema define an internal structuring and uses whole, part and configuration as construction units.
- The link schema connects thing of interest. It uses various kinds of links for associating or un-associating things.
- The center-periphery schema is based on some notion of a center. Peripheral elements are not as important than those in the center.
- The source-path-goal schema uses source (or starting point), destination, path, and direction. It allows also to discuss main and side tracks.
- Typical ordering schemata are the up-down, front-back and the linear ordering schema. They use spatial and temporal associations.

We call a modelling language cognition-complete if these six schemata can be represented.

The classical ER modelling language suffers from a number of restrictions. It uses the container and the link schemata. It allows to mimic the part-whole schema via special links (called IsA). This work-around is however badly misunderstood. In order to become cognition-complete integrity constraints must be used. Their cognitive complexity is however beyond surveyability of humans. A typical flaw of the classical ER model is the use of monster types that integrate stable - almost not changing - properties and transient - often changing - properties. Objects are then taken as a whole. Unary relationship types easily resolve this problem if higher-order types are permitted.

Extended ER modelling languages are however also not cognitive complete. The center-periphery schema can only be emulated. The source-path-goal schema can be represented by higher-order relationship types. The part-whole schema is supported by the specialisation via unary relationship types and by generalisation via cluster types. Ordering schemata can be defined using the order types and bulk types.
4 Syntax of Conceptual Models: Structuring and ‘Functioning’

Syntax of models is build on deictic context-based rules [20] for construction of complex expressions using a domain-dependent vocabulary and governed by a set of meta-rules for construction (styles, pattern, abstraction).

4.1 Morphology of Conceptual Models and the Form of Elements

Morphology is the science of word form structure. The part of syntax is completely neglected in conceptual modelling. It is however equally important. Elements of a modelling language can be classified according to their categories and roles within a model and according to their specific expression within a model. Expression might similarly ruled by inflection, deviation, and composition. Therefore, techniques like lemmatisation (reduction of words to their base form) and characterisation by the (morpho-syntactic) role within a model.

Based on [11] we distinguish five morphological features: full or partial specification, layering within a model, integrity constraints, cyclic or acyclic structuring, complete set of schemata for cognitive semantics, open or closed context, and kind of data types.

Syntax for models is context-dependent. Constructs are bound by an implicit construction semantics [14] that is an integral component of any language. Models are governed by syntactic rules or explicit and implicit social norms. They are constructed with implicit styles and architectures.

4.2 The Lexicography and the Namespace of Models

Lexicography has developed a number of principles for coding and structuring lexical elements based on the lexicon on the application domain. Most ontological research in Computer Science does not got beyond lexicography and uses a topical annotation of model elements while hoping that every stakeholder has the same interpretation for words such as ‘name’, ‘description’, ‘identifier’ etc. If we consider however more complex entries such as ‘address’ then we detect that such kind of annotation does not work even for one language. The situation becomes far worse if we consider different languages, cultures, or application domains. Then the nightmare of “integration” becomes a challenge.

Models typically use a general and an application-dependent namespace. Moreover, the model is a product of a community of practice with its needs, its common-speak, its specific functions of words, its specific phrases and abbreviations, and its specific vocabulary.
5 Semantics of Conceptual Models

5.1 Kinds of Semantics

Semantics is the study of meaning, i.e. how meaning is constructed, interpreted, clarified, obscured, illustrated, simplified, negotiated, contradicted and paraphrased. It has been treated differently in the scientific community, e.g., in the area of knowledge bases and by database users.

- The scientific community prefers the treatment of ‘always valid’ semantics based on the mathematical logic. A constraint is valid if this is the case in any correct database.
- Database modellers often use a ‘strong’ semantics for several classes of constraints. Cardinality constraints are based on the requirement that databases exist for both cases, for the minimal and for the maximal case.
- Database mining is based on a ‘may be valid’ semantics. A constraint is considered to be a candidate for a valid formula.
- Users usually use a weak ‘in most cases valid’ semantics. They consider a constraint to be valid if this is the usual case.
- Different groups of users use an ‘epistemic’ semantics. For each of the group its set of constraints is valid in their data. Different sets of constraints can even contradict.

Semantics is currently one of the most overused notions in modern computer science literature. Its understanding spans from synonyms for structuring or synonyms for structuring on the basis of words to precise defined semantics. This partial misuse results in a mismatch of languages, in neglecting formal foundations, and in brute-force definitions of the meaning of syntactic constructions.

Semantics of models uses also commonsense, intended and acceptable meanings, various kinds of quantifications, (logical) entailment, deduction, induction, and abduction.

5.2 The Lexicology and the Namespace of Models

Ontologies are becoming very popular in Computer Science research. Philosophy developed now a rather restrained usage of ontologies. Lexicology [4] - as a part of philology - or semasiology is based on semantic relations in the vocabulary of a language. Lexicology of models studies elements of models and their meaning, relations between these elements, sub-models and the namespace in the application
domain. Classical linguistic relations such as homonym, antonym, paronym, synonym, polysemy, hyponym, etc. are used for stereotyped semantics in the namespace.

Models combine two different kinds of meaning in the namespace: referential meaning establishes an interdependence between elements and the origin (‘what’); functional meaning is based on the function of an element in the model (‘how’). The referential meaning is well investigated and uses the triangle between element, concept and referent. The functional meaning relates elements in a model to the model context, to the application context, and to the function of this element within the model. It thus complements the referential meaning. Additionally model lexiconology use the intext (within the model), the general, the part-of-model, and the differential (homonym-separating) meaning. Further, we need to handle the change of meaning for legacy models.

6 Pragmatics of Conceptual Models

6.1 General Pragmatics of Modelling in a Community of Practice

Pragmatics for modelling is the study how languages are used for intended deployment functions in dependence on the purposes and goals within a community of practice. Functions, purposes and goals are ruling the structure and function of the model. We distinguish the descriptive-explanatory and persuasive-normative functions of a model. Models are used for (1) acting (2) within a community, especially the modeller and have (3) different truth or more generally quality [8]. We may distinguish between far-side and near-side pragmatics separating the ‘why’ from the ‘what’ side of a model. General pragmatics allows to describe the overall intentions within the community and strategies for intention discovery.

We may distinguish between the development and deployment modes. The first mode starts with abstraction and mapping and then turns to the representation. The second mode inverses the first mode. Based on this distinction we infer a number of basic principles of modelling pragmatics similar to Hauss [6]: model surface compositionality (methodological principle), model presentation order’s strict linearity relative to space (empirical principle), model interpretation and production analysed as cognitive processes (ontological principle), reference modelled in terms of matching an model’s meaning with context (functional principle).

Models must be methodologically valid, support subjective deductive (paraductive) inference with an open world interpretation, allow context-dependent reasoning (implicature), provide means for collaborative interaction, weaken connectives and quantifications, and integrate deductive, inductive, abductive and paraductive reasoning.
6.2 Visualisation or the ‘Phonetics’ of Conceptual Models

Phonology is the science of language sounds. *Phonetics* investigates the articulatory, acoustic, and auditive process of speech. It is not traditionally considered not considered to be a part of a grammar. But it is equally important for practical language deployment. Phonology of models is concerned with the ways in which intentions can be conveyed using conventional and non-conventional resources. The modeller uses reference for transferring the specific point of view that has been used for modelling.

Visualisation is the ‘phonetics’ in modelling. It is based on three principles:

**Principles of visual communication** are based on three constituents: Vision, cognition, and processing and memorizing characteristics. We may use specific visual features such as contrast, visual analogies, presentation dramaturgy, reading direction, visual closeness, symmetric presentation and space and movement.

**Principles of visual cognition** refer to ordering, effect delivery, and visualisation. We base those on model organisation, model economy, skills of users, and standards.

**Principles of visual design** are based on optical vicinity, similarity, closeness, symmetry, conciseness, reading direction. These principles help to organise the model in a way that correspond to human perception.

7 Summary

Models are artifacts that can be specified within a 
(W^4+W^{17}H)-frame that is based on the classical rhetorical frame introduced by Hermagoras of Temnos\textsuperscript{1}. Models are primarily characterised by W^4: wherefore (purpose), whereof (origin), where-with (carrier, e.g., language), and worthiness ((surplus) value). Secondary characterisation W^{17}H is given by:

- user or stakeholder characteristics: by whom, to whom, whichever;
- characteristics imposed by the application domain: wherein, where, for what, wherefrom, whence, what;

\textsuperscript{1}The rhetor Hermagoras of Temnos, as quoted in pseudo-Augustine’s De Rhetorica defined seven “circumstances” as the loci of an issue: Quis, quid, quando, ubi, cur, quem ad modum, quibus adminiculis(W^7: Who, what, when, where, why, in what way, by what means). See also Cicero, Thomas Aquinas, and Quintilian’s loci argumentorum as a frame without questioning. The Zachman frame uses an over-simplification of this frame.
• purpose characteristics characterising the solution: how, why, where, when, for which reason; and

• additional context characteristics: where, whereabout, whither, when.

Modelling is the art, the systematics and the technology of model (re)development and model application. It uses model activities and techniques. This paper is going to be extended by more specific aspects of the modelling art in the context of semiotics and linguistics.

References


The Conceptual Model \equiv
An Adequate and Dependable Artifact
Enhanced by Concepts

Bernhard THALHEIM

Christian-Albrechts University Kiel, Computer Science Institute, 24098 Kiel, Germany

Abstract. Conceptional modelling is one of the central activities in Computer Science. Conceptual models are mainly used as intermediate artifact for system construction. The notion of the conceptual model is at present rather informal. Conceptional modelling is performed by a modeller who directs the process based on his/her experience, education, understanding, intention, and attitude.

This paper develops a definition of the model and the conceptual model that encompasses model notions used in Computer Science, Mathematics, Natural and Social Sciences, and Humanities. Models are artifacts which have a background, a basis, and a context. They are products that are used by community of practice such as programmers, learners, business users, and evaluators. Models must be adequate for the representation of origins, dependable for their signification, functional; thus providing the necessary capability and be able to provide effective deployment.

Keywords. model, models in science, model theory, modelling.

1. Introduction

It is noted (e.g. [1]) (it can be observed indeed also in almost all textbooks on Computer Science) that there is no commonly accepted adequate definition of the concept of the model. This claim is considered as one of the big lacuna of the science and art of modelling.

As a starting point, a model can be simply considered to be a material or virtual artifact (1) which is called model within a community of practice (2) based on a judgement (3) [7] of appropriateness for representation of other artifacts (things in reality, systems, ...) and serving a purpose (4) within this community. We distill in the sequel criteria for an artifact to become a model. We can use two approaches: abstract properties and criteria for artifacts. We can observe, however, that most properties of models are hidden or implicit, and thus not given. The user must be a member of a community and base his/her understanding on the background accepted in this community. This implicitness is the main source of misunderstandings concerning models.

Additionally, models are used in many different deployment scenarios and stories (Gebrauchsspiel [22]). They are used for certain purposes and have a function within their deployment. Often they should not or cannot be used outside of their purpose.
1.1. Stachowiak, Aristoteles, Galilei and Mahr Properties of Models

Models are often defined through abstract properties that they must satisfy [14,15].

(1) **Mapping** property: each model has an origin and is based on a mapping from the origin to the artifact.

(2) **Truncation** property: the model lacks some of the ascriptions made to the original and thus functions as an Aristolean model by abstraction by disregarding the irrelevant.

(3) **Pragmatic** property: the model use is only justified for particular model users, the tools of investigation, and the period of time.

(4) **Amplification** property: models use specific extensions which are not observed in the original.

(5) **Distortion** property: models are developed for improving the physical world or for inclusion of visions of better reality, e.g. for construction via transformation or in Galilean models.

(6) **Idealisation** property: modelling abstracts from reality by scoping the model to the ideal state of affairs.

(7) **Carrier** property: models use languages and are thus restricted by the expressive capacity of these languages.

(8) **Added value** property: models provide a value or benefit based on their utility, capability and quality characteristics.

(9) **Purpose** property: models and conceptual models are governed by the purpose. The model preserves the purpose.

The first three properties were introduced by Stachowiak [12]. The fourth and fifth property were introduced by Steinmüller [13]. The seventh property is discussed by Mahr [9]. The sixth, eight and ninth properties [15] are, however, equally if not the most important ones.

1.2. The Manifold of Meaning for the Word “Model”

The notion ‘model’ carries many meanings: Encyclopedia Britannica [10] contains two main meanings:

1. A model is a miniature representation of something (a model of the dam that was accurate down to the last detail). Synonyms are miniature, and pocket edition. Related words are copy, mock-up, replica, reproduction; dummy, and effigy.

2. A model is something set or held before one for guidance or imitation (Samuel Johnson’s literary style is often used as a model for writers seeking precision and clarity). Synonyms are in this case archetype, beau ideal, ensemble, example, exemplar, ideal, mirror, paradigm, pattern, standard. We may compare this meaning with the word paragon. Related words are (2.1) apotheosis, nonesuch, nonpareil, paragon; (2.2) emblem, symbol, type; (2.3) embodiment, epitome, quintessence; and (2.4) criterion, gauge, touchstone.

The synonyms.org dictionary presents a more detailed picture for the German word ‘Modell’. It consists of 111 synonyms that can be arranged into 10 groups².

²The groups are split along rather generalised lines. We have used the most characteristic synonyms for the presentation of the categories.
(1) **Material representation object**: image, statuary, artwork, sculpture, figure, illustration, gestalt, marionette, ornament;

(2) **Derived object**: derivative, derivable, fork, spinoff, descendant;

(3) **Produced object**: artifact, product, article, result, creation, commodity, achievement, master piece, oeuvre, opus, works;

(4) **Completed object**: completion, achievement, design, accomplishment, implementation, conversion, variant;

(5) **Constructed object**: construction, style of construction, pattern, sample;

(6) **Class of construction**: construction, pattern, mock-up, archetype, type, kind;

(7) **Typical object**: sample, paradigm, probe, prototype, schema;

(8) **Starting object**: initial release, original (edition), draft, guideline, oddball, foggy;

(9) **Ideal object**: ideal, overall concept, antetype, paragon;

(10) **Generalised object**: concept, pattern, plan, principle, schema, standard.

This variety of the definition can, however, be represented by pairs *(meaning, purpose)*. The definition of the model depends on the use of models. The use can be characterised by processes, i.e. the *Gebrauchsspiel* in the sense of Wittgenstein. We relate the definitions of ‘model’ to deployment or exploitation functions and corresponding workflows [16]:

**Description-prescription function**: models as images, figures, standard, opus, exposition, representation, composition, realisation;

**Explanation function**: models ‘gestalt’, pattern, guidance, typ, family or species, original, concept, principle, form, workout;

**Optimisation-variation function**: models as creation, ideal, achievement, probe, article, plan, variant, substitute;

**Verification-validation-testing function**: models as sample, schema, specimen, pattern;

**Reflection-optimisation function**: models as creation, design, construction, type, derivative, master piece, product;

**Explorative function**: models as result, product, work, art piece, metaphor, paradigm, first edition, style, realisation, artefact;

**Hypothetical function**: models as copy, release, original form, offshoot, simulation or experiment product;

**Documentation-visualisation function**: models as presentation, figure, illustration, demonstration, explanation, adornment, plastic, structure.

### 1.3. The Story of the Paper

Models are working instruments which are widely used in Computer Science, Mathematics, Natural and Social Sciences, and the Humanities. For instance, there are many different notions in Business Informatics, e.g., [19]. However, most of them are rather informal statements and cannot be considered to be definitions. This paper aims at providing a definition of the notion of a conceptual model. We start first with a case study and then introduce a general notion of a model in sequel.
2. Models, Models, Models in Computer Science

2.1. The Variety of Models Used in Computer Science

Computer Science uses more than 50 different kinds of models in all of its sub-disciplines [18]. It seems to be difficult to survey all of these approaches. We may, however, classify models based on their orientation. Models are used for understanding, for a description of the general architecture of a system, for investigation, and for construction of systems. Additionally, models are also used as a guidance for the development process of software and hardware systems. Therefore, we may visualise the range of models in Figure 1.

![Figure 1. A general classification of models in computer science](image)

The main purpose of models developed and deployed in Computer Science is the construction of systems. Therefore, the function of a model is to give a description of the part of the application world, to use this description as a prescription for requirements, to base on the model the specification of the system, and finally to realise the system [16]. Therefore, we have to distinguish the relevance stage, the modelling stage, and the realisation stage and may associate the model to its functions within these stages. The variety of construction modelling languages is very large. The UML uses more than 140 different kinds of diagrams for different aspects. We can recognise a good number of system modelling languages and kinds of system models. We may concentrate on different aspects of a system such as task models, usage models, object models, component
models, process models, functional models, state models, event models, and information system models. The latter kind of models can also be separated according to a variety of aspects. We thus arrive at an overview in Figure 2.

This variety of models in courses of Computer Science [18] has a tendency to confuse. We target however on a general view of the notion of model. We observe that these models are oriented towards their specific purpose. As already observed by [13], the main purposes are: (a) construction of systems, (b) explanation of structure, computation and control of systems, and (c) quality assurance of software and hardware systems. Models should not be used outside their purpose.

The variety of models and conceptual models in Computer Science is overwhelming. But at the same time we observe a number of commonalities. Models are driven by their purpose and thus incorporate thus methods for their language-based development. The main purpose is construction of systems. There are, however, also other purposes of interest, such as theoretical investigation. The community of practice consists typically of authors and addressees. Models may also be distinguished by their aspects such as static or dynamic, by their abstraction level, such as micro or macro, by their level of formality and by their capability, such as syntactic or semantic concerns.

We may learn from these models. There are general rules for (α) combination of models, (β) separation of concern, (γ) model construction, (δ) theoretical foundations etc.
2.2. A Case Study for one of the Models: The Turing Machine

The Turing Machine is one of the most widely used models for computation in Computer Science. A Turing Machine is given, for instance, as a 7-tuple \((Q, \Sigma, \Gamma, B, q_0, F, \delta)\) with a finite, non-empty set of states \(Q\), a finite, non-empty input alphabet \(\Sigma\), a finite tape alphabet \(\Gamma\) with \(\Gamma \supseteq \Sigma\) and the blank symbol \(B\), a start state \(q_0 \in Q\), a set of final states \(F \subseteq Q\), and a partial state transformation function \(\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{R,N,L\}\).

The Turing Machine is often defined using some kind of Von Neuman architecture similar to the one in Figure 3. Instead of a state transformation function we could use a control unit for computation on \(\Gamma\) and \(Q\) through a state transformation relation. Instead of one tape we could use a finite set of tapes. It is however proven that any general Turing Machine can be represented by a machine in the form introduced here.

The second main element of the model of a Turing Machine is its computation process for a Turing Machine. It uses the concept of configuration. A (finite) configuration \(wqv\) is given by two words \(w, v \in \Gamma^+ \cup \{\lambda\}\) (for the empty word \(\lambda\)) and a state \(q \in Q\). It defines that the tape of the Turing Machine is empty (populated by \(B\)) except the word \(wv\), that the head is on the first letter of \(v\) and that the state of the Turing Machine is \(q\). Given a configuration \(wqv\) then the next configuration is determined by \(\delta\). A configuration \(wqv\) is final if \(q \in F\). The result of computation is then \(v\). The initial configuration is \(q_0v_0\).

There is a multiplicity of books which introduces this model for computation. It is commonly believed that any computational process can be described through a computation of Turing Machines.

Turing Machines represent a general notion of algorithm or computer from one side and are a representation of a computation from the other side. The representation of computation serves at least the two purposes in Figure 4.

Turing Machine represent a variety of computational systems such as Von Neuman computers or 1-address machines. They do, however, not represent interactive machines (HCI machines), analog computers or parallel computer systems. We observe, however, four properties: (a) they are well-formed (coherent, consistent, non-ambiguous) in the
sense of mathematics; (b) they behave analogously to computers considered; (c) they are far simpler than those computers; (d) they can be used for understanding computation in the sense of Von Neuman, for proof of computability or non-computability, for proof of expressive power of other computation models or other variants of Turing Machines, for consideration of complexity of problems which are computable or decidable. They are therefore adequate.

Turing Machines are justified: (α) they are corroborated by accepted understanding of state-transforming computations; (β) they are rationally coherent and conform in semiotics given by first-order predicate logics; (γ) they support exclusion of non-intended next world transformations and allow to falsify which kind of transformation is not considered; (δ) they are stable and plastic, e.g., robust since adding tapes or control units, coding of alphabets, restricting to (one-sided) tapes do not change the notion of computable functions. Additionally, they are sufficient due to firm quality for other models of computation in general due to their (i) internal quality (e.g., simplicity, minimality, robustness), (ii) external quality (e.g., correctness, comprehensibility, generality), and (iii) quality in use (e.g., parsimony). Due to viability and firm quality Turing Machines are plausible devices for computation in general. This judgement of plausibility is based on the evaluation of these machines as precise, rigid and non-tolerant mechanisms without any modality and with full confidence of the computational process.

There are, however, many hidden assumptions, paradigms, postulates, and conventions which are intentionally used but not defined in an explicit form.

Paradigms of computing: we typically assume a Von-Neuman architecture of machines and thus bound computation to determinism and sequentiality and use a separation into storage, computation and control. Computation is based on first-order predicate logics in its canonical form. Results of computation are well-determined and do not depend on context. Computation is performed by next-step transformations (sequential time postulate [2]) and thus uses a special time model.

Observation (1): Paradigms may impose a number of restrictions to the model.

Postulates of computing: computations are state transformations (abstract state postulate [3,5]). Each computational step uses a finite set of resources (bounded exploration postulate [3,5]). State transformations are carried out if they do not result in update conflicts (update postulate [3,5]). Coding of the alphabets does not influence computation (isomorphism postulate [2]). The finite result function entirely depends on the input (interaction postulate [2]). Parallelity can be bounded (bounded choice non-determinism postulate [3,5,20]). The alphabets might be defined on infinite resources (background
Observation (2): Postulates restrict the model to a specific form.

Assumptions and principles for Turing Machines: a machine has exclusive rights on the data space. The abstraction is not changed for different time or steps. The result of a computation step is the final result and cannot be negotiated. The final result is achieved if we reach a final state. The meaning is definable on top of the (first-order predicate) syntax. Input and output are finite. The tape can only be changed at the place of the read/write head. Implicit principles such as compositionality, functional or relational state transformations, step-wise computation, and context-freeness.

Observation (3): Assumptions and principles might be negotiated or changed.

Background models, languages and theories and resulting restrictions: Turing Machine computation is based on the sequential time model. Each step is performed sequentially. Languages used are logical ones, functions or relations, and graphs. Background theories are, for instance, logic and canonical set theory. Non-trivial semantical properties of Turing Machines are undecidable (Rice Theorem).

Observation (4): Background heavily impacts on the model.

Thought community and thought style: the thought community uses a tape that is infinite, a finite set of states, unchangeability of the transformation function, and a separation into control and storage units. Therefore, these machines cannot be certified (Second Rice Theorem).

Observation (5): The thought approach is an implicit and authority-driven background.

Profile, goal, purpose and function of the model: the profile of a model is defined through its goal, purpose or function [14,15,17]. The profile of the Turing Machine is oriented on a proof of computability and the corresponding computational complexity. However, it does not consider many aspects, e.g., efficient programs. Construction of real machines is beyond its purpose.

Observation (6): The model should not be used beyond its profile.

Concepts and cargo used within the model: the transformation is based on the notion of a function, uses a coded alphabet and recursion. The cargo of the Turing Model is based on the definition of this model as step-wise computation and potential alternatives. It uses time or space complexity measures, movement of either head or tape, termination through reaching a final state, and language and word algebras.

Observation (7): Concept enrichment depends on the application domain.

CS approaches do not try to get to the bottom of paradigms, postulates, foundations, theories, assumptions, principles, concepts or cargo, thought community or the background. We must, however, consider this whenever we want to understand and to properly use a model. We call these elements of an artifact fundament.

However, unless we do not have methods for its utilisation within its deployment, the Turing Machine itself is useless. Most methods used originate from Mathematical Logistics and Algebra. Complexity is based on tape or transformation step complexity. These complexity measures allow to use methods from Mathematical Analysis. Therefore, we can prognose the time or space consumed for certain computations. Furthermore, graph theory is used for visualisation through finite labelled graphs. The development of a state transformation function, the integration of machines, the exploration and interpretation is not supported. Turing Machine models are considered to be models that are not to be developed beside the development of a state transformation function. The deploy-
\textit{ment recount} is mainly oriented towards a description of computation and prognosis of computability with or within bounded resources. Construction, definition, exploration, communication, documentation of real computers and other functions of artifacts are not supported by Turing Machines.

2.3. The Informal Notion of a Model

To summarise: a model can be any real or virtual artifact that represents other artifacts. The artifact must, however, be adequate and plausible. Each artifact introduces its fundament. An artifact is deployed as an instrument in science and engineering. This deployment is evolving and might incorporate other artifacts. Therefore, we might use \textit{model suites} [4] instead of a singleton model. This deployment is based on the profile or capacity of the artifact. Artifacts are used in a context by some users with their own intentions, goals and tasks. Therefore, an artifact serves their goals, purposes or functions within the given portfolio.

The example in the previous subsection allows us to gain an insight into the properties of a model. The mapping property cannot be defined through some kind of homomorphism of structure and/or behaviour. It is far more complex. Models are, however, abstractions of some origins. Therefore, any model supports truncation. Models are used within their profile and for the tasks defined for the deployment portfolio. Therefore, the pragmatic property is essential. Models might augment reality. Turing Machines show how amplifications might work. They also have properties of Galilean models. They idealise the computational mechanism. The languages used may be far more strict and rigid than those that may be be used for the origin. Models are used as instruments and bring in some added value. Models serve a purpose based on their profile.

The deployment ‘game’ of a model is based on an embedding into the story space of the specific discipline. Models such as Turing Machines are neither a good means for construction of an architecture of a computer nor for well-designed programs. They do not allow to reason on information systems beside the general notion of computation. The Turing Machine deployment is a typical explorative function.

3. The Notion of a Model

3.1. The Formal Notion of a Model

Any artifact can be used as a model. It faithfully or dependable represents other artifacts and must provide facilities or features for its use. Based on our observations (1) - (8) we conclude that a model is implicitly based on the \textit{background of a model} consisting of a \textbf{basis} $B$ from one side, i.e., basement, paradigms, postulates, restrictions, theories, culture, foundations, conventions, commonsense and \textbf{grounding} $G$ from other side, i.e., concepts, foundations, language as carrier, and the cargo,

\textbf{context} $C$ such as application domain or discipline, school of thought, time, space, granularity and scope, and

\textbf{community of practice} $\mathcal{P}$ with corresponding roles and potentially with specific plays of these roles, and rights, obligations, and the practice commonly accepted within this CoP $\mathcal{P}$.
Given now a community of practice $\mathcal{P}$ ($\subseteq \mathcal{D}$), a grounding $\mathcal{G}$ ($\subseteq \mathcal{C}$), bases $\mathcal{B}$ ($\subseteq \mathcal{E}$), and a context $\mathcal{C}$ ($\subseteq \mathcal{C}$). Artifacts are explicitly or mostly implicitly given together with some grounding $\mathcal{G}$, with bases $\mathcal{B}$ and within some context $\mathcal{C}$ by a community of practice $\mathcal{P}$. We call these artifacts ($\mathcal{G}, \mathcal{B}, \mathcal{C}, \mathcal{P}$) artifacts.

An artifact is a profile. The profile is based on the goal or purpose or function of the artifact. If the artifact is devoted to its profile $\mathcal{D}$ the artifact is called purposeful, given the following measures and thresholds: an analogy criterion or an analogy measure \textit{analogy($A^*$; $\mathcal{A}$)} for artifacts (or collections of artifacts), a analogy threshold $\Theta_{\text{analogy}}$; a complexity measure for artifacts \textit{complex($M$)} and a simplification threshold $\Theta_{\text{complex}}$.

A ($\mathcal{G}, \mathcal{B}, \mathcal{C}, \mathcal{P}$) artifact $A^*$ is called \textit{adequate} for a collection of artifacts $\mathcal{A}$ if
- it is well-formed according to $\gamma_{\text{form}}$ and governed by a ($\mathcal{C}, \mathcal{P}$) convention,
- it is analogous to the artifacts $\mathcal{A}$ to be represented according to some analogy criterion, e.g., \textit{analogy($A^*$, $\mathcal{A}$)} > $\Theta_{\text{analogy}},$
- it is simpler than the artifacts $\mathcal{A}$ according to some complexity criterion, e.g., \textit{complex($A^*$)} < $\Theta_{\text{complex}} \cdot \text{complex($\mathcal{A}$)}$, and
- it is purposeful for the profile $\mathcal{D}$.

An artifact is \textit{justified} by a justification $J$, i.e. [6], by empirical corroboration (according to purpose, background, etc.) for the representation of the artifacts $\mathcal{A}$ that is supported by some argument calculus ($\text{BasicArguments, Corrobolation}$), by rational coherence and conformity explicitly stated through formulas ($\gamma_{\text{coherence}}, \gamma_{\text{conform}}$), by falsifiability that can be given by an an abductive and/or inductive logical system ($\text{BasicExclusion, Falsifiable}$) (with tests, reduction, parsimony), and by stability and plasticity (depending on the scope, grounding, basis, context and quality) explicitly given through formulas ($\gamma_{\text{stability}}, \gamma_{\text{plasticity}}$). The artifact is \textit{sufficient} by its quality characterisation $Q$ for internal quality, external quality and quality in use or through quality characteristics [14] such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Justification and sufficiency characterise the signification of an artifact for deployment, deployment and degree of precision efficiency for satisfying the deployment necessities, and extent of coverage depending on deployment. It is typically combined with some assurance evaluation $E$ (tolerance, modality, confidence, and restrictions) for $\mathcal{A}$.

An artifact $A^*$ is called ($J$, $\Omega$, $E$) \textit{dependable} for some of the justification properties $J$ ($\subseteq J$) and some of the sufficiency characteristics $\Omega$ ($\subseteq \mathcal{Q}$) if
- the quality criteria $\Omega$ are satisfied through $E$, and
- it is justified by $J$ through $E$.

A ($\mathcal{G}, \mathcal{B}, \mathcal{C}, \mathcal{P}$) artifact $A^*$ is called \textit{model} of $\mathcal{A}$ if it is $\mathcal{A}$ \textit{adequate} and ($J$, $\Omega$, $E$) \textit{dependable}.

The model and the artifact are \textit{functional} if there are methods $M_D$ for utilisation of the artifact in dependence on the profile $\mathcal{D}$ ($\subseteq \mathcal{D}$) of the artifact. In this case the artifact provides a service. Functional artifacts have their capability and capacity [14]. They are thus deployable. Additionally, development methods $M_E$ for development of artifacts might exist.

Artifacts are used for application cases. These cases are embedded into application stories such as description-prescription, explanation, optimisation-variation, verification-validation-testing, reflection-optimisation, exploration, hypothesis development, documentation-visualisation, or also for substitution. These application stories (or
‘deployment games’ [22]) are supported by the task portfolio [8] which an artifact might serve. Typical tasks include defining, constructing, exploring, communicating, understanding, replacing, substituting, documenting, negotiating, replacing, reporting, and accounting. We call an artifact and a model effective if it can be deployed according to its portfolio.

3.2. Facets of Models

A (matured) model is thus an artifact that uses

**a fundament** with
- the grounding, and
- the (meta-)basis,

**four governing directives** given by
- the artifacts to be represented by the model,
- the deployment or profile of the model such as goal, purpose or functions,
- the community of practice acting in different roles on certain rights through some obligations, and
- the context of time, discipline, application and scientific school,

**two pillars** which provide
- methods for development of the model, and
- methods for utilisation of the model,

and finally

**the model portfolio and function** for the deployment of the model in the given application.

The *model house* in Figure 5 displays these different facets of the model. The house consists of a cellar and a fundament, two pillars, four driving or governing forces, and finally the deployment roof. The *grounding* is typically implicitly assumed. It contains paradigms, the culture in the given application area, the background, foundations and theories in the discipline, postulates, (juristical and other) restrictions, conventions, and the commonsense. The *basis* is the main part of the background. It is typically given for modelling. The development uses a variety of methods for *description, construction, evolution, corroborate*, and *evaluation*. The utilisation is based on methods for *applying, prognosis, reasoning, explanation, and exploration*. We have used different verbs for classification of the activities. The model can be used for completion of certain tasks. These tasks may be combined into a *model portfolio*. The model is used for certain functions or deployment scenario (*Gebrauchsspiel*). Finally, the model is governed by four directives: *artifacts, profile, community of practice, and context*.

4. Conceptual Models and Conceptional Modelling

4.1. Differences between Conceptual Models and Conceptional Modelling

The words ‘conceptual’ and ‘conceptional’ are often considered to be synonyms. R.T. White [21] has already observed that concepts are not the same as conceptions. Concepts can be used in the meaning of classification and as an abstraction of a set of knowledge
Figure 5. Facets of the model with grounding and basis as the fundament, with four governing directives, with technical and technological pillars for development and utilisation, and with the application roof

A person associates with the concept’s name. However, conceptions are systems of explanation. Conceptions are thus far more complex and difficult to define than the either meanings of the concept.

The word ‘conceptual’ is linked to concepts and conceptions. Conceptual means that a thing, e.g. artifact is characterised by concepts or their conceptions. The word ‘conceptional’ associates a thing as being or being of the nature of a notion or concept. Therefore, we distinguish the ‘conceptual model’ from ‘conceptional modelling’. Conceptual modelling is modelling with associations to concepts. A conceptual model incorporates concepts into the model.

4.2. Conceptional Modelling: Modelling Enhanced by Concepts

An information systems model is typically a schematic description of a system, theory, or phenomenon of an origin that accounts for known or inferred properties of the origin and may be used for further study of characteristics of the origin. Conceptional modelling aims to create an abstract representation of the situation under investigation, or more precisely, the way users think about it. Conceptual models enhance models with
concepts that are commonly shared within a community or at least between the stakeholders involved in the modelling process. A general definition of concepts is given in [15]. Concepts specify our knowledge what things are there and what properties things have. Concepts are used in everyday life as a communication vehicle and as a reasoning chunk. Concept definition can be given in a narrative informal form, in a formal way, by reference to some other definitions, etc. We may use a large variety of semantics [11], e.g., lexical or ontological, logical, or reflective.

**Conceptualisation** aims at collection of objects, concepts and other entities that are assumed to exist in some area of interest, and the relationships that exist amongst them. It is thus an abstract, simplified view or description of the world that we wish to represent.

### 4.3. Conceptualisation of Models

Conceptualisation extends the model by a number of concepts that are the basis for an understanding of the model and for the explanation of the model to the user. A general theory of concepts that has been used for conceptualisation was introduced by [15]. Concepts are used in everyday life as a communication vehicle and as a reasoning chunk.

Our revision of the design science and the conceptual modelling frameworks in [4] shows that concept enhancement is an orthogonal activity. It can be partially completed without having a negative impact on the realisation phase if stakeholders (i.e., the community of practice) involved have a common understanding of the model and its properties, and a commonsense about the objectives for realisation and a good mapping facility. Therefore, conceptualisation may also be implicit and may use some kind of lexical semantics, e.g. word semantics, within a commonly agreed name space.

### 5. Conclusion

In this paper we introduced a formal definition of the notion of a conceptual model. This notion has been applied and tested in Computer Science, in Philosophy, in Physics, and other sciences. As far as we discovered the notion is sufficient.

We have been aiming at development of a formal notion of a model. Such formal notion is necessary whenever we need a theory of conceptional modelling. It allows to exclude artifacts to become a model outside the judgement frame.

### References


Acknowledgment. I would like to thank my colleagues at the Christian-Albrechts University Kiel for the fruitful discussions on many facets of models within the context of the faculties and institutes of Archeology, Arts, Biology, Chemistry, Computer Science, Economics, Electrotechnics, Environmental Sciences, Farming and Agriculture, Geosciences, Historical Sciences, Humanities, Languages and Semiotics, Mathematics, Medicine, Ocean Sciences, Pedagogical Science, Philosophy, Physics, Political Sciences, Sociology, and Sport Science. We are thankful to the International Institute of Theoretical Cardiology (IIfTC) for the evaluation of our approach. These discussions lasted in weekly ‘Tuesday’ open-end evening seminars from 2009 until 2012 and in monthly seminars at the IIfTC. They resulted in a general understanding of the notion of a model in most sciences, engineering and technology.
Das Modell des Modelles

Bernhard Thalheim

October 5, 2014

Vorbemerkung


Eingrenzung

(4) Ein allgemeiner Modellbegriff wurde in einer Vielzahl von Arbeiten - moderne Arbeiten folgen meist den Ansätzen von H. Stachowiak [5, 1] - seit der En-

**Ein Modellbegriff**

((5)) Ein Modell ist ein Instrument, das adäquat und verläßlich andere Originale repräsentiert je nach beabsichtigtem Zweck (oder Funktion in der Nutzung dieses Instrumentes oder auch damit verbundenem Ziel)[6, 7]. Das Modellsein hängt von der Nutzergemeinschaft und dem Kontext ab.


Zweckmäßige Instrumente sollen für den Zweck tauglich sein, d.h. ihnen sind Methoden zur Nutzung und zur Entwicklung zugeordnet.

Instrumente sollen als Modelle in Nutzungsszenario effektiv eingesetzt. Solche Instrumente sind dann effektive Modelle.


((7)) Ein (wohldefiniertes) Instrument heißt adäquat für die Repräsentation von Originalen, wenn es zum einem analog aufgrund eines Analogiemaßes, zum zweiten fokussierter aufgrund eines Komplexitätsmaßes und zielgerecht für entsprechende angegebene Ziele ist.

**Der Cargo**


**References**


Bernhard Thalheim  
Institut für Informatik  
Christian-Albrechts-Universität zu Kiel, 24098 Kiel, Germany  
thalheim@is.informatik.uni-kiel.de
27 The Notion of a Model

Definition: A model is a well-formed, adequate, and dependable instrument that represents origins. Its criteria of well-formedness, adequacy, and dependable must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilization scenarios and use spectra. As an instrument, a model is grounded in its community’s sub-discipline and is based on elements chosen from the sub-discipline.

27.1 The Conception of the Model

Science and technology widely uses models in a variety of utilization scenarios. Models function as an instrument in some utilization scenarios and a use spectrum. Their function in these scenarios is a combination of functions such as explanation, optimization-variation, validation-verification-testing, reflection-optimization, exploration, hypothetical investigation, documentation-visualization, and description-prescription as a mediator between a reality and an abstract reality that developers of a system intend to build. The model functions determine the purposes of the deployment of the model.

Models have several essential properties that qualify an instrument as a model (Tha12; Tha14):

– An instrument is well-formed if it satisfies a well-formedness criterion.
– A well-formed instrument is adequate for a collection of origins if (i) it is analogous to the origins to be represented according to some analogy criterion, (ii) it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and (iii) it sufficient to satisfy its purpose.
– Well-formedness enables an instrument to be justified: (i) by an empirical corroboration according to its objectives, supported by some argument calculus, (ii) by rational coherence and conformity explicitly stated through formulas, (iii) by falsifiability that can be given by an abductive or inductive logic, and (iv) by stability and plasticity explicitly given through formulas.
– The instrument is sufficient by a quality characterisation for internal quality, external quality and quality in use or through quality characteristics (Tha10) such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).
A well-formed instrument is called *dependable* if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

An instrument is called *model* if it is *adequate* and *dependable*. The adequacy and dependability of an instrument is based on a *judgement* made by the community of practice.

An instrument has a *background* consisting of an undisputable grounding from one side (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities) and of a disputable and adjustable basis from other side (assumptions, concepts, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense).

A model is used in a *context* such as discipline, a time, an infrastructure, and an application.

Not only should a model faithfully represent a collection of origins by being well-formed, adequate, and dependable, it should also provide facilities or methods for its use. A model is *functional* if there are methods for utilization of the instrument to achieve the objectives for which an instrument might serve. Typical task objectives include defining, constructing, exploring, communicating, understanding, replacing, substituting, documenting, negotiating, replacing, optimizing, validating, verifying, testing, reporting, and accounting. We call a model *effective* if it can be deployed according to its objectives.

### 27.2 Properties of Models

Models satisfy several properties that make them functional and effective (Mah08; Mah15; Sta73; Tha10; Tha11; Tha12; Tha14):

1. *Mapping* property: the model has an origin and can be based on a mapping from the origin to the instrument.

2. *Analogy* property: the model is analogous to the origins based on some analogy criterion.

3. *Truncation* (or *reduction*) property: the model lacks some of the ascriptions made to the origin and thus functions as an Aristotelian model by abstraction by disregarding the irrelevant.

4. *Pragmatic* property: the model use is only justified for particular model users, the tools of investigation, and the period of time.

5. *Amplification* property: models use specific extensions which are not observed in the original.
(5) *Idealisation* property: modelling abstracts from reality by scoping the model to the ideal state of affairs.

(6) *Carrier (cargo)* property: models reflect a conception on origins based on the capacity of a language and are filled with anticipation. They carry a cargo(Mah08).

(6') *Utilisation* property: the model functions well within its intended scenarios of usage according to its capacity and potential.

(7) *Divergence* property: models (e.g. Galilean models) are developed for improving divergence, deviation, discrepancy the physical world or for inclusion of visions of better reality, e.g. for construction via transformation.

(8) *Added value* property: models provide a value or benefit based on their utility, capability and quality characteristics.

(9) *Purpose* property: models are governed by the purpose. The model preserves the purpose.

**Literatur**


Enhancing Entity-Relationship Schemata for Conceptual Database Structure Models

Bernhard Thalheim and Marina Tropmann-Frick

Department of Computer Science, Christian-Albrechts-University Kiel, 24098 Kiel, Germany

Abstract. The paper aims at development of well-founded notions of database structure models that are specified in entity-relationship modelling languages. These notions reflect the functions a model fulfills in utilisation scenarios.

1 Utilisation Scenarios of Conceptual Models

Conceptual models are used as an artifact in many utilisation scenarios. Design science research [4] and ER schema development methodologies (e.g. [3, 8, 11]) developed so far a good number of such scenarios.

Communication and negotiation scenario: The conceptual model is used for exchange of meanings through a common understanding of notations, signs and symbols within an application area. It can also be used in a back-and-forth process in which interested parties with different interests find a way to reconcile or compromise to come up with an agreement. The schema provides negotiable and debatable propositions about the understanding of the part of the reality but does not have well-developed justificatory explanations.

Conceptualisation scenario: The main application area for extended entity-relationship models is the conceptualisation of database applications. Conceptualisation is typically shuffled with discovery of phenomena of interest, analysis of main constructs and focus on relevant aspects within the application area. The specification incorporates concepts injected from the application domain.

Description scenario: In a description scenario, the model provides a specification how the part of the reality that is of interest is perceived and in which way augmentations of current reality are targeted. The model says what the structure of an envisioned database is and what it will be.

Prescription scenario: The conceptual model is used as a blueprint for or prescription of a database application, especially for prescribing the structures and constraints in such applications. The schema proposes what the structure of a database is on the one hand and how and where to construct the realisation on the other hand. ER schemata can be translated to relational, XML or other schemata based on transformation profiles [11] that incorporate properties of the target systems.
These scenarios are typically bundled into use spectra. For instance, design science uses three cycles: the relevance cycle based on the design cycle based on description and communication and negotiation scenario, and the rigor cycle based on a knowledge discovery and experience propagation scenario. Database development is mainly based on description, conceptualisation, and construction scenarios. The re-engineering and system maintenance use spectrum is based on combination of documentation scenarios with an explanation and discovery scenario from one side and communication and negotiation scenario from the other side. Models are also used for documentation scenarios, explanation and discovery scenarios for applications or systems, and for knowledge experience scenario. We concentrate here on the four scenarios.

Contribution of the Paper

The first main contribution of this paper is an analysis whether an entity-relationship schema suffices as a model for database structures. We realise that the four scenarios require additional elements for the ER schema in order to become a model. The second main contribution of this paper is a proposal for an enhancement of ER schemata which allows to consider the artifact as a model within the given four scenarios. The paper partially presupposes our research (esp. [15], see also other papers in [16]).

2 The General Notion of a Model

Science and technology widely use models in a variety of utilisation scenarios. Models function as an artifact in some utilization scenario. Their function in these scenarios is a combination of functions such as explanation, optimization-variation, validation-verification-testing, reflection-optimization, exploration, hypothetical investigation, documentation-visualization, and description-prescription as a mediator between a reality and an abstract reality that developers of a system intend to build. The model functions determine the purposes of the deployment of the model.

The following notion of the model has been developed [17] after an intensive discussion in workshops with researchers from disciplines such as Archeology, Arts, Biology, Chemistry, Computer Science, Economics, Electrotechnics, Environmental Sciences, Farming and Agriculture, Geosciences, Historical Sciences, Humanities, Languages and Semiotics, Mathematics, Medicine, Ocean Sciences, Pedagogical Science, Philosophy, Physics, Political Sciences, Sociology, and Sport Science.

Definition 1. A model is a well-formed, adequate, and dependable artifact that represents origins. Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilisation scenarios. As an artifact, a model is grounded in its community’s sub-discipline and is based on elements chosen from the sub-discipline.
This notion has been tested against the notions of a model that are typically used in these disciplines. We could state that these notions are covered by our notion. Origins of a model [7] are artifacts the model reflects. Adequacy of models has often been discussed, e.g. [6, 9, 10]. Dependability is only partially covered in research, e.g. [5].

Models have several essential properties that qualify an artifact as a model [15, 16]:

- An artifact is well-formed if it satisfies a well-formedness criterion.
- A well-formed artifact is adequate for a collection of origins if (i) it is analogous to the origins to be represented according to some analogy criterion, (ii) it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and (iii) it sufficient satisfies its purpose.
- Well-formedness enables an artifact to be justified: (i) by an empirical corroboration according to its objectives, supported by some argument calculus, (ii) by rational coherence and conformity explicitly stated through formulas, (iii) by falsifiability that can be given by an abductive or inductive logic, and (iv) by stability and plasticity explicitly given through formulas.
- The artifact is sufficient by its quality characterisation for internal quality, external quality and quality in use or through quality characteristics [13] such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).
- A well-formed artifact is called dependable if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.
- An artifact is called model if it is adequate and dependable. The adequacy and dependability of an artifact is based on a judgement made by the community of practice.
- An artifact has a background consisting of an undisputable grounding from one side (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities) and of a disputable and adjustable basis from other side (assumptions, concepts, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense).
- A model is used in a context such as discipline, a time, an infrastructure, and an application.

The Taxonomy of Conceptual Models

The starting point in our investigation was the observation that there is no unique and commonly agreeable notion of the conceptual database structure model as such. The model supports different purposes and has different functions in utilisation scenarios. Therefore, we must have different notion of the conceptual database structure model.

The conceptual model functions within the utilisation scenarios in different roles with different rigidity, modality and confidence. Models are used as perception models (reflection of one party's current understanding of world; for
understanding the application domain), situation models (reflection of a given state of affairs), conceptual models (based on formal concepts and conceptions) experimentation models (as a guideline and basis for experimentation), formal models (based some formalism within a well-based formal language), mathematical models (in the language of mathematics), computational models (based on some (semi-)algorithm), physical models (as physical artifact), visualisation models (for representation using some visualisation), representation models (for representation of some other notion), diagrammatic models (using a specific language, e.g. UML), exploration models (for property discovery), and heuristic models (based on some Fuzzyness, probability, plausibility, correlation), etc. The large variety of known and used model notions (e.g. see [14, 15, 17] and the collection [16]) mainly reflects these different kinds. The situation model might use a rigorous structured English (OMG proposal) and represents the nature of the business within the language of the business. In this case it is also called reality model.

Due to space limitations we concentrate here on the four utilisation scenarios described in section 1. The other scenarios, such as documentation scenario, explanation and discovery scenario for applications, explanation and discovery scenario for systems and knowledge discovery and experience propagation scenario, are supported by specific conceptual models in a similar form.

3 Conceptual Database Structure Models for Communication and Negotiation

Communication aims at exchange of meanings among interested parties. The model is used as a means for communication. It truly represents some aspects of the real world. It enables clearer communication and negotiation about those aspects of the real world. It has therefore potentially several meanings in dependence on the parties. Communication acts essentially follow rhetoric frames\(^1\), i.e. they are characterised through “who says what, when, where, why, in what way, by what means” (Quis, quid, quando, ubi, cur, quem ad modum, quibus adminiculis). In our case, the model (“what”) incorporates the meaning of parties (semantical space; “who”) during a discourse (“when”) within some application with some purpose (“why”) based on some modelling language.

Typically, artifacts used for communication and negotiation follow additional principles: Viewpoints and specific semantics of users are explicitly given. The artifact is completely logically independent from the platform for realisation. The name space is rather flexible. The model is functioning and effective if methods for reasoning, understanding, presentation, exploration, explanation, validation, appraisal and experimenting are attached.

Conceptual model for communication: The conceptual database structure model comprises the database schema, reflects viewpoints and perspectives of different involved parties \(U\) and their perception models, and implicitly links to (namespaces or) concept fields of parties. Adequacy and dependability are based on the

\(^1\) It relates back to Hermagoras of Temnos or Cicero more than 2000 years ago.
association of the perception models to viewpoints and of the viewpoints with the schema.

A partial communication model does not use a schema and does not associate viewpoints to schema elements. Therefore, the model can be formally defined as a quintuple

\[(S, \{(V_i, \Phi_i) | i \in U\}, \{(P_i, \Psi_i) | i \in U\}, A, D)\]

that relates elements of the conceptual schema \(S\) to the perception model \(P_i\) of the given party \(i\). The perception model is reflected in the schema via viewpoints \(V_i\). It implicitly uses concept fields \(C_i\) of parties \(i\). The mapping \(\Psi_i : P_i \to V_i\) associates the perception model of a given party \(i\) to the agreed viewpoint. In the global-as-design approach, the viewpoint \(V_i\) is definable by some constructor \(\Phi_i\) defined on \(S\). The adequacy \(A\) is directly given by the second and third parts of the model. The justification \(J\) and the dependability \(D\) are extracted from the properties of \(\Phi_i\) and \(\Psi_i\).

The negotiation scenario can thus be understood as stepwise construction of the mappings, stepwise revision of the schema and the viewpoints, and analysis whether the schema represents the corresponding perception model.

4 Conceptual Database Structure Models for Conceptualisation

Conceptualisation is based on one or more concept or conception spaces of business users. Given a business user community \(U\) with their specific concept fields \(\{C_i | i \in U\}\). Let us assume that the concept fields can be harmonised or at least partially integrated into a common concept field of users \(C_U\) similar to construction approaches used for ontologies.

Conceptual model for conceptualisation: The conceptual database structure model comprises the database schema, a collection of views for support of business users, and a mapping for schema elements that associates these elements to the common concept field.

Therefore, the model can be formally defined as a quintuple

\[(S, \mathcal{V}, \mathcal{M}, A, D)\]

consisting of the conceptual schema \(S\) and a mapping \(\mathcal{M} : S \to C_U\). The adequacy \(A\) is based on the mapping. The justification \(J\) and the dependability \(D\) are derived from the concept fields.

5 Conceptual Database Structure Models for Description and Prescription

An artifact that is used as a conceptual model for database system description can be either understood as a representation, refinement and amplification \([13, 16]\) of situation or reality models or as a refinement and extension of the
communication model. The main approach to conceptual modelling for system construction follows the first option. The second option would however be more effective but requires a harmonisation of the perception models. The first option may start with reality models that reflect the nature of the business in terms and in the language of the business. It includes also the top management view, a corporate overview, and a sketch of the environment. The reality models are reasonable complete, are described in terms of the business and use general categories that are convergent.

Conceptual model for description: The conceptual database structure model comprises the database schema, a collection of views for support of business users, a collection of a commonly accepted reality models that reflects perception or situation models with explicit association to views, and the declaration of model adequacy and dependability.

Therefore, the model can be formally defined as a quintuple

\[(S, \mathfrak{V}, (\mathfrak{R}, \Psi), \mathfrak{A}, \mathfrak{D})\].

The conceptual model may be enhanced by an association \(\Phi\) of views to the schema. This enhancement is however optional.

Descriptive models adequately explicate main concepts [12] from the reality models and combine them into views. The descriptive model reflects the origins and abstracts from reality by scoping the model to the ideal state of affairs.

Prescriptive models that are used for system construction are filled with anticipation of the envisioned system. They deliberately diverge from reality in order to simplify salient properties of interest, transforming them into artifacts that are easier to work with. They may follow also additional paradigms and assumption beyond the classical background of conceptual database structure models: Salami slicing of the schema by rigid separation of concern for all types; conformance to methods for simple (homomorphic) transformation; adequateness for direct incorporation; hierarchical architecture within the schema, e.g. for specialisation and generalisation of types; partial separation of syntax and semantics; tools with well-defined semantics; viewpoint derivation; componentisation and modularisation; integrity constraint formulation support; conformance to methods for integration; variations for the same schema for more flexible realisation etc.

Directives (or pragmas) [1] prescriptively specify properties for the realisation. Transformation parameters [11] for database realisation are, for instance, treatment of hierarchies, controlled redundancy, NULL marker support, constraint treatment, naming conventions, abbreviation rules, set or pointer semantics, handling of weak types, and translation options for complex attributes. Based on [2] we give an explicit specification of directives for the realisation. The prescription model also consists of a general description of a realisation style and tactics, of configuration parameters (coding, services, policies, handlers), of generic operations, of hints for realisation of the database, of performance expectations, of constraint enforcement policies, and of support features for the system realisation. These parameters are combined to the realisation template
The realisation template can be extended by quantity matrix for database classes $\Omega$ and other performance constraints $\mathcal{C}$ and by business tasks and their reflections through business data units $\mathcal{B}$. Directives can be bound to one kind of platform and represent in this case a technological twist, e.g. by stating how data is layered out. They are typically however bound to several platforms in order to avoid evolution-proneness of models.

Conceptual model for prescription: The conceptual database structure model comprises the database schema, a collection of views for both support of business users and system operating, a realisation template, and the declaration of model adequacy and dependability.

Therefore, the model can be formally defined as a quintuple

$$(S, \mathcal{V}, \mathcal{T}, \mathcal{A}, \mathcal{D})$$

6 Conclusion

This paper shows that the ER schema is a central unit in a conceptual database structure model. The conceptual database structure model contains also other elements in dependence on its function in utilisation scenarios. As long as we use a global-as-design approach, the ER schema is essential and the kernel of such database structure models.

We may combine the conceptual models to description/prescription models

$$(S, \mathcal{V}, (\mathcal{R}, \Psi), \mathcal{T}, \mathcal{A}, \mathcal{D})$$

and to description/prescription models with conceptualisation

$$(S, \mathcal{V}, (\mathcal{R}, \Psi), \mathcal{M}, \mathcal{T}, \mathcal{A}, \mathcal{D})$$

The combination with communication/negotiation is more problematic since the corresponding models are based on divergent perception models that might represent the very personal viewpoint of business users in different context and work organisation.

The notion of the model for conceptual database structure models can be summarised in dependence on their utilisation scenario:

**Table 1.** Conceptual database structure models that extend the conceptual database schema in dependence on utilisation scenarios

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Model origin</th>
<th>Add-ons to the conceptual database schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communication and negotiation</td>
<td>Perception (and situation) models</td>
<td>Views representing the viewpoint variety and associated with the perception models</td>
</tr>
<tr>
<td>Conceptualisation</td>
<td>Perception and reality models</td>
<td>Associations to concepts and conceptions, semantics and meanings, namespaces</td>
</tr>
<tr>
<td>Description</td>
<td>Reality model</td>
<td>View collection, associations to origins</td>
</tr>
<tr>
<td>Prescription</td>
<td>Reality (and situation) models</td>
<td>View collection, realisation template</td>
</tr>
</tbody>
</table>
References

A Conceptual Model for Services

Bernhard Thalheim\textsuperscript{1} and Ajantha Dahanayake\textsuperscript{2}

\textsuperscript{1} Department of Computer Science, Christian Albrechts University Kiel, D-24098 Kiel
\textsuperscript{2} Prince Sultan University, Dept. of Computer Information Science, Riyadh, Kingdom of Saudi Arabia

Abstract. Models are a mainstay of every scientific and engineering discipline. Models are typically more accessible to study than the systems. Models are instruments that are effectively functioning within a scenario. The effectiveness is based on an associated set of methods and satisfies requirements of usage of the model. A typical usage of a model is explanation, informed selection, and appropriation of an opportunity. This usage is declared through information and directions for usage or more specifically through an informative model in the case of a service model.

1 Services and The Conception of a Service

Today, the service has gained recognition as the more realistic concept for dealing with complexities of cross-disciplinary systems engineering extending its validity beyond the classical information systems design and development realm [4]. In this respect the service concept combines and integrates the value created in different design contexts such as person-to-person encounters, technology enabled self-service, computational services, multi-channel, multi-device, location-based and context-aware, and smart services [13]. Therefore, the service concept reveals the intrinsic design challenges of the information required to perform a service, and emphasizes the design choices that allocate the responsibility to provide this information between the service provider and service consumer.

1.1 Some Well-Known Service Notions

The service is being defined using different abstraction models with varying applications representing multitude of definitions of the service concept [7]. The increasing interests in services have introduced service concept’s abstraction into levels such as; business services, web services, software-as-a-service (SaaS), platform-as-a-services, and infrastructure-as-a-service [2]. Service architectures are proposed as means to methodically structure systems [1, 5, 16].

There are number of service notations available in the in the literature, and research has looked into the service mainly from two perspectives, (a) from the

\textsuperscript{1} thalheim@is.informatik.uni-kiel.de
\textsuperscript{2} adahanayake@psuw.psu.edu.sa
low-level technological point of view and (b) from the higher abstract business point of view. These two categories of service descriptions have derived number of service notations. Some of those main stream service notations are:

The REA (Resource-Event-Agent) ontology [8, 11] uses as core concepts resources, economic event, and agent. The RSS (Resource-Service-Systems) model [12] is an adaptation of REA ontology stressing that REA is a conceptual model of economic exchange and uses a Service-Dominant Logic (SDL) [21]. The model of the three perspectives of services uses abstraction, restriction, and co-creation. It concentrates on the use and offering of resources [2]. The perspectives addressed by this model are: service as a means for abstraction; service as means for providing restricted access to resources; and service as a means for co-creation of value. The logics behind is the Goods Dominant Logic (GDL) model [22]. Web service description languages concentrate on Service-Orientated Architectures (SOAs) for web service domain. Software systems are decomposed into independent collaborating units [14]. Named services interact with one another through message exchanges. The seven contexts of service design [6, 9, 13] combine person-to-person encounters, technology-enhanced encounters, self-service, computational services, multi-channel, multi-device, and location-based and context-aware services description.

1.2 The Explanation, Selection, and Appropriation

Explanation, understanding and informed selection of a tool is one of the main usage scenarios for a software models. People want to solve some problems. Services provide solutions to these problems and require a context, e.g. skills of people, an infrastructure, a specific way of work, a specific background, and a specific kind of collaboration. In order to select the right service, a model of the service is used as an instrument for explanation and quick shallow understanding which service might be a good candidate, what are the strengths and weaknesses of the service under consideration, which service meets the needs, and what are the opportunities and risks while deploying such a service.

The best and simplest instrument in such usage scenario is the instruction leaflet or more generally as a specification of the information and directions on the basis of the informative model. We shall show in the sequel that this model of a service extends the cargo dimension [10] to the general notion of the informative model. Such models of a service enable people in directed, purposeful, rewarding, realistic, and trackable deployment of a service within a given usage scenario, i.e. use according to the qualities of the model [4]. After informed selection of a service, it might be used in the creation of new work order based on the assimilation of the service into the given context, i.e. appropriation of the service.

1.3 Developing a Service Model based on the W*H Frame

Systems are typically characterised by a combination of large information content with the need of different stakeholders to understand at least some system aspects. People need models to assist them in understanding the context of their
own work and the requirements on it. We concentrate in this paper on the support provided by models to understand how a system works, how it can be used or should not be used, and what would be the benefit of such a model. We illustrate this utilisation of models for services.

We develop a novel service model based on the W*H specification frame [4]. The W*H model [4] provides a high-level and conceptual reflection and reflects on the variety of aspects that separates concerns such as service as a product, service as an offer, service request, service delivery, service application, service record, service log or archive and also service exception, which allows and supports a general characterization of services by their ends, their stakeholders, their application domain, their purpose and their context.

2 The Notion of a Model

The theory of models is the body of knowledge that concerns with the fundamental nature, function, development and utilisation of models in science and engineering, e.g. in Computer Science. In its most general sense, a model is a proxy and is used to represent some system for a well-defined purpose. Changes in the structure and behaviour of a model are easier to implement, to isolate, to understand and to communicate to others. In this section we review the notion of the model that has been developed in [18–20].

2.1 Artifacts that are Models

A model is a well-formed, adequate, and dependable artifact that represents origins. Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilisation scenarios.

The model should be well-formed according to some well-formedness criterion. As an instrument or more specifically an artefact a model comes with its background, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form. A model is used in a context such as discipline, a time, an infrastructure, and an application.

Models function as an instrument in some usage scenarios and a given usage spectrum. Their function in these scenarios is a combination of functions such as explanation, optimization-variation, validation-verification-testing, reflection-optimization, exploration, hypothetical investigation, documentation-visualisation, and description-prescription functions. The model functions effectively in some of the scenarios and less effectively in others. The function determines the purpose and the objective (or goal) of the model. Functioning of models is supported by methods. Such methods support tasks such as defining, constructing, exploring, communicating, understanding, replacing, substituting, documenting, negotiating, replacing, optimizing, validating, verifying, testing, reporting, and accounting. A model is effective if it can be deployed according to its objectives.
Models have several essential properties that qualify an artifact as a model. An well-formed artifact is adequate for a collection of origins if it is analogous to the origins to be represented according to some analogy criterion, it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its purpose.

Well-formedness enables an artifact to be justified by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through formulas, by falsifiability, and by stability and plasticity.

The artifact is sufficient by its quality characterisation for internal quality, external quality and quality in use or through quality characteristics [17] such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

A well-formed artifact is called dependable if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

2.2 Artifacts as Instruments in Some Usage Scenario

Models will be used, i.e. there is some usage scenario, some reason for its use, some goal and purpose for its usage and deployment, and finally some function that the model has to play in a given usage scenario. A typical usage scenario is problem solving. We first describe a problem, then specify the requirements for its solutions, focus on a context, describe the community of practices and more specifically the skills needed for the collaborative solution of the problem, and scope on those origins that must be considered. Next we develop a model and use this model as an instrument in the problem solving process. This instrument provides a utility for the solution of the problem. The solution developed within the model setting is then used for derivation of a solution for the given problem in the origin setting.

A similar use of models is given for models of services. Service models might be used for the development of a service system. They might be used for assessment of services, for optimisation and variation of services, for validation-verification-testing, for investigation, and for documentation-visualization. In this paper we concentrate on the explanation, informed selection, and appropriation use of a service model. It must provide a high level description of the service itself. This usage is typical for a process of determining whether a service is of high utility in an application. Such usage is based on specific usage pattern or more specifically on a special model that is the usage model of an instrument as a model.

2.3 Conceptional Modelling: Modelling Enhanced by Concepts

An information systems model is typically a schematic description of a system, theory, or phenomenon of an origin that accounts for known or inferred properties of the origin and may be used for further study of characteristics of the
origin. *Conceptional modelling* \(^1\) aims to create an abstract representation of the situation under investigation, or more precisely, the way users think about it. *Conceptual models* enhance models with concepts that are commonly shared within a community or at least within the community of practice in a given usage scenario. Concepts specify our knowledge what things are there and what properties things have. Their definition can be given in a narrative informal form, in a formal way, by reference to some other definitions, etc. We may use a large variety of semantics \([15]\), e.g., lexical or ontological, logical, or reflective.

### 2.4 Adequacy and Dependability of Informative Models

Models are used in *explanation, informed selection, and appropriation* scenarios. We call such models *informative models*. Their main aim of is to inform the user according to his/her information demand and according to the profile and portfolio. The instrument steers and directs its users which are typically proactive. It supplies information that is desired or needed. Users may examine and check the content provided. Typical methods of such instruments are communication, orientation, combination, survey, and feedback methods.

Users have to get informed what is the issue that can be solved with the instrument, what are the main ingredients of the instrument and how they are used, what is the main background behind this instrument, and why they should use this instrument. They need a quick shallow understanding how simple, how meaningful, how adequate, how realistic, and how trackable is the instrument (*SMART*). They must be enabled to select the most appropriate instrument, i.e. they should know the strengths, weaknesses, opportunities, and threats of the given instrument (*SWOT*).

The SWOT and SMART evaluation is the basis for adequateness and dependability of informative models. The informative model must be analogous in structure and function to its origins. It is far simpler than the origin and thus more focussed. Its purpose is to explain the origin in such a way than a user can choose this instrument because of its properties since all demanded properties are satisfied. The selection and appropriation of an instrument by the user depends on the explanatory statement on the profile and the portfolio of the given instrument, on coherence to the typical norms and standards accepted by the community of practice, on a statement on applicability and added value of the instrument, and the relative stability of the description given. The instrument usage becomes then justified. Furthermore, the instrument must suffice the demands of such scenarios. The quality in use depends on understandability and parsimony of description, worthiness and eligibility of presented origins, and

\(^1\) The words ‘conceptual’ and ‘conceptional’ are often considered to be synonyms. The word ‘conceptual’ is linked to concepts and conceptions. ‘Conceptual’ means that a thing - e.g. an instrument or artifact - is characterised by concepts or conceptions. The word ‘conceptional’ associates a thing as being or of the nature of a notion or concept. Conceptional modelling is modelling with associations to concepts. A conceptual model incorporates concepts into the model.
the added value it has for the given utilisation scenarios. The external quality is mainly based on its required exactness and validation. The internal quality must support these qualities. The quality evaluation and the quality safeguard is an explicit statement of these qualities according to the usage scenarios, to the context, to the origins that are represented, and to community of practice.

2.5 The Cargo of a Model

The cargo of any instrument is typically a very general instrument insert like the package insert in pharmacy or an enclosed label. It describes the instrument, the main functions, the forbidden usages, the specific values of the instrument, and the context for the usage model. Following [10, 20] we describe the cargo by a description of the mission of the instrument in the usage scenarios, the determination of the instrument, an abstract declaration of the meaning of the instrument, and a narrative explanation of the identity of the instrument.

The mission of a model consists of functions (and anti-functions or forbidden ones) that the model has in different usage scenarios, the purposes of the usages of the model, and a description of the potential and of the capacity of the model. The determination contains the basic ideas, features, particularities, and the usage model of the given instrument. The meaning contains the main semantic and pragmatic statements about the model and describes the value of the instrument according to its functions in the usage scenarios, and the importance within the given settings. Each instrument has its identity, i.e. the actual or obvious identity, the communicated identity, the identity accepted in the community of practice, the ideal identity as a promise, and the desired identity in the eyes of the users of the instrument.

2.6 The Informative Model

The informative model consists of the cargo, the description of its adequacy and dependability, and the SMART and SWOT statements. It informs a potential user through bringing facts to somebody’s attention, provides these facts in an appropriate form according their information demand, guides them by steering and directing, and leads them by changing the information stage and level. Based on the informative model, the user selects the origin for usage with full informed consent or refuses to use it. It is similar to an instruction leaflet provided with instruments we use. The informative model is semantically characterized by: objectivity; functional information; official information; explanation; association to something in future; different representational media and presenters; degree of extraction from open to hidden; variety of styles such as short content description, long pertinent explanation, or long event-based description.

In the case of a service model, the informative model must state positively and in an understandable form what is the service, must describe what is the reward of a service, and must allow to reason about the rewards of the service, i.e. put the functions and purposes in a wider context (PURE). Informative models of a service are based on a presentation that is easy-to-survey and to understand,
that is given in the right formatting and form, that supports elaboration and surveying, that avoids learning efforts for their users, that provides the inner content semantics and its inner associations, that might be based on icons and pictographs, and that presents the annotation and meta-information including also ownership and usability.

We shall now explore in the sequel what are the ingredients of such informative instruments in the case of a service model.

3 Service Specifications

3.1 Scenarios and Functions of Service Specifications

To capture the scenarios and functions of service specification we introduce $W^*H$ model in Figure 1 that is a novel conceptual model for service modelling.

<table>
<thead>
<tr>
<th>Service</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concept</td>
<td>Ends</td>
</tr>
<tr>
<td></td>
<td>$Wherefore?$</td>
</tr>
<tr>
<td></td>
<td>Purpose</td>
</tr>
<tr>
<td></td>
<td>$Why?$</td>
</tr>
<tr>
<td></td>
<td>$Where to?$</td>
</tr>
<tr>
<td></td>
<td>$For When?$</td>
</tr>
<tr>
<td></td>
<td>$For Which reason?$</td>
</tr>
<tr>
<td>Content</td>
<td>Supporting means</td>
</tr>
<tr>
<td></td>
<td>$Wherewith?$</td>
</tr>
<tr>
<td></td>
<td>Application</td>
</tr>
<tr>
<td></td>
<td>Domain</td>
</tr>
<tr>
<td></td>
<td>Application are $Wherein?$</td>
</tr>
<tr>
<td></td>
<td>Application case $Wherefrom?$</td>
</tr>
<tr>
<td></td>
<td>Problem     $For What$</td>
</tr>
<tr>
<td></td>
<td>Organizational unit $Where$</td>
</tr>
<tr>
<td></td>
<td>Triggering Event $Whence$</td>
</tr>
<tr>
<td></td>
<td>IT $What$</td>
</tr>
<tr>
<td></td>
<td>How</td>
</tr>
<tr>
<td>Annotation</td>
<td>Source</td>
</tr>
<tr>
<td>Party</td>
<td>Supplier $By whom?$</td>
</tr>
<tr>
<td>Consumer</td>
<td>To whom?</td>
</tr>
<tr>
<td>Producer</td>
<td>Whichever?</td>
</tr>
<tr>
<td>Activity</td>
<td>Input $What in?$</td>
</tr>
<tr>
<td></td>
<td>Output $What out?$</td>
</tr>
<tr>
<td>Added Value</td>
<td>Surplus Value</td>
</tr>
<tr>
<td>Context</td>
<td>Systems Context $Where at?$</td>
</tr>
<tr>
<td>Story Context</td>
<td>$Where about?$</td>
</tr>
<tr>
<td>Coexistence Context</td>
<td>$Wither?$</td>
</tr>
<tr>
<td>Time Context</td>
<td>$When?$</td>
</tr>
</tbody>
</table>

Fig. 1. The $W^*H$ Specification Frame for the Conceptual Model of a Service

The $W^*H$ model in Figure 1 fulfills the conceptual definition of the service concept composing the need to serve the following purposes:
The composition of the W*H model consisting of content space, concept space, annotation space, and add value space as orthogonal dimensions that captures the fundamental elements for developing services.

It reflects number of aspects neglected in other service models, such as the handling of the service as a collection of offering, a proper annotation facility, a model to describe the service concept, and the specification of added value. It handles those requirements at the same time.

It helps capturing and organizing the discrete functions contained in (business) applications comprised of underlying business process or workflows into interoperable, (standards-based) services.

The model accommodates the services to be abstracted from implementations representing natural fundamental building blocks that can synchronize the functional requirements and IT implementations perspective.

It considers by definition that the services to be combined, evolved and/or reused quickly to meet business needs.

Finally, it represents an abstraction level independent of underlying technology.

In addition, the W*H model in 1 also serves the following purposes:

The inquiry through simple and structured questions according to the primary dimension on wherefore, whereof, wherewith, and worthiness further leading to secondary and additional questions along the concept, annotation, content, add value or surplus value space that covers usefulness, usage, and usability requirements in totality.

The powerful inquiring questions are a product of the conceptual underpinning of W*H grounded within the conceptional modelling tradition in the Concept-Content-Annotation triptych extended with the Added Value dimension and further integration and extension with the inquiry system of Hermagoras of Temnos frames.

The W*H model is comprise of 24 questions in total that cover the complete spectrum of questions addressing the service description; (W5 + W4 + W10H +W4) and H stands for how.

The models compactness helps to validate domain knowledge during solution modelling discussions with the stakeholders with high demanding work schedules.

The comprehensibility of the W*H model became the main contributor to the understanding of the domain’s services and requirements.

The model contributes as the primary input model leading to the IT-service systems projection on solution modelling.

It contribute as the primary input model leading to the IT-service systems projection on the evaluations criteria of systems functioning on its trustworthiness, flexibility to change, and efficient manageability and maintainability.

### 3.2 Dimensions of Service Specification

*The Content Dimension: Services as a Collection of Offerings.* The service defines the what, how, and who on what basis of service innovation, design, and
development, and helps mediate between customer or consumer needs and an organizations strategic intent. When extended above the generalized business and technological abstraction levels, the content of the service concept composes the need to serve the following purposes:

- Fundamental elements for developing applications;
- Organizing the discrete functions contained in (business) applications comprised of underlying business process or workflows into inter operable, (standards-based) services;
- Services abstracted from implementations representing natural fundamental building blocks that can synchronize the functional requirements and IT implementations perspective;
- Services to be combined, evolved and/or reused quickly to meet business needs; Represent an abstraction level independent of underlying technology.

The abstraction of the notion of a service system within an organizations strategic intent emphasized by those purposes given above allow us to define the content description of services as a collection of offers that are given by companies, by vendors, by people and by automatic software tools [3]. Thus the content of a service system is a collection of service offerings.

The service offering reflects the supporting means in terms of with what means the service’s content is represented in the application domain. It corresponds to identification and specification of the problem within an application area. The problem is a specific application case that resides with an organizational unit. Those problems are subject to events that produce triggers needing attention. Those triggering events have an enormous importance for service descriptions. They couple to the solution at hand that is associated with how and what of a required IT solution.

The Annotation Dimension. According to [14], annotation with respect to arbitrary ontologies implies general purpose reasoning supported by the system. Their reasoning approaches suffer from high computational complexities. As a solution for dealing with high worst-case complexities the solution recommends a small size input data. Unfortunately, it is contradicting the impressibility of ontologies and define content as complex structured macro data. It is therefore, necessary to concentrate on the conceptualisation of content for a given context considering annotations with respect to organizations intentions, motivations, profiles and tasks, thus we need at the same time sophisticated annotation facilities far beyond ontologies. Annotation thus must link the stakeholders or parties involved and activities; the sources to the content and concept.

The Concept Dimension. Conceptional modelling aims at creation of an abstract representation of the situation under investigation, or more precisely, the way users think about it. Conceptual models enhance models with concepts that are commonly shared within a community or at least between the stakeholders involved in the modelling process.
According to the general definition of concept as given in [19], Concepts specify our knowledge what things are there and what properties things have. Concepts are used in everyday life as a communication vehicle and as a reasoning chunk. Concept definition can be given in a narrative informal form, in a formal way, by reference to some other definitions etc. We may use a large variety of semantics, e.g., lexical or ontological, logical, or reflective.

Conceptualisation aims at collection of concepts that are assumed to exist in some area of interest and the relationships that hold them together. It is thus an abstract, simplified view or description of the world that we wish to represent. Conceptualisation extends the model by a number of concepts that are the basis for an understanding of the model and for the explanation of the model to the user.

The definition of the ends or purpose of the service is represented by the concept dimension. It is the curial part that governs the service’s characterization. The purpose defines in which cases a service has a usefulness, usage, and usability. They define the potential and the capability of the service.

The Added Value Dimension. The added value of a service to a business user or stakeholder is in the definition of surplus value during the service execution. It defines the context in which the service systems exists, the story line associated within the context, which systems must coexist under which context definitions prevailing to time. Surplus value defines the worthiness of the service in terms of time and labor that provide the Return of Investment (ROI).

4 Conclusion

There are many other usage models for services. This paper elaborated the explanation, informed selection, and appropriation usage model for a service. Other usage models of an instrument as a model are, for instance, optimization-variation, validation-verification-testing, understanding, extension and evolution, reflection-optimization, exploration, documentation-visualization, integration, hypothetical investigation, and description-prescription usage models. We introduced in this paper a general notion of the model and showed what makes description or specification a service to be become a model of the service.

References


Abstract. Models are one of the main instruments in Computer Science. The notion of model is however not commonly agreed due to the wide usage of models. It is challenging to find an acceptable and sufficiently general notion of model due to the large variety of known notations. Such notion should incorporate all of the different notations and at the same time should allow to derive the specific notation from the general notion. We introduce a universal parameterised notion of the model. The parameters in this notion support adaptation of the universal notion of the model to the specific notation of interest. We finally apply this notion and this adaptation to development of business process models that are specified in BPMN.

1 The Model - an Artifact and an Instrument

Classical Computer Science research considers models as artifacts\(^1\) that are constructed in certain way and prepared for their utilisation according to the purpose under consideration such as construction of systems, verification, optimization, explanation, and documentation.

Creation for a practical purpose means that the main target of model development is its application in utilisation scenarios. Models are considered to be artifacts in a stronger sense. We observe however that models are developed for their utilisation within some scenario. They are functioning in this scenario. That means models are instruments in these scenarios. The notion of an instrument\(^2\) concentrates on this utilisation of models. Models are therefore mainly instruments that are effectively functioning within a scenario. The effectiveness is based on an associated set of methods and satisfies requirements of usage of the model.

1.1 Models - The Third Dimension of Science

Models are used as perception models, experimentation models, formal models, conceptual models, mathematical models, computational models, physical

---

\(^1\) An artifact is “something that is created by humans usually for a practical purpose” or “something characteristic of or resulting from a particular human institution, period, trend, or individual” or “a product of artificial character due usually to extraneous (as human) agency” [16]. The last meaning of the notion of an artifact is not taken into consideration for models in most sciences and also in Computer Science.

\(^2\) An instrument is among others (1) a means whereby something is achieved, performed, or furthered; (2) one used by another as a means or aid or tool [16].
Models are one of the main instruments in scientific research. They are considered to be the third dimension of science [26] (Figure 1). They provide a tool for research and have an added value in research, e.g., for construction of systems, for education, for the research process itself. Their added value is implicit but can be estimated based on the capability, potential and capacity of the model. Models are common culture and practice in sciences. However, each discipline has developed its own modelling expertise and practice.

Models are often language based. Their syntax uses the namespace and the lexicography from the application domain. Semantics is often implicit. The lexicology can be inherited from the application domain and from the discipline. Models do not need the full freedom for interpretation. The interpretation is governed by the purpose of the model within the research scenario,

\[3\] The title of the book [4] has inspired this observation.
is based on disciplinary concerns (postulates, paradigms, foundations, commonsense, culture, authorities, etc.) and is restricted by disciplinary practices (concepts, conceptions, conventions, thought style and community [6], good practices, methodology, guidelines, etc.). Models combine at least two different kinds of meaning in the namespace: referential meaning establishes an interdependence between elements and the origin ('what'); functional meaning is based on the function of an element in the model ('how'). The pragmatics of a model depends on the community of practice, on the context of the research task and especially on the purpose or function of the model.

A model can be used for different purposes and various usage scenarios. Therefore, a model is typically also extended by views or viewpoints that reflect certain parts of the model and that hide details which are not necessary. This reflection is often only provided in a non-systematic or implicit way. Additionally, we need a refinement notion, methods for combination and for evaluation of models.

1.2 Scenarios of Model Utilisation

Models are used as an instrument in some utilisation scenario. At the same time, the model might be useless and not productive in other scenarios. Their function in these scenarios is a combination of functions such as explanation, optimization-variation, validation-verification-testing, reflection-optimization, exploration, hypothetical investigation, documentation-visualization, and description-prescription as a mediator between a reality and an abstract reality that developers of a system intend to build.

Traditionally, purposes or goals are considered first. The purposes and the goals are used to determine the functions of a model. This approach is centered around the purpose or goal and requires a definite understanding of the purpose and goal. Purposes and goals are often underspecified or blurry at the beginning. They become more clear after the model is being used. Compared to this approach, it is simpler to understand the application cases of a model and thus the utilisation scenarios. In this case we may derive the functions that a model has in these scenarios. Therefore, we use the approach that the functions of the model determine the purposes of the deployment of the model.

1.3 The Storyline of the Paper

The large variety of model notations (see, for instance, [13, 23, 30]) does not allow to transfer experience gained with one notation to other notations. Methods for utilisation or development are therefore mainly bound to one notation. Each subdiscipline has therefore its own understanding of modelling. It would however be beneficial to have a general notion of model that can be adapted to the specific notations of interest.

We introduce in Section 2 a universal notion of a model. This notion is based on the understanding of a model as an instrument in some utilisation scenarios. We only consider well-formed instruments since models must be intuitive and easy to understand. The model definition is based on two general
parameter sets, adequateness and dependability. Each of the parameters can be instantiated in dependence of the function that the model should have in a given utilisation scenario within the sub-discipline. This instantiation facility is based on a conception frame for the model notion.

The approach is applied to BPMN modelling in Section 3. We describe the business process modelling approach and derive the capability of this modelling technique. We can now also explicitly describe the obstacles of BPMN modelling. Furthermore, we derive the evaluation procedure for the BPMN approach in Section 4.

This approach to modelling in Computer Science can now be used as a starting point of a theory of modelling (Section 5). We start with some, often implicitly given restrictions that a model has, esp. its burden by the background and by the directives. The evaluation of models also supports a statement on not-supported utilisations, called anti-profile. Finally, the conception frame can also be used for development of question forms that support model specification.

2 The Universal Notion of the Model

There are many notions of models. Each of them covers some aspects and concentrates on some properties such as the mapping, analogy, truncation, pragmatism, amplification, distortion, idealisation, carrier, added value, and purpose properties [11, 17, 18, 21]. The main property is however the function property: The model suffices in its function in the utilisation scenarios that are requested. This property results in the following notion of the model [25, 27, 29].

2.1 The Model Notion

Models have several essential properties that qualify an instrument as a model [22, 24]:

Definition 1. An instrument is well-formed if it satisfies a well-formedness criterion.

Definition 2. A well-formed instrument is adequate for a collection of origins if (i) it is analogous to the origins to be represented according to some analogy criterion, (ii) it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and (iii) it is sufficient to satisfy its purpose.

Definition 3. Well-formedness enables an instrument to be justified: (i) by an empirical corroboration according to its objectives, supported by some argument calculus, (ii) by rational coherence and conformity explicitly stated through formulas, (iii) by falsifiability that can be given by an abductive or inductive logic, and (iv) by stability and plasticity explicitly given through formulas.
Definition 4. An instrument is sufficient by a quality characterisation for internal quality, external quality and quality in use or through quality characteristics [20] such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

Definition 5. A well-formed instrument is called dependable if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

Definition 6. An instrument is called model if it is adequate and dependable. The adequacy and dependability of an instrument is based on a judgement made by the community of practice.

Definition 7. An instrument has a background consisting of an undisputable grounding from one side (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities) and of a disputable and adjustable basis from other side (assumptions, concepts, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense).

Definition 8. A model is used in a context such as discipline, a time, an infrastructure, and an application.

The model notion can be depicted in Figure 2 based on the following conceptions:

1. **A fundament or background** with
   - the grounding, and
   - the (meta-)basis,
2. **Four governing directives** given by
   - the artifacts or better origins to be represented by the model,
   - the deployment or profile of the model such as goal, purpose or functions,
   - the community of practice (CoP) acting in different roles on certain rights through some obligations, and
   - the context of time, discipline, application and scientific school,
3. **Two pillars** which provide
   - methods for development of the model, and
   - methods for utilisation of the model,
4. **The model utilisation scenario** for the deployment of the model in the given application.

The model house in Figure 2 abstracted from its full version [24, 27] displays these different facets of the model. The house consists of a cellar (basis in Figure 2) and a fundament (grounding in Figure 2), two pillars (development resp. utilisation methods), four driving or governing forces (origins, purpose of function, community of practice, context), and finally the deployment roof (utilisation scenario). The grounding is typically implicitly assumed and not disputable. It contains paradigms, the culture in the given application area, the background, foundations and theories in the discipline, postulates, (juristic and other) restrictions, conventions, and the commonsense. The basis is the main part of the background and is typically disputable.
Definition 9. A fully-specified model is function-purpose-goal invariant if the model can be used instead of the origins in the given scenario and have the same goal, the same purpose, and the same function. A model is solution-faithful if the solution of the problem solved with the model is analogous in the world of the origins based on the analogy criterion that is used for stating adequacy.

2.2 The Conception Frame for the Model Notion

The model notion covers many different aspects. It might thus be of interest whether there is a guideline for development of models. Models are artifacts that can be specified within a W*H-frame [5] that extends the classical rhetorical frame introduced by Hermagoras of Temnos\(^4\). Models are primarily characterised by W\(^4\): wherefore (purpose), whereof (origin), wherewith (carrier, e.g. language), and worthiness ((surplus) value). The secondary characterisation dimensions are given by: (1) stakeholder: by whom, to whom, whichever; (2) additional properties of the application domain: wherein, where, for what, wherefrom, whence, what; (3) solution: how, why, whereto, when, for which reason; and (4) context: whereat, whereabout, whither, when.

A practical guideline may just

\(^4\) Quis, quid, quando, ubi, cur, quem ad modum, quibus adminiculis (Who, what, when, where, why, in what way, by what means), The Zachman frame uses a simplification of this frame.
1. start with fixation of two directives: origins to be represented and community of practice that accepts this model;
2. restrict the model utilisation scenario and the usage model to those that are really necessary and thus derive the purpose and function of the instrument;
3. define adequateness and dependability criteria of the instrument within the decision set made so far;
4. explicitly describe the background of the model, i.e. its undisputable grounding and the selected basis; and
5. explicitly specify the context for utilisation of the model.

Fig. 3. Conception frame for systematic development of a model

The model development and utilisation depends in this case from:
Judgements of some members of the CoP to deploy the instrument as a model for some origin based on an assessment (deployability, rigidity, modality, confidence) within a CoP, utilisation scenario, and within a context.
Utilisation scenarios and use spectra accepted for the instrument with functions of the instrument in utilisation scenario, roles and deployment of the instrument in those scenario, and resulting purposes and goals for the utilisation.
The instrument as such, with some appreciation
as a well-shaped instrument, on the basis
– of some criteria in dependence on intended utilisation and criteria
for:
  • what is accepted in a CoP, and
  • what is syntactically, semantically, pragmatically well-shaped,
that fits to the intended use, and
is appropriate for the utilisation scenarios and the use spectra.

The orientation also reflects our understanding of a model as an instrument.

3 BPMN Diagrams as Models

The Business Process Modeling and Notation (BPMN) language [8,14] is a
conceptual business process specification language and is standardized by the
Object Management Group (OMG). There are many different languages for
description of business stories (e.g. SiteLang), of business rules (e.g. business
use cases), and of workflows that are essentially specifications of business pro-
cesses, activities of participants, utilisation with resources, and of com-mu-
nication among the participants. Languages such as S-BPM, BPMN, and EPC
concentrate on different aspects of business processes, vary in scope and fo-
cus, use different abstraction levels, and are thus restricted in the capacity and
potential for modelling. Most of the existing languages evolved over their life-
span and extensionally added features, more features, and other features again.
BPMN is not an exception for this kind of overloading.

A business process consists of an ordered set of one or more activities (tasks)
which collectively realize a business objective or policy goal. A workflow is the
executable specification of a business process. It may describe all or some of
the five aspects of business processes [15]:

(1) control flow description for the partial order of the activities, events or
steps;
(2) organisation description with participants, their roles and plays within
the processes, their rights and obligations, their resources, and their assign-
ments;
(3) the data viewpoint description with an association to process elements
and access rights for participants;
(4) the functional description that specifies semantics, pragmatics, and beh-
aviour of each element of the workflow, e.g. the operations to be per-
formed, pre- and postconditions, priority, triggers, and time frames for the
operations;
(5) the operational assignment of programs that support all elements of the
workflow.

The entire modelling process is based on a local-as-design perspective. A
holistic or global view on a diagram collection is the task of a designer and
becomes problematic in the case of specification evolution.
BPMN 2.0 defined four different kinds of diagrams for workflow specification. We shall briefly review these diagrams in the sequel. The diagram in Figure 4 combines these different aspects. It describes the accomplishment of requirements issued by a customer.

**Fig. 4.** Fulfillment of customer demands by vendors

### 3.1 Diagrams in BPMN

**Process Diagrams.** Process diagrams (also called orchestration diagrams) describe the stepwise task flow for one agent. A task flow might reflect different roles of an agent. These roles are separated by swimlanes. Processes are either public or private. Public processes can also be abstractions of private processes that represent the detailed control and task flow for a singleton agent. Main process elements are (a) atomic or complex activities for direct representation of stepwise actions of an agent, (b) gates for exclusive, non-exclusive, event-based or parallel splitting and joining of the control flow, (c) events for the start of a workflow, for the end of a workflow path, for the complete end of the entire workflow, and an intermediate event for representation of interaction events with agents outside the workflow, and (d) control flow arrows for representation of the order of process elements. Basic activities reflect abstract, service, send, receive, user, manual, business rule, or script tasks. Complex tasks reflect an entire sub-workflow, loops, parallel or sequential multiple executions, ad-hoc workflows, transactions, or specific exception handling workflows such as compensation. Interaction reflects message exchange, timer interaction, escalation enforcement, compensations, conditional interactions, links, and signals. Interaction can be sequential or parallel multiple. Interaction events may also
be boundary events for a complex activity. All elements can be explicitly annotated by comments, by consumed data, or by produced data objects or data stores.

**Choreography Diagrams.** Choreography diagrams describe the message exchange among agents with reference to sending and receiving events, the message issue, and the graph-based representation of the partial order among these messages.

**Collaboration Diagrams.** Collaboration use choreography diagrams and process diagrams for explicit binding of senders and receivers of messages to black-box abstractions of agent workflows and abstract from message issues.

**Conversation Diagrams.** Conversation diagrams survey communication flow among agents as a birds view. They allow to derive dependences among process diagrams of agents.

### 3.2 Capability of BPMN Diagrams

BPMN modelling becomes nowadays a standard for typical business applications. Therefore, the capability of processes must be specified and well understood. It is thus necessary to know what is the ability to achieve a good model through a set of controllable and measurable features.

BPMN diagrams require a work-around for a number of conceptions such as macro-state, history, and system architecture. There are redundancies in the language itself that lead to flavour- or taste-oriented programming due to the overwhelming number of elements, construct excess and overload, e.g. groups, pool and lane, transformations, off-page connectors. The structuring becomes unclear since activities can be itself a workflow or a collection of workflows. This rather specific kind of abstraction should not be mixed with abstraction in general. Exception handling is completely confusing and only partially defined.

BPMN diagrams can represent only 8 out of 43 workflow resource pattern [10]. The data aspect is provided through properties of tasks, processes, and sub-processes. Their interrelationship is left to the developer community. It is the task of the developers to keep in mind the entire picture of the BPMN diagram collection.

BPMN uses an informal approach to semantics description what has been a matter of confusion. A formal approach to BPMN semantics can however be developed [1–3].

Furthermore, there is no conception of well-formed diagrams. Decomposition and composition is left to the developer. BPMN does not properly support the aspects (2), (4), and (5). The data aspect (3) is partially represented.

### 3.3 Deficiencies of Diagrams and Diagrammatical Reasoning

Diagrams are not universal for modelling. It is often claimed that diagrams are simple to use, are easy to interpret, have an intuitive semantics, are unique within a user community and have thus a unique pragmatics, and are thus powerful instruments. We observe however a number of obstacles that must be resolved before accepting a diagram as a model, e.g. the following ones:
Habituation versus unfamiliarity: Diagram should be familiar to their users, have a unique semantics and pragmatics without any learning effort. Readers of diagrams must be literal with them.

Ambiguity of interpretation versus well-formedness: Diagrams should not confuse by multiple interpretations (e.g. arrows), by instability and by context-dependence of form-content relations.

Incremental graphical construction: Diagrams should follow the same construction pattern as the origin and should concentrate on typicality.

Naturalness of local reasoning: Local-as-design approaches presuppose locality within the world of origins.

Unfamiliarity with non-linear behaviour: Users are mainly linearly reasoning. Non-linear reasoning should be supported in a specific form.

Additional and supplementary elements without meaning: Diagrams of use elements which do not have a unique or any meaning, e.g. colours, shapes, grid forms for lines etc.

Hidden dimensions within the diagram: Diagrams cannot reflect all aspects although there are essential ones, e.g. time.

Representation as fine and visual art: Finding a good representation is a difficult task and should be supported by a culture of modelling.

All these obstacles are observed in the case of BPMN diagrams [10].

Diagrams must be developed on the principles of visual communication, of visual cognition and of visual design [12]. The culture of diagramming is based on a clear and well-defined design, on visual features, on ordering, effect, and delivery, and on familiarity within a user community.

One of the main obstacles of diagrams is the missing abstraction. The simplest way to overcome it is the development of a model suite [19] consisting of a generic model and its refinement models where each of them is adequate and dependable. Generic models [31] reflect the best abstraction of all models within a model suite.

4 Evaluation of the BPMN Approach

BPMN is a powerful diagrammatic languages that uses more than 100 modelling elements. The same situation in the reality or the implemented system can be specified by a variety of diagrams. Since a theory of diagram equivalence is missing, [27] introduced seven evaluation methods for models:

- PURE – SMART – CLEAR evaluation for the goal-purpose-function evaluation of an instrument in a given application context, for given artifacts to be represented, for a given community of practice, and for a given profile (goal, purpose, and function) under consideration of the utilisation scenarios;
- PEST evaluation for assessment of internal, external, and quality of use;
- QUARZSAND evaluation for assessment of the model development, and
- SWOT – SCOPE evaluation for description of the potential of the model, i.e. the general properties of a given instrument or the modelling method.
Since we did not explore the directives in detail nor the adequateness and dependability of an instrument that is a candidate model, we concentrate on the last two methods in this section. The evaluation of adequacy and dependability has been developed in [28]. We concentrate here on the capacity and potential of the BPMN approach.

4.1 The Capacity of the BPMN Approach

Capacity is a strategic measure whereas the potential is a tactical one. The potential can be used to derive the added value of a utilisation of a model within a given scenario. The potential allows to reason on the significance of a model within a given context, within a given community of practice, for a given set of origins, and within the intended profile.

The capacity relates an instrument to utilisation scenarios or the usage spectra. We answer the questions whether the instrument functions well and beneficial in those scenarios, whether it is well-developed for the given goals and purposes, whether it can be properly, more focused, comfortably, simpler and intelligible applied in those scenarios instead of the origins, and whether the instrument can be adapted to changes in the utilisation. The answers to these questions determine the main content or cargo, the comprehensiveness, and the authority or general value of a model. Another important aspect is the solution-faithfulness of the instrument. The capacity is an essential element of the model cargo, especially of the main content of the model.

BPMN diagrams can be used in description, prescription, explanation, documentation, communication, negotiation, inspiration, exploration, definition, prognosis, reporting and other scenarios. We discover that communication, negotiation, and inspiration are supportable. Description, prescription, and definition can be supported if the BPMN diagrams are enhanced and a precise semantics of all BPMN elements is commonly used in all four kinds of diagrams. The adequacy and especially the analogy to the origins (i.e. storyboards or business processes) is assumed to be based on homomorphy what is rarely achieved. This homomorphy is suitable if all processes are completely and in detail specified and all variations and exceptions are consistent.

The general utility of BPMN diagrams becomes rather low if the specific background of the modelling approach is not taken into consideration. BPMN diagrams are process-oriented, based on an orthogonal separation of flow element into activities, gates, and events, differentiate actors within their roles, and support communication among actors based on message exchange. The execution semantics is based on a token interpretation of control flow. Actors are isolated in their execution if binding is not done through message exchange or implicit hidden resource conditions. Data and resource are however local. All processes are potentially executed in parallel. The local-as-design approach might be appropriate if business processes are not intertwined. The concentration on the same abstraction level restricts the applicability of BPMN modelling. Generic workflows [31] provide a solution to this limitation.
4.2 The Potential of the BPMN Approach

The potential describes the (in-)appropriateness of a modelling approach within the directives. The suitability of BPMN diagrams depends on whether the application and the context support the local-as-design approach, on whether the demands of the community of practice can be satisfied, on whether the instrument is adequate (analogous, focused, purposeful), on whether the goals can be achieved with the given instrument, on the fruitfulness of the instrument compared with other instruments, and on the threats and obstacles of utilisation of BPMN diagrams.

4.3 The SWOT Evaluation of the Potential

The SWOT analysis is a high-level method that allows to evaluate the general quality of an instrument and its general assumptions of deployment.

**Strengths.** The BPMN approach is standardised and uses a large variety of constructs. It thus allows development of detailed models. It has a high expressibility. Both intra- and inter-organisational aspects can be represented. The approach is well supported by tools.

**Weaknesses.** The large variety of competing elements is also a weakness. The complexity and integration of diagrams may cause solution-unfaithfulness. The language requires high learning efforts. Processes that are dynamic at runtime cannot be modelled. Exchange among tools is an open problem.

**Opportunities.** Most business processes can be adequately described due to the variety of elements. The standardisation provides at least a base semantics.

**Threats and Risks.** None-technical users might be unable to cope with diagrams. Work-arounds hinder comprehensibility. Vendors define their own extensions. The BPMN standard does not completely define the execution.

4.4 The SCOPE Evaluation of the Potential

The SCOPE analysis of a model embeds the model into the application context, refines the capacity evaluation of an instrument, and considers the community of practice and their specific needs.

**Situation.** BPMN diagram suites provide some kind of formalisation of business processes. Communication is specified to a certain degree. Control flow is well-represented.

**Competence.** BPMN diagrams must be combined with other models since the other four aspects (organisation, data flow, functions, operational assignment) are only partially reflected.

**Obstacles.** Typical challenges of BPMN modelling are the specification complexity, diagram coherence, exception handling, and the development of an execution semantics. There is no common agreement on well-formedness of diagrams.
Prospects. A separate BPMN diagram is easy to read and to interpret.

Expectations. The BPMN approach can be combined with local-as-design-oriented conceptual data models, storyboards, business rule specification and other modelling approaches as one kind of models within a model suite.

4.5 The Resulting Potential of the BPMN Approach

The BPMN diagram has a high potential for communication and negotiation utilisation scenario. The potential for system construction within a description-prescription scenario is however rather limited due to missing co-design support. A similar inappropriateness can be stated for explanation, prognosis, exploration, definition, and reporting scenario. The potential within a documentation scenario is rather small. The highest potential of the BPMN approach can be however observed for inspiration scenario. The process, choreography, conversation, and collaboration diagrams are an appropriate means for an implementation plan based on inspiring diagrams.

Similar to SPICE assessments [7], we may rate maturation of a model and a modelling approach to: (0) ad-hoc, (1) informal, (2) systematic and managed, (3) standardised and well-understood, and (4) optimising and adaptable, and (5) continuously improvable styles. The evaluation shows that the BPMN approach has not yet reached level (2). This observation leads us to the conclusion that PURE-SMART-CLEAR and PEST evaluations are heavily dependent from the directives for BPMN diagram modelling.

A model must be of high utility, must have a high added value, and should have a high potential. These parameters also depend on the well-formedness of the instrument. The BPMN approach can be enhanced by criteria for well-formedness for syntactical, semantical and pragmatically well-shaped diagrams [28].

5 Towards a Theory of Modelling

5.1 Models Burdened with Directives and Background

The directives and the background (see Figure 2) heavily influence the way how a model is constructed, what is taken into consideration and what not, which rigidity is applied, which basis and grounding is taken for granted, and which community of practice accepts this kind of model.

The model incorporates these influences without marking them in an explicit form. The model is laden or burdened by these decisions. Additionally, models are composed of elements that are selected, changed and adapted within a development process. Figure 5 depicts elements of this burden and this development history.

5.2 The Anti-Profile of an Instrument as a Model

We may now directly conclude that an instrument might or might not be adequate and dependable for any utilisation scenario due to its insufficiency to function in this scenario.
Definition 10. A utilization pattern of an instrument describes the form of usage of an instrument, the discipline of usage, the applications in which the instrument might be used, and the conditions for its utilisation.

Definition 11. A utilization scenario consists of a utilization pattern and a number of functions a specific instrument might play in this utilization pattern.


Definition 13. A profile of an instrument as a model consists of the goals, the purposes, and the functions of the instrument within a portfolio.

We can now roughly describe an anti-profile of a model and resulting utilisation proscriptions of an instrument as a model by answering the following questions:

- For which scenarios is the instrument useless?
In which of the following scenarios efficiency and effectiveness is not given for the instrument: description & prescription, realisation & coding, theory development, theory refinement, causality consideration, inexplicability, demonstration, prediction, explanation, mastering of complexity, understanding, or ... ?

- Are essential parameters of the origins missing? Are some of the essential parameters only represented via mediating or dependable parameters? Are there dummy or pseudo dependences among the parameters?

- What cannot be adequately represented? Is the dependability really sufficient? In which case users need a special understanding and education? Which tacit knowledge is hidden in the instrument?

- In which cases the instrument cannot be effectively used? What must a user respect and obviate before using the instrument?

- Which biases and which background are palmed off? Which assumptions, postulates, paradigms, and schools of thought are hidden and not made explicit? Models might condition conclusions.

Since models are instruments their utilisation conditions conclusions and results. Therefore, it is appropriate to describe the anti-profile of a model as well.

5.3 Questions to Answer Before Using an Instrument as a Model

The rhetoric frame and its extension to the W*H frame [5] can now be used for derivation of questions one must answer before using the model:

- What is the function of the model in which scenario? What are consequential purposes and goals? What are anti-goals and anti-purposes?

- Which origins are going to be represented? Which are not considered? Does the model contain all typical, relevant and important features of the origins under consideration and only those?

- Is the instrument adequate and dependable within the utilisation scenario? What are the parameters for adequacy and dependability? How purpose-invariance and solution-faithfulness is going to be defined?

- What kind of reasoning is supported? What not? What are the limitations? Which pitfalls should be avoided?

- Do you want to have a universal model that contains all and anything? Would it be better to use a model suite where each of the models represent some specific aspects? What about the nonessential aspects?

6 Conclusion

A general understanding of the notion of a model has been already started with development of Computer Science. Milestones are the papers and books by H. Stachowiak (1980ies and 1990ies), B. Mahr (2000ies until 2015), W. Steinmüller (1993), and R. Kaschek (since 1990ies) [9, 11, 17, 18, 21]. These notions treat models from a phenomenological point of view through properties that a model should have (e.g. as main properties: mapping or analogy, truncation, pragmatic
properties). We need however also an explicit definition of the notion of model. Such general notion has been developed in a series of papers, e.g. [22, 24, 25, 27, 29].

The model notion is universal one and based on two parameter sets for adequateness and dependability. The parameter sets seem to be complex and need a methodological support. This paper develops such a support facility based on the notion of a conception frame. The practicality of the approach is demonstrated for the workflow specification language BPMN. BPMN shares the positive treatment with most other formal or informal languages in Computer Science. The capacity and thus the restrictions or obstacles are not explicitly communicated. We see however that the evaluation, capacity, potential, and capability can be explicitly provided based on our approach.

Since the model notion is a mathematical definition, it seems to be achievable to develop a theory of modelling in the sense of a theory. In this paper, we only discuss two components of such a theory: the explicit description of the background of models and the anti-profile. The conception frame for the model definition may also be used for derivation of question forms that a modeller can use before delivering an instrument as a model to a community of practice. The development of a full theory is however a research issue for the next decades.

References


Open Problems
of Information Systems
Research and Technology

Bernhard Thalheim

Christian Albrechts University Kiel, Department of Computer Science, D-24098 Kiel, Germany;
thalheim@is.informatik.uni-kiel.de

Abstract. Computer science and technology is an area of very intensive research
and development. It is estimated that the half-life period of knowledge in this
area is about 12 to 18 months. No other branch of science has such short half-
life period. Therefore new problems have to solved. At the same time some of
the old open problems must be solved. This invited talk aims at a systematic
survey of open problems and proposes a number of solutions to their solution. We
structure these open problems in structuring problems, into size or more generally
complexity problems, functionality problems, interactivity problems, distribution
problems, and general problems.

1 Computer Science and Technology

Let us first consider the primary sources for open problems of information systems re-
search and technology. They are mainly caused by the technology itself and stream of
research in this area. They are also caused by the evolution of information systems. An-
other obstacle are the notions we use differently in different settings. Computer Science
and Technology (CS&T) is a young branch of science and technology. It took hundred if
not thousands of years for other sciences to become matured. Mathematics, for instance,
can be based on three main principles: topology, order, and algebra. Social sciences use
the principles individual, development, and society. CS&T may however also be based
on four main principles in Figure 1: structuring, evolution, collaboration and modelling.
Each of these principles has a number of aspects. Structuring is either structuring in the

![Fig. 1. The four principles of Computer Science and Technology](Fig.1.png)

large, i.e., architecture, or in the small that is based on the notion of the state. Evolution
can also be either evolution in the small which is based on rules for state transformation or evolution in the large with its temporal, integration or migration aspects aspects. Collaboration is far less well understood. Collaboration services can be build based on the 3C framework [5] that separates supporting services into communication services, coordination services, and cooperation services. It is either on interaction between social systems and computer systems or on distribution among systems. Modelling is far less well understood although more than 50 different kinds of models are known for CS&T [12]. It is however possible to develop a general notion of the model [10], of the activity of model development [8] and systematic methodology backed modelling [9].

Information, data and knowledge are often used as synonyms. The separation into data, information and knowledge is displayed in Figure 2. For the definition of notion of knowledge we refer to [11]. There are several definitions for information. Information

**Data dimension:** correct, complete, in right format, at right moment of time, in a form requested, in right size and structuring

![Diagram of data dimensions](image)

- Data for user A
- Data demanded by user B
- Gossip
- News
- Validated/verified
- Proven correctness
- Valid forever

**User dimension:** information perceivable, noticable, digestable, integratable

**Knowledge dimension:** quality, validation/verification, sustainable, grounded, on consensus

**Fig. 2.** Three dimensions of semiotics: Syntax by data, semantics by knowledge, and pragmatics by information
human beings. If we consider information systems as an element of social systems then we can use the anthroposophic form: Information as processed by humans, is carried by data that is perceived or noticed, selected and organised by its receiver, because of his subjective human interests, originating from his instincts, feelings, experience, intuition, common sense, values, beliefs, personal knowledge, or wisdom, simultaneously processed by his cognitive and mental processes, and seamlessly integrated in his recallable knowledge. The separation into data, information and knowledge is displayed in Figure 2.

Computer science and technology has four sources: mathematics, electronics, engineering and applications. The first two sources are well acknowledged. The scientist adds to the store of verified, systematised knowledge of the physical or virtual world. The engineer brings this knowledge to bear on practical problems. The software engineering triptych \([3, 1]\) consists of the application domain description, the requirements prescriptions, and finally the systems specifications. Requirements must be reasonably well understood before software can be designed, programmed, and coded. The application domain must be reasonably understood in all its dependencies, weak and sufficient properties before requirements can be expressed properly. Software engineering classically considers only the second and the third dimension. Application domain engineering has not yet got its foundations.

2 Open Problems

In the sequel we list main open problems. The suggestions, hypotheses and proposals for their solution are given in the lecture and in an extended variant of this paper. We first start with a list of problems that is now over 25 years old [6, 7].

2.1 The MFDBS List

(MFDBS1) **Satisfiability** of specification: Given a structure specification of a database, e.g., a schema with integrity constraints. Is this specification by a finite, non-empty database.

(MFDBS2) **Integration of static and operational constraints**: Information systems use both static and operational constraints. Are there systems that allow a coherent treatment of both kinds of constraints?

(MFDBS3) **Strong and soft interpretation** of constraints: Integrity constraints are often only considered in their strong logical semantics. We might however use also natural semantics and allow systematic exceptions of validity of these constraints in a database. Is there any theory that can handle this?

(MFDBS4) **Global normalisation**: Classical normalisation theory treats each type in a separate form despite the association of these types within a schema. Is there a mechanism to integrate normalisation for global handling?

\[\text{\footnotesize \ref{footnote1}}\] “Scientists look at things that are and ask ‘why’; engineers dream of things that never were and ask ‘why not’.” (Theodore von Karman) [4]
(MFDBS5) **Continuous engineering** and consistent extensions: Database systems are constantly evolving, are extended, are integrated with other systems. How we may support this kind of evolution?

(MFDBS6) **Integration of quality requirements** into specification: Software engineering knows more than 40 different characteristics of quality of a software product. The same applies to information systems. How these quality characteristics can be given in a coherent and manageable form?

(MFDBS7) **Complexity of constraint sets:** Constraint set complexity has so far mainly considered only for simple classes of constraints. Develop an approach for complexity characterisation of constraint sets!

(MFDBS8) **Enforcement of integrity constraint sets:** Constraints are declared within a formal language. They must be enforced within a database system in both declarative and procedural way. Develop a formal theory of enforcement of integrity constraints that allows to enforce constraints either procedurally or at the interface level!

(MFDBS9) **Implication problems** on the level of kinds of constraints: The logical handling of integrity constraints is still based on a strict separation of these constraints according to their kind. Practical applications are however using functional, inclusion, join, exclusion, cardinality etc. constraints all together. Develop an approach that allows to manage constraints without a separation into kinds of constraints!

(MFDBS10) **Treatment of semantics by views:** Database technology uses still a local-as-view approach and specifies views on top of the logical or conceptual schemata. Applications use however views and only in very rare cases the global schema. Develop an approach for management of semantics at the view level!

(MFDBS11) **Distributed integrity management:** Is there any mechanism to manage integrity constraints in distributed environments?

(MFDBS12) **Integration of vertical, horizontal and deductive normalisation:** Classics considers only vertical normalisation. How to incorporate horizontal and deductive normalisation in this management?

(MFDBS13) **Treatment of incomplete specifications:** Specifications are normally incomplete. How to handle this incompleteness?

### 2.2 Open Problems: Technology

(TT1) **Partial identifiability:** Objects can be partially depending on the profile and portfolio of their utilisation. Develop a treatment of this identification!

(TT2) **Query optimisation** considers so far relational algebra. Develop an extension that allows aggregation, grouping, ordering!

(TT3) **Inheritance** is defined at design, code, or decision layers. Develop a layer-independent approach to inheritance.

(TT4) **Maintenance optimisation:** Systems provide their own mechanism for maintenance, e.g. time/mode/strictness of integrity maintenance. Develop a general approach to maintenance optimisation!

(TT5) **View towers:** Typical information system applications based on the local-as-view approach use towers of views. Develop a technology for view tower handling, e.g., with partial local enforcement!
(TT6) **Component systems** support industrial production-line development of information systems. Their technology needs sophisticated collaboration support.

(TT7) **Quality management for distributed information systems:** Information systems are often not entirely correct or complete or up-to-date. We need however a supporting quality management for these systems.

### 2.3 Open Problems: Co-Design

Modern information systems development is based on development of structuring, on functionality development, and on development of interaction features. Additionally, systems are distributed. Therefore, co-design and co-evolution of systems becomes a major feature.

Co-design is still under development. The current state of the art is challenging. Without conceptual models we face late specification, inflexibility, and un-maintainability of systems. Extension, change management and integration become a nightmare.

(C1) **Coherence of models:** Co-design can be based a separation of concern and on representation by various models. A model suite must however be based on coherence conditions and associations.

(C2) **Compiler-based transformation of conceptual schemata:** Conceptual schemata are often directly mapped by an interpreter. The logical schema is later optimised. We may however develop a compiler approach beyond the interpretation or rule-based approach.

(C3) **Semantics treatment:** Classical semantics of databases and information systems uses the strict approach of mathematical logics. It is often inappropriate. Therefore, we need a flexible treatment of semantics, e.g., depending on the kind of constraints or sets of constraints.

(C4) **Global-as-view schemata:** The classical information system architecture is based on a local-as-view approach. Develop new approaches opposite to classical local-as-view 2/3-layer architectures!

(C5) **Object-relational design** starts with an OR-schema (e.g., HERM-schema) and ending with a object-relational schema of modern DBMS. Develop novel design methods for OR logical schemata.

(C6) **Content services:** Information services are becoming one of the major technologies in the web. They are based on content delivery to different users and content extraction from various resources. Develop a faithful content service on top of database systems.

(C7) **Faithful mapping from and to UML:** UML is becoming a communication language for systems development. It does however not provide a holistic semantics for its diagrams. Brute-force interpretation of diagrams is the current state of art. Develop a faithful mapping that allows to use collections of diagrams of different kinds.

### 2.4 Open Problems for Structuring

(S1) **Conceptual modelling in the large** is partially as extension of conceptual modelling in the small. Develop additional solutions for modelling in the large.
List semantics for (extended) entity-relationship modelling: ER models are typically interpreted based on set semantics. Sometimes ER schemata are also mapped to XML schemata. We need however an approach that supports mapping to list-based languages.

Centre-periphery integration into schemata: Classical information system modelling treat the schema as a holistic solution. Schemata have however an inner structure. Especially they have central or kernel types and peripheric types. This separation is based on overlay techniques. Develop a technique for centre-periphery separation within a schema.

Level of locality and globality: Beside global as view and local as view models we might also look for compromises between local-centric or global-centric schemata. Develop structuring with the best globality.

Null marker logics: Null markers (inappropriately called values) carry a variety of different application semantics. Develop techniques for management of these markers beyond the existence, non-applicability and unknown.

Pattern of structures: Information system development can be based on experience and skills of developers. They use general solutions and pattern for their work. Develop pattern beyond the small patterns of Blaha.

Null marker logics: Null markers (inappropriately called values) carry a variety of different application semantics. Develop techniques for management of these markers beyond the existence, non-applicability and unknown.

Pattern of structures: Information system development can be based on experience and skills of developers. They use general solutions and pattern for their work. Develop pattern beyond the small patterns of Blaha.

Redundant schemata: Schemata need redundant types. Develop an approach for explicit maintenance of redundancy of types and data in information systems.

Open Problems: Constraints

Complexity: The complexity of constraint sets is only known for some basis kinds of constraints. Develop a complexity theory for real life constraint sets! Develop an approach to average complexity of such constraint sets!

Incomplete constraint sets: Normalisation requires complete constraint knowledge. Develop an approach for incomplete knowledge.

Denormalisation: Normalisation is lucky break. Typical applications need however well-settled denormalisation as well. Develop a definition, treatment, algorithms for denormalisation.

Global normalisation: Develop an approach to global schema normalisation beyond classical local vertical normalisation.

Partial axiomatisation: Incomplete constraint sets are the normal case for specification. Therefore we need an approach for deductive systems that provide a partial axiomatisation.

Graphical reasoning has shown to be useful for handling, reasoning and management of functional and multivalued dependencies. Develop graphical reasoning systems for other constraint sets.

Real-life constraint sets: Constraints cannot be handled only within its specific kind. We need techniques for constraint handling outside the orientation to kinds.

Open Problems: Functionality

(e)ER-SQL: SQL is becoming the main database language and is currently going to be combined with NoSQL features. (e)ER-SQL can be graphically developed based on VisualSQL. Develop an integration of VisualSQL with NoSQL features!
(F2) **Holistic functionality description:** Functionality specification is often using different languages for the application domain, for business processes and for conceptual functionality specification. We need however a holistic specification technique.

(F3) **Dynamic semantics** is still a step-child of integrity constraint handling. Develop an approach to dynamic semantics beyond transition rules that reflect static semantics.

(F4) **Robust workflows:** Workflows often use exception handling and try to reflect any potential case. This technique is infeasible for most processes. Instead develop an engineering approach that handles errors and failures without requesting a complete specification.

(F5) **Flexible workflows:** Workflows typically specify a hard-coded control flow. Life is however more flexible. Develop techniques for controllable deviations from flow and coherent finalisation.

(F6) **Information systems functions:** Information systems are often entirely based on database techniques. Users need however also functions on their own beyond retrieval and state change operations.

(F7) **Generic views:** View tower development may result in hundreds of almost unmanageable views. Develop a view derivation technique similar to generic functions.

2.7 Open Problems: Algebra

(TA1) **Transaction semantics for higher SQL:** SQL:1999, SQL:2003, and SQL: 2007 provide extended and sophisticated techniques for transaction handling. We need a holistic management for such transactions.

(TA2) **Spatial operations** have been for more advanced spatial data structures. We need however an algebra that allows to compute also spatial data.

(TA3) **Program transformation:** Database programs can be specified at the conceptual level. We lack however program transformation techniques.

(TA4) **Greatest consistent specialisation:** The GCS approach allows to derive specialisations of operations that internally maintain integrity constraints. Develop GCS techniques beyond functional and inclusion constraints!

(TA5) **Trigger assembly creation:** Triggers are currently linear static programs. They need very sophisticated development techniques and a deep understanding of their impact. Develop techniques for trigger assemblies.

(TA6) **Transformation of expressions:** Queries can be simplified if we know integrity constraints. We need a systematic transformation of expressions in the presence of IC.

(TA7) **Extension of structural recursion:** Structural recursion is the main definition technique for algebraic operations. We need however also structural recursion for holistic expressions.

2.8 Open Problems: Distribution

(D1) **Partial consistency of databases:** Distributed databases and information systems follow a variety of consistency paradigms. We need however also techniques that support collaboration of systems based on specific contracts and protocols.
(D2) **Recharging of partner databases:** Databases may collaborate based on pattern such as publish-subscribe. Develop a data integration technique depending on subscription mode of partner databases.

(D3) **Coordination:** The 3C approach to collaboration uses coordination as a central conception. Develop techniques for coordination beyond contracts.

(D4) **General model of services:** One of the most overused notions in CS&T is the service as specific software with support and with compliance. Develop a general model for services.

(D5) **Exchange frames:** The 3C framework uses techniques of communication similar to protocol engineering. It depends on the chosen architecture. Exchange frames are typical supporting means. They are given in a very technical way and thus depend on the underlying firmware. Develop a general technique for exchange frames.

(D6) **Component database systems:** with specific coupling facilities, collaboration contracts

(D7) **Pattern of distributed systems:** Pattern have been developed for specific integration in software engineering research. Information systems are more specific. Therefore, we need a specialisation of these pattern to information and database systems and specific pattern for these systems.

2.9 **Open Problems: Interactivity**

Interactivity is nowadays the main concern for web information systems. Their technology, their specification and their theory are not yet systematised. They will however have the same fate as classical information systems: they will stay with us longer than it has been expected when they have been developed. At the same time they face the same problems as we have already observed for human-computer interfaces.

(I1) **Edutainment stories:** One main kind of web information systems are edutainment systems (often called e-learning systems). Develop techniques and solutions for edutainment beyond classical blended learning and towards collaboration and true interaction.

(I2) **New stories:** In the app age we face a large variety of solutions based on small components. Their integration and coherent deployment is still a challenge to the user. It seems that they are partially arbitrarily combinable and thus only integratable by a human user. There are however main deployment paths based on mini-stories with data coherence. Develop a technology for such systematic treatment of new stories.

(I3) **Screenography** is a generalisation of scenography and dramaturgy. It supports systematic development of screens and allows an adaptation to the user. Develop an approach integration of screenography into conceptual modelling.

(I4) **Life case bakery:** System development is still based on packages that provide a fully fledged solution to a collection of application problems. Real life is however based on life event and life situations. Life cases may reflect both. They can be mapped to business use cases and business stories which are the basis for requirements prescription. We need techniques for continuous adaptation to the specific life event or life situation.
(15) $I^*$-generalisation: The $I^*$ model allows to model the intentions, desires, beliefs and goals of users. Goals can also be soft goals. These specification techniques for (Soft)Goals $\cup$ Tasks $\cup$ Actors $\cup$ Resources can be generalised to the story spaces, obligations and life cases.

(16) Privacy of users: Privacy of users becomes a major bottleneck of the 21st century. We currently lack techniques privacy profile, controlled opening of shared data, flexible protection etc.

(17) Workspace integration of users: Users have their own systems with their own workrooms, workspace and libraries. These systems vary a lot and cannot be generalised to some holistic system environment. Instead we need techniques for flexible integration of user workspaces into current information systems.

References

Models, To Model, and Modelling
Towards a Theory of Models, especially Conceptual Models and Modelling
Second Collection of Recent Papers (2015-2017)
Bernhard Thalheim
Christian-Albrechts-University Kiel, Department of Computer Science, 24098 Kiel, Germany
thalheim@is.informatik.uni-kiel.de

A model is a well-formed, adequate, and dependable instrument that represents origins.
Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice
within some context and correspond to the functions that a model fulfills in utilisation scenarios.
As an instrument or more specifically an artifact a model comes with its background, e.g. paradigms, assumptions,
postulates, language, thought community, etc. The background its often given only in an implicit form. The background
is often implicit and hidden.
A well-formed instrument is adequate for a collection of origins if it is analogous to the origins to be represented
according to some analogy criterion, it is more focused (e.g. simpler, truncated, more abstract or reduced) than the
origins being modelled, and it sufficiently satisfies its purpose. Well-formedness enables an instrument to be justified
by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through
conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of
origins. The instrument is sufficient by its quality characterisation for internal quality, external quality and quality in use
or through quality characteristics such as correctness, generality, usefulness, comprehensibility, parsimony, robustness,
novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and
restrictions). A well-formed instrument is called dependable if it is sufficient and is justified for some of the justification
properties and some of the sufficiency characteristics.

Content of This Second Collection
Compare with [DT11,TN15,Tha11a,Tha12b].
See also previous work on model suites , model composition [MST09a,ST10,Tha10a].
15. B. Thalheim and M. Tropmann-Frick. Models and their capability [TTF16b] . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 19
Compare with the BPMN foundation work in the BPMN collection, especially with [BST09,KST09,TT13],
[TTFZ14,TFTL+ 14].
16. B. Thalheim and M. Tropmann-Frick. Enhancing entity-relationship schemata for conceptual database structure
models [TTF15] . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 38
Compare with the foundational book [Tha00,Tha09c,Tha09a,Tha09b] and my previous work on entity-relationship
approach.
B Thalheim. Theories in business and information systems engineering [BFA+ 16] . . . . . . . . . . . . . . . . . . . . . . . . . . . 46
Compare with our research on semantics and logical foundation.
18. B. Thalheim and A. Dahanayake. A conceptual model for services [TD15] . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 75
See also [DT13a,DT10b,MSTW09a,DT12,ADT12a,ADT12b,DT15,TD16], [MSTQ09,MSTW09b,MSTW11],
[MSTW12].
19. B. Thalheim and M. Tropmann-Frick. Wherefore models are used and accepted? The model functions as a quality
instrument in utilisation scenarios [TTF16c] . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 86
This paper continues research quality [JT10], architecture [JT11,NT14,TELT14], and privacy [AFT09].


See our approaches to development of methodologies for modelling, especially database modelling and the codesign framework in the 90ies and 00s.


22. B. Thalheim. Normal models and their modelling matrix [Tha] ................................................... 131

See the previous work on data mining design [BT12,JP10,ST12].

See also our contributions to design science foundations of conceptual modelling [DT10b,DT10a,DT11,DT13b].

See our previous work on the entity-relationship approach to modelling, e.g. [ET11,That00,ST09b,ST15], on semantics [ST13,That11b], on programming, on the theory of databases and information systems and on the technology of information systems at http://dblp.uni-trier.de/pers/hd/t/Thalheim:Bernhard
or https://www.researchgate.net/profile/Bernhard_Thalheim/publications or
https://scholar.google.com/citations?user=lK3h9gAAAAJ or
http://independent.academia.edu/BernhardThalheim or ...

The First Collection

available through research gate and academia

Table of Content of the First Collection

Remark: page numbers from the first collection in the sequel.

Preliminary version: Lecture Notes in Computer Science 5833, [That9d]


Preliminary version: Lecture Notes in Computer Science 6860, [That11c]


6. The Definition of the (Conceptual) Model. EJC’13, [That13a,That14] ........................................... 135-148

7. Das Modell des Modelles. EWE’15 [That15] ................................................................. 149-151

8. Wissenschaft und Kunst der Modellierung: Modelle, Modellieren, Modellierung.
The main book on the Kiel approach to models: [TN15].


10. A Conceptual Model for Services. CMS’15@ER’15 [TD15] .................................................... 164-174


References


Conceptual Model
The Notion of a Model in Conceptual Modeling

Bernhard Thalheim
Christian-Albrechts University Kiel, http://www.is.informatik.uni-kiel.de/~thalheim/models.htm

SYNONYMS
Modeling, Model

DEFINITION
A model is a well-formed, adequate and dependable artefact that represents other artefacts based on criteria of adequacy and dependability commonly accepted by its community of practice within some context. A conceptual model incorporates concepts into the model. A conceptual database model is a conceptual model that represents the structure and the integrity constraints of a database within a given database system environment.

A model has - as an artefact - its background with a undisputable grounding of the sub-discipline and with a basis consisting of chosen elements from the sub-discipline. A model is functioning if it is combined with utilisation/deployment methods. A functioning model is effective if it can be successfully deployed according to its deployment scenarios and its portfolio. They thus function in the utilisation scenario and use spectrum.

MAIN TEXT
Conceptual modeling widely uses models for construction of (database or information) systems. It is a widely applied practice and has led to a large body of knowledge on constructs that might be used for modelling and on methods that might be useful for modelling.

Models are artifacts that are well-formed, adequate and dependable within a given context for some community of practice. They satisfy a purpose (or goal or function in some utilisation scenario). The profile of an artifact is based on the goal or purpose or function of the artifact. Models represent other artifacts or origins. For instance, they image or describe some reality and serve as a prescription for development of a (database) system.

Conceptual models are models that incorporate concepts into the model. Concepts are used as semantical units for classification. Model elements are associated with the concept’s names. A concept is also typically given through an embedding into the application domain and into the knowledge space.

Models function as an instrument in some utilisation scenario. The main function of a conceptual database model is the description-prescription function. In this case, the instrument is used as a mediator between a reality and an augmented reality that developers of a database system intend to build. The application of a model in a utilisation scenario is initiated by a goal or purpose that is agreed within some community of practice in some context. Other functions of a model despite the description-prescription function are the explanation, the optimisation-variation, the validation-verification-testing, the reflection-optimisation, the explorative, the hypothetical, and the documentation-visualisation functions.

An artefact is well-formed if it satisfies a well-formedness criterion. If the artefact is devoted to its profile then the artefact is called purposeful. A well-formed artefact is adequate for a collection of artifacts if it is analogous to the artifacts to be represented according to some analogy criterion within the analogy threshold, it is more focused (e.g. simpler, truncated, more abstract or reduced) than the artifacts within the given focus for some focus criterion, and it is purposeful for the given profile.

An artifact is justified, i.e. (i) by an empirical corroboration (according to purpose of its use, background, etc.) for the representation of the artifacts that is supported by some argument calculus, (ii) by rational coherence and conformity explicitly stated through formulas, (iii) by falsifiability that can be given by an abductive and/or inductive logical system, and (iv) by stability and plasticity (depending on the scope, grounding, basis, context and quality) explicitly given through formulas. The artifact is sufficient by its quality characterisation for internal quality, external quality and quality in use or through quality characteristics such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc.
Justification and sufficiency characterise the signification of an artifact for deployment, reliability and degree of precision efficiency for satisfying the deployment necessities, and extent of coverage depending on deployment. It is typically combined with some assurance evaluation.

A well-formed artefact is dependable for some of the justification properties and some of the sufficiency characteristics if the quality criteria are satisfied based on the assurance evaluation, and it is justified by a justification and based on the assurance evaluation.

A model is a well-formed, adequate and dependable artifact that represents other artifacts based on criteria of adequacy and dependability commonly accepted by its community of practice within some context.

Any artefact can be used as a model. It faithfully represents other artifacts and must provide facilities or methods for its use. An artefact is implicitly based on its background consisting of (I) an undisputable and well-accepted grounding from one side, i.e. paradigms, postulates, restrictions, theories, culture, foundations, conventions, commonsense, basement, authorities, and (II) a basis from other side, i.e. concepts, conceptions, assumptions, foundations, language as carrier, routine, school of thought, thought community, pattern, methodology, good practices, guidelines, and the cargo. The basis is negotiable.

The model and the artifact are functional if there are methods for utilisation of the artifact in dependence on the profile of the artifact. Artifacts are used for application cases that are supported by the task portfolio which an artifact might serve. Typical tasks include defining, constructing, exploring, communicating, understanding, replacing, substituting, documenting, negotiating, replacing, reporting, and accounting. We call an artifact and a model effective if it can be deployed according to its portfolio.

Models satisfy typically properties:

1. Mapping property: each model has an origin and is based on a mapping from the origin to the artifact.
2. Analogy property: the model is analogous to the origins based on some analogy criterion.
3. Truncation or reduction property: the model lacks some of the ascriptions made to the origin and thus functions as an Aristolean model by abstraction by disregarding the irrelevant.
4. Pragmatic property: the model use is only justified for particular model users, the tools of investigation, and the period of time.
5. Amplification property: models use specific extensions which are not observed in the original.
6. Distortion property: models are developed for improving the physical world or for inclusion of visions of better reality, e.g. for construction via transformation or in Galilean models.
7. Idealisation property: modelling abstracts from reality by scoping the model to the ideal state of affairs.
8. Carrier property: models use languages and are thus restricted by the expressive capacity of these languages.
9. Added value property: models provide a value or benefit based on their utility, capability and quality characteristics.
10. Purpose property: models and conceptual models are governed by the purpose. The model preserves the purpose.

CROSS REFERENCE

I. Data Model
   a. Semantic data model
   b. Conceptual Modeling
   c. Entity-relationship model
   d. Conceptual Data Model

II. Database design
   a. Conceptual schema design

REFERENCES


Model Capsules
for Research and Engineering Networks

Bernhard Thalheim and Marina Tropmann-Frick

Department of Computer Science, Christian-Albrechts-University Kiel,
Olshausenstrasse 40, 24098 Kiel, Germany
thalheim@is.informatik.uni-kiel.de

Abstract. Multi-model utilisation is a common practice in many sciences, e.g. computer science. Coherence and co-evolution of models is however still an open problem. Multi-model approaches suffer however from the impedance mismatch due to differences in modelling languages. The collaboration approach is based on preservation of local models and on explicit association of derived sub-models. Each discipline has developed its specific know-how in modelling and model deployment. Models evolve in dependence on the progress of the research work. If a model or one of its sub-models has been exchanged with a team member then this evolution must also be applied to models of the partner if those sub-models are used elsewhere.

We develop a novel approach to multi-model development and utilisation, to common use and utilisation of models and modelling experience, to systematic assessment of models and systematic extraction of the potential and capacity of models for a research community, and to the co-evolution of model networks.

1 Introduction

1.1 Complex Problems are Solved in Interdisciplinary Communities

Consider two typical research situations and problems that can be observed for interdisciplinary research in interacting communities of researchers.

(1) What are the causes for an inflammatory disease, especially for those triggered by dysfunction at boundary surfaces? Why is it a phenomenon of civilisation? How are cells and tissues infected? What kind of patient-specific treatment can be developed? How can life with a disease be improved and under what circumstances? What societal changes are required to move towards preventive medicine? Many branches of biology, medical science, economy, and social sciences participate in the research team (e.g. the Cluster of Excellence “Inflammation@Interfaces” at CAU Kiel). The collaboration relies on models that are exchanged within the team and that are the basis for a common understanding. The use of models is different. Such teams typically span over all four facets of scientific methods. Models are used in the way how empirical sciences use them, e.g. for exploration, experimentation, interpretation, and hypothesis exploration. At the same time models are used in the setting of theory-oriented sciences for explanations, for exploration, for illustration, for proofs and for concept surveys. In computational science models are used for instance for simulation, for emulation of
complex processes, for refinement of a general model by data, and for prognosis. Data sciences use models for detection of pattern, for mining of relations, and for generation of hypotheses. Models are the main exchange instrument for scientists in such teams.

(2) How climate is going to change in the future? How much will this change affect daily life? How should society and politics respond today? In order to answer such questions, teams with different backgrounds and from different sciences must be brought together. Teams must have in-depth expertise in their specific area, a common understanding, and a culture of collaboration. Each discipline and team member uses a specific background, a specific way of working, a specific language and a specific manage data and information. Team members need to exchange their insights and knowledge through models if they are to be easily understood and integrated in a multi-disciplinary manner. Reliable judgements would have to be made for example for climate change forecasting. To this date, this collaboration is not satisfactorily supported; resolving this issue will be a major research breakthrough.

The same situation can also be observed in Computer Engineering. Large systems typically consist of several components. They are developed in teams where a team member solves a certain development task with a specific scope and with an appropriate model. For instance, UML proposes several dozens of diagram languages for system development, e.g. use case, class, object, activity, package, interaction, sequence, time diagrams. Models developed vary in their scopes, aspects and facets they represent and their abstraction. Multi-modelling [3, 11, 20, 23, 24] is a culture in computer science. Maintenance of coherence, co-evolution, and consistency among models has become a bottleneck in development.

1.2 Multi-Modelling is the State-of-the-Art in Research and Engineering

Disciplines often use a combination of empirical research that mainly describes natural phenomena, of theory-oriented research that develops concept worlds, of computational research that simulates complex phenomena and of data exploration research that unifies theory, experiment, and simulation [10]. All these research methods use models as one of their main instruments. Typically, a suite or ensemble of models is simultaneously used due to the complexity of the real world, due to orientation on some of the aspects and facets, due to the abstraction level that fits best to the investigation goal, and due to the supporting instruments such as mathematics and visualisation.

Most disciplines integrate a variety of models or a society of models, e.g. [2, 14]. Models used in computer science are mainly at the same level of abstraction. It is already well-known for threescore years that they form a model ensemble (e.g. [8, 21]) or horizontal model suite (e.g. [3, 26]).

One of the main obstacles beside coherence of models is co-evolution of models within a model suite. However, this can be supported by strict or eager binding with some toleration of deviation. Coherence can be based on collaboration modi such as master-slave or handshake protocols. It is however an unsolved problem how shared elements can be managed within a model suite. At present, models are in some kind of coopetition (cooperation and competition) within a model suite. Often different languages, different backgrounds and different modelling styles are used and are not harmonised.
1.3 Overview of the Approach

In this paper we tackle the collaboration challenge by developing a flexible system to manage locally and to exchange globally models for collaboration in networks. In this case, models become thus a crosscutting concern to reflect competence for interdisciplinary collaboration and for interactive research on complex society issues that cannot be solved within a singleton discipline.

We remind in Section 2 a novel notion of the model. This notion generalises the notions of models used in archeology, arts, biology, chemistry, computer science, economics, electrotechnics, environmental sciences, farming and agriculture, geosciences, historical sciences, humanities, languages and semiotics, mathematics, medicine, ocean sciences, pedagogical science, philosophy, physics, political sciences, sociology, and sport science.

Next we discuss in Section 3 model-based collaboration in research and development. This research can be supported by model suites. They establish coherence maintenance among models. The main novel contributions of the paper are the introduction of the notion of the model capsule in Section 4 and the proof of concept in Section 5.

2 The Notion of the Model

Disciplines have developed a different understanding of the notion of a model, of the function of models in scientific research and of the purpose of the model. Many different notions are used, e.g. [4, 12, 18]. There is however not yet a general notion of a model. Our definition of a model [32] summarises the bottom-up approach to models and modelling developed at CAU Kiel.

Models are often language based. Their syntax uses the namespace and the lexicography from the application domain. Semantics is often implicit. The lexicology can be inherited from the application domain and from the discipline. Models do not need the full freedom for interpretation. The interpretation is governed by the purpose of the model within the research scenario, is based on disciplinary concerns (postulates, paradigms, foundations, commonsense, culture, authorities, etc.) and is restricted by disciplinary practices (concepts, conceptions, conventions, thought style and community [5], good practices, methodology, guidelines, etc.). Models combine at least two different kinds of meaning in the namespace: referential meaning establishes an interdependence between elements and the origin (‘what’); functional meaning is based on the function of an element in the model (‘how’). The pragmatics of a model depends on the community of practice, on the context of the research task and especially on the purpose or function of the model.

2.1 A Model is a Well-Formed, Adequate and Dependable Instrument

A model is a well-formed, adequate and dependable instrument that represents origins.

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilisation scenarios.
The model should be well-formed according to some well-formedness criterion. As an instrument or more specifically an artifact a model comes with its background, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background is often given only in an implicit form.

A well-formed instrument is adequate for a collection of origins if it is analogous to the origins to be represented according to some analogy criterion, it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its purpose.

Well-formedness enables an instrument to be justified by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through formulas, by falsifiability, and by stability and plasticity.

The instrument is sufficient by its quality characterisation for internal quality, external quality and quality in use or through quality characteristics [31] such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

A well-formed instrument is called dependable if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

The model has a profile (goal or purpose or function), represents artifacts and is used for some deployment scenario. As an instrument, a model has its own background (e.g. foundation (paradigms, postulates, theories, disciplinary culture, etc.) and basis (concepts, language, assumptions, practice, etc.)). It should be well-defined or well-formed. Adequacy is based on satisfaction of the purpose, analogy to the artifacts it represents and the focus under which the model is used. Dependability is based on a justification for its usage as a model and on a quality certificate. Models can be evaluated by one of the evaluation frameworks. A model is functional if methods for its development and for its deployment are given. A model is effective if it can be deployed according to its portfolio, i.e. according to the tasks assigned to the model. Deployment is often using some deployment model, e.g. for explanation, exploration, construction, description and prescription.

A model can be used for different purposes and various usage scenarios. Therefore, a model is typically also extended by views or viewpoints that reflect certain parts of the model and that hide details which are not necessary. This reflection is often only provided in a non-systematic or implicit way. Additionally, we need a refinement notion, methods for combination and for evaluation of models.
2.2 Models as a Means in Research and Engineering Networks

A common understanding of the nature of models, of the methods and techniques that are used for model development and model deployment and of systematic approaches to modelling enables also model-based collaboration in networks.

Models are built and modelling is performed in a similar form, with similar background and theories and within similar investigation scenarios despite the variety of models, the variety of purposes, the complexity, the range from micro to macro, and the variety of solutions. Each discipline has been developing also specific solutions to modelling and model deployment. These solutions may also be used for other disciplines, may be combined with their solutions, or may replace their solutions.

Our notion of the model has been validated and verified against the model notions of many disciplines. The validation [33] brought an insight into the specific understanding of adequacy, dependability, functioning, and effectiveness used in each of these disciplines. The validation has also resulted in an understanding of the added value of the model within the discipline, in an evaluation of the model maturity, in detection of features which are missing and should be added to the model or which can be deleted from the model, and in restrictions to model deployment which must be observed.

In Section 4 the notion of the model is generalised in order to cope with the requirements for model-based collaboration.

2.3 Local-As-Design for Disciplinary Models

Disciplines have however also their own foundation, their own background, their own culture and their own way of model use. Therefore, it is infeasible to develop a holistic model for everybody in a research team. Models should remain in their local setting and should not be integrated into general global model that is commonly agreed and used. For this reason, we prefer a local approach. A model remains within its local environment. It is however enhanced in such a way that it can be used in a collaboration and thus support exchange of ideas and results. This local approach is similar to the global-as-view integration approach used for integration of database systems. The model enhancement needs however a generalised-global-as-view approach. The collaboration environment thus supports a peer-to-peer exchange of exchange sub-models.

3 Model-Based Collaboration in Interacting Communities

3.1 Models — The “Intergalactic” Communication Instrument

Collaboration on the basis of models preserves local models which have sub-models for collaboration activities thus providing an explicit association of derived sub-models. Models vary in their abstraction, their foci and scales, their scopes, their aspects and their purposes. They are deployed in different scenarios and are backed by heterogeneous data with different granularity and at different levels of abstraction. A model-based collaboration cannot be based on an exchange of models as they are. Models must be fitted to the partner. We use typically parts or abstractions of a model for exchange. This model transformation is not yet performed in a systematic manner. We
might however develop an algebra for such model transformations. In this case we can generate derivatives or exchange sub-models of a model.

If a derivative of a model is used for exchange then the model of the partner can incorporate the derivative. The derivative is typically transformed to the model. It is then integrated into the model that is under revision in such collaboration activities. The derivative is associated to a sub-model of the new model. This association can be the basis for future communication. Modelling itself becomes now teamwork.

Models are simultaneously used in interdisciplinary teams for different interleaved purposes. For instance, a conceptual model of an information system is used for construction and inspiration in an implementation phase, for planning and resource allocation, for verification against the requirements model, for optimisation of the structure, for prognosis of behaviour of the system that is under construction, for explanation and understanding its components, and as the basis for system integration. Each of these functions can be used by different stakeholders at the same time. Typically, only some of model elements are of interest to different team members. These members should be better supported by specific views defined on top of the model. These views should be defined in dependence on the viewpoints that are requested by the partner. If the model is changed then these views must also be changed and the change must be communicated in an appropriate form. Model views are therefore exported to partners.

A change in one model may also result in a change to models of collaborators. The changes should be integratable into the model. The result of integration by a partner should be communicated within a research team. Model views that are derived from one model are imported into model views of another model. Since models might use different languages the model view that is exported by one model must be transformed before integration into another model.

### 3.2 Model Suites

Model suites are an extension of model ensembles [22] used for distributed or collaborating databases [25].

A model suite [3, 26] consists

- of set of models \( \{ M_1, \ldots, M_n \} \),
- of an association or collaboration schema among the models,
- of controllers that maintain consistency or coherence of the model suite,
- of application schemata for explicit maintenance and evolution of the model suite, and
- of tracers for the establishment of the coherence.

*Coherence* describes a fixed relationship between the models in a model suite.

The *collaboration* style of a model suite is based on supporting programs, data access pattern, style of collaboration, and coordination workflows. Collaboration pattern generalize protocols and their specification [16].

Let us assume that a model is defined in a language that uses constructors \( \mathcal{C} \) for the structuring and defining a model \( M \), i.e. \( M \in \text{Term}(\mathcal{C}) \) for the set of all terms defined in \( \mathcal{C} \). These constructors can be combined with an algebra \( \mathfrak{A} \) of expressions defined over \( \mathcal{C} \). Typical operations of the algebra are set operations such as union, difference
and intersection, constructing operations such as join, projection, selection, nesting (or integration/combination) and unnesting (or disintegration), and abstraction (or more specifically aggregation) operations. This approach to algebras follows approaches for universal algebras [19].

Each operation can be classified as either an identification-preserving or an identification-loosing one. Identification-preserving operations are, for instance, difference, intersection, nesting, and unnesting. An expression is identification-preserving if all its sub-expressions have this property.

We may use additional identification auxiliaries, i.e. constructions that define together with the given construction an identification-preserving expression.

A sub-model of $M$ can be either defined as a sub-expression of the expression that defines $M$ or as the application of an expression $E(M) \in \mathcal{A}(\mathcal{C})$ with one free variable $M$. For collaboration networks we choose the second approach and call them exchange sub-models. Sub-models can be identity-preserving or identity-loosing. Given any model or sub-model, an expression $E(M)$ defined on this model can also be considered as a mapping from $M$ to the resulting structure $E(M)$. Furthermore, we can use infomorphisms [9, 28] among models. Two models $M_1, M_2$ are $E_1, E_2$-infomorph though two transformations $E_1, E_2$ with $E_1(M_1) = M_2$ and $E_2(M_2) = M_1$ if any object $o$ defined on $M_1$ can be mapped via $E_i$ to objects defined on $M_j$ for $i, j \in \{1, 2\}, i \neq j$.

4 The Model Capsule

4.1 The Model Capsule $\equiv$ Model $\oplus$ Exchange Sub-Models

Models are extended by sub-models that are either

1. abstractions of the given model similar to roll-up or aggregation techniques used in database technology [17] or
2. specialisations to a more specific model similar to refinement techniques used for abstract state machines [1] or
3. specific viewpoints of the given model similar to view schemata [29].

Sub-model specification is based on an algebra for abstraction, refinement and filtering. The algebra is also used for transformation of sub-models. Sub-models to be exported to another model can be transformed before becoming imported by another model.

A model capsule consists of a main model and many exchange sub-models. An exchange sub-model is either an export or an import exchange-sub-model. If it is an import sub-model then it must be identity-preserving\(^1\).

Exchange sub-models are used as mediator in research teams and provide all details that are necessary for collaboration (completeness) but only those (minimality). Exchange sub-models are either derived from the main model in dependence on the viewpoint, on foci and scales, on scope, on aspects and on purposes of partners or are sub-models provided by partners and transformed according to the main model. A team member thus can integrate an exchange sub-model in his/her main model, can propagate changes made by him/herself to other partners and can change the main model.

\(^1\) This restriction can be weakened if additional identification auxiliaries are used.
according to changes by partners. This model capsule is the main communication vehicle for collaboration. The propagation and transformation from and to partners can be based on contracts or protocols.

### 4.2 Collaboration Model Capsules

Model suites can be associated to each other based on exchange sub-models. Given model suites $M_i = (M_{i,1}, ..., M_{i,n_i})$ with $n_i$ exchange sub-models. A model suite $M_1$ is bound to a model suite $M_2$ via export/import sub-models $M_{1,2}$ and/or import sub-models $M_{2,1}$ if there exist expressions $E_{1,2}, E_{1,2}'', E_{2,1,2}, E_{2,2,1}, E_{2,1}$ such that $E_{1,2}$ extracts the sub-model $M_{1,2}$ from $M_1$, the transformation expression transforms this sub-model to a model $M_{T,1,2}$ that is infomorph to the import sub-model $M_{2,1}$ of $M_2$, i.e. formally

\[ E_{1,2}(M_1) = M_{1,2}, \]
\[ E_{2,1}(M_2) = M_{2,1}, \]
\[ E_{1,2}'(M_{1,2}) = M_{T,1,2}, \text{ and} \]
\[ M_{T,1,2} \text{ and } M_{2,1} \text{ are } E_{2,1,2}, E_{2,2,1}-\text{infomorph}. \]

We notice without proof that the infomorphism can be integrated into the transformation expressions for some special cases.

Expressions we use for model association may be, for instance, aggregation or abstraction expressions, viewpoint expressions, specialisation expressions, or also combination expressions. Therefore, a sub-model of a first model that is used for association with a second model may be more abstract, or may be oriented on specific elements of the first model, or may extend the first model. Abstraction allows to form a kind of generalisation, i.e. a vertical hierarchy. The model capsule is bound vertically. Specific or extended models are typically defined on the same level of abstraction. The model capsule is then bound horizontally.

This approach is sufficiently general for model-based communication and reasoning in interacting research and engineering communities. Each branch of engineering or science uses its specific model suite. In order to collaborate, an interdisciplinary theory is formed. The interdisciplinary theory corresponds to the association in the real world. For instance, model capsules are based on models $A$ and $B$ that use corresponding scientific disciplines and corresponding theories as a part of their background. The models...
have three derived exchange sub-models that are exported to the other capsule and that are integrated into the model in such a way that the imported sub-model can be reflected by the model of the capsule. The two models and the two scientific disciplines are the kernel for an interdisciplinary theory.

5 Realisation and Implementation of the Approach

Model suites have already been investigated for UML-based software engineering in [26] on the basis of [30]. M. Skusa investigated the association among modelling languages based on language mappings. Each of the diagram types got its own profile. These profiles have been used for automatic derivation of associations among UML diagrams. The direction of enforcement follows in this case waterfall development strategies, i.e. requirements diagrams cannot be changed by conceptual diagram changes. He also developed controllers that maintain consistency of diagrams within a model suite. These controllers have been written as rules based on Abstract State Machines [1]. Since ASM rules run in parallel all controller run in parallel.

The Extract-Transform-Load paradigm can be enhanced by derivation of functions that provide the basic database system CRUD functionality [34]. Therefore, exchange sub-models support database processing similar to classical technology.

Traditional object-relational approaches only support singleton table views. To overcome this limitation we define a complex view as a collection of views that are associated through integrity constraints - mainly (pairwise) (generalised) inclusion constraints. The view classes are computed in the first step from the basic database using the view expression and then mapped to a database based on the association schema. They thus form a local database on their own.

The concept of view towers [15] has already been used for the generation of interfaces. Views of level $i$ are schemata on their own and are incrementally constructed of the base database schema (level 0) and of views of level less than $i$. It has been shown that SQL and database technology nicely support such complex views [13]. The construction of view towers can be enhanced by a characterisation whether the view is updatable. A higher level view is strongly updateable if the algebraic expression that defines this view does not destroy updateability and each of its components of lower level is updateable. Views can be enhanced by auxiliary views that provide an enhanced updateability based on a combined view of the original one and the auxiliary ones that is itself updateable.

5.1 Realisation 1: Applicability of the Approach in Research Communities

Our approach supports collaboration for more complex applications discussed in Subsection 1.1. We define explicit transformation expressions. The notion of info-morphism becomes then however far more complex. Both research collaborations in the Clusters of Excellence at CAU Kiel are using ad-hoc model associations. In [6] model suites and views have been used for automatic recharge of archives. The development and maintenance of integrated, reusable and coherent archives for all data capturing project is a mandatory requirement issued by the German Research Council to integrated projects.
such as Clusters of Excellence. In [7], a general data store has been realised for all archeology and pre-historic data. The local projects have their import views to and export views from the general data store. The global data store consists of one component. This component contains all data from all projects in the Graduate School “Human Development of Landscapes” and a pair of an import and an export view for each of the projects. Project collaboration is based on collaboration export views for each collaborating community. The projects themselves have their own database schemata that correspond to the import view through an extract-transform-load feature.

Both database support projects are the basis and the background for the model capsule approach developed in this paper.

5.2 Realisation 2: Collaboration Model Capsules in Software Engineering

Let us now exemplify the concept for classical software engineering with an example adapted from [27].

Given a use case diagram, a class diagram, a package diagram, and an interaction diagram. These four diagrams can be associated by exchange sub-model for a use_case-package association in the upper part and package-interaction, package-class, and class-use_case associations in the lower part. Controllers maintain the coherence of the different viewpoints. In the lower part, we consider the package diagram to be the leading diagram for the class and interaction diagrams and the class diagram as a leading diagram for the use case diagram. The class diagram has an export sub-model to the use_case diagram that has an identity-preserving sub-model as an import sub-model. Controllers may use a restrict, eager or lazy approach, i.e. a change in the class-diagram is allowed
- only if this can be directly reflected in the use_case sub-model (restrict) or if this changed be directly (eager) or at a later stage (lazy) propagated to the importing sub-model of the use_case model and
- the change modifies the export sub-model in the class diagram.

The application schemata are derived from controllers based on templates or pattern similar to integrity maintenance for referential inclusion constraints in databases. Tracers are then small demons that observe whether a model changes its export and import models.

6 Conclusion

This paper proposes an approach to models and model-based reasoning for interacting research and engineering communities. Models are an “intergalactic” communication
and reasoning instrument and a crosscutting concern in such networks. Model-based communication and reasoning is based on the concept of the model capsule that provides a flexible and powerful mechanism for model-based reasoning and collaboration. They provide a flexible system to manage locally and to exchange globally models for collaboration in teams. Models thus become a crosscutting concern to reflect competence for an interdisciplinary collaboration and for interactive research on complex society issues that cannot be solved within a singleton discipline.

The role and potential of models in networked research communities has not yet been systematically investigated, explored and generalised. This paper tackles the collaboration challenge based on model-based data exchanging collaboration. Model-based collaboration is only one kind of collaboration beside the data-based, concept-based, workpiece-based, process-oriented etc. collaborations. It seems however that models are a central instrument for any qualified and dependable collaboration.

As the next step, we aim at a general model description language ModelML that allows to collect models in networks in a form similar to an online interactive encyclopedia or model web. This model web supports systematic elicitation and exploration of modelling experience in research networks.

References

Models and their Capability

Bernhard Thalheim and Marina Tropmann-Frick
Department of Computer Science, Christian-Albrechts-University Kiel, 24098 Kiel, Germany

Abstract. Models are one of the main instruments in Computer Science. The notion of model is however not commonly agreed due to the wide usage of models. It is challenging to find an acceptable and sufficiently general notion of model due to the large variety of known notations. Such notion should incorporate all of the different notations and at the same time should allow to derive the specific notation from the general notion. We introduce a universal parameterised notion of the model. The parameters in this notion support adaptation of the universal notion of the model to the specific notation of interest. We finally apply this notion and this adaptation to development of business process models that are specified in BPMN.

1 The Model - an Artifact and an Instrument

Classical Computer Science research considers models as artifacts that are constructed in certain way and prepared for their utilisation according to the purpose under consideration such as construction of systems, verification, optimization, explanation, and documentation.

Creating for a practical purpose means that the main target of model development is its application in utilisation scenarios. Models are considered to be artifacts in a stronger sense. We observe however that models are developed for their utilisation within some scenario. They are functioning in this scenario. That means models are instruments in these scenarios. The notion of an instrument concentrates on this utilisation of models. Models are therefore mainly instruments that are effectively functioning within a scenario. The effectiveness is based on an associated set of methods and satisfies requirements of usage of the model.

1.1 Models - The Third Dimension of Science

Models are used as perception models, experimentation models, formal models, conceptual models, mathematical models, computational models, physical

1 An artifact is “something that is created by humans usually for a practical purpose” or “something characteristic of or resulting from a particular human institution, period, trend, or individual” or “a product of artificial character due usually to extraneous (as human) agency” [16]. The last meaning of the notion of an artifact is not taken into consideration for models in most sciences and also in Computer Science.

2 An instrument is among others (1) a means whereby something is achieved, performed, or furthered; (2) one used by another as a means or aid or tool [16].
Models, visualisation models, representation models, diagrammatic models, exploration models, heuristic models, etc. Experimental and observational data are assembled and incorporated into models and are used for further improvement and adaptation of those models. Models are used for theory formation, concept formation, and conceptual analysis. Models are used for a variety of purposes such as perception support for understanding the application domain, for shaping causal relations, for prognosis of future situations and of evolution, for planning, for retrospection of previous situations, for explanation and demonstration, for preparation of management, for optimisation, for construction, for hypothesis verification, and for control of certain environments.

Models are one of the main instruments in scientific research. They are considered to be the *third dimension of science* [26]³ (Figure 1). They provide a tool for research and have an added value in research, e.g. for construction of systems, for education, for the research process itself. Their added value is implicit but can be estimated based on the capability, potential and capacity of the model. Models are common culture and practice in sciences. However, each discipline has developed its own modelling expertise and practice.

Models are often language based. Their syntax uses the namespace and the lexicography from the application domain. Semantics is often implicit. The lexicology can be inherited from the application domain and from the discipline. Models do not need the full freedom for interpretation. The interpretation is governed by the purpose of the model within the research scenario,

³ The title of the book [4] has inspired this observation.
is based on disciplinary concerns (postulates, paradigms, foundations, commonsense, culture, authorities, etc.) and is restricted by disciplinary practices (concepts, conceptions, conventions, thought style and community [6], good practices, methodology, guidelines, etc.). Models combine at least two different kinds of meaning in the namespace: referential meaning establishes an interdependence between elements and the origin (‘what’); functional meaning is based on the function of an element in the model (‘how’). The pragmatics of a model depends on the community of practice, on the context of the research task and especially on the purpose or function of the model.

A model can be used for different purposes and various usage scenarios. Therefore, a model is typically also extended by views or viewpoints that reflect certain parts of the model and that hide details which are not necessary. This reflection is often only provided in a non-systematic or implicit way. Additionally, we need a refinement notion, methods for combination and for evaluation of models.

1.2 Scenarios of Model Utilisation

Models are used as an instrument in some utilisation scenario. At the same time, the model might be useless and not productive in other scenarios. Their function in these scenarios is a combination of functions such as explanation, optimization-variation, validation-verification-testing, reflection-optimization, exploration, hypothetical investigation, documentation-visualization, and description-prescription as a mediator between a reality and an abstract reality that developers of a system intend to build.

Traditionally, purposes or goals are considered first. The purposes and the goals are used to determine the functions of a model. This approach is centered around the purpose or goal and requires a definite understanding of the purpose and goal. Purposes and goals are often underspecified or blurry at the beginning. They become more clear after the model is being used. Compared to this approach, it is simpler to understand the application cases of a model and thus the utilisation scenarios. In this case we may derive the functions that a model has in these scenarios. Therefore, we use the approach that the functions of the model determine the purposes of the deployment of the model.

1.3 The Storyline of the Paper

The large variety of model notations (see, for instance, [13, 23, 30]) does not allow to transfer experience gained with one notation to other notations. Methods for utilisation or development are therefore mainly bound to one notation. Each subdiscipline has therefore its own understanding of modelling. It would however be beneficial to have a general notion of model that can be adapted to the specific notations of interest.

We introduce in Section 2 a universal notion of a model. This notion is based on the understanding of a model as an instrument in some utilisation scenarios. We only consider well-formed instruments since models must be intuitive and easy to understand. The model definition is based on two general
parameter sets, adequateness and dependability. Each of the parameters can be instantiated in dependence of the function that the model should have in a given utilisation scenario within the sub-discipline. This instantiation facility is based on a conception frame for the model notion.

The approach is applied to BPMN modelling in Section 3. We describe the business process modelling approach and derive the capability of this modelling technique. We can now also explicitly describe the obstacles of BPMN modelling. Furthermore, we derive the evaluation procedure for the BPMN approach in Section 4.

This approach to modelling in Computer Science can now be used as a starting point of a theory of modelling (Section 5). We start with some, often implicitly given restrictions that a model has, esp. its burden by the background and by the directives. The evaluation of models also supports a statement on not-supported utilisations, called anti-profile. Finally, the conception frame can also be used for development of question forms that support model specification.

2 The Universal Notion of the Model

There are many notions of models. Each of them covers some aspects and concentrates on some properties such as the mapping, analogy, truncation, pragmatism, amplification, distortion, idealisation, carrier, added value, and purpose properties [11, 17, 18, 21]. The main property is however the function property: The model suffices in its function in the utilisation scenarios that are requested. This property results in the following notion of the model [25, 27, 29].

2.1 The Model Notion

Models have several essential properties that qualify an instrument as a model [22, 24]:

Definition 1. An instrument is well-formed if it satisfies a well-formedness criterion.

Definition 2. A well-formed instrument is adequate for a collection of origins if (i) it is analogous to the origins to be represented according to some analogy criterion, (ii) it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and (iii) it is sufficient to satisfy its purpose.

Definition 3. Well-formedness enables an instrument to be justified: (i) by an empirical corroboration according to its objectives, supported by some argument calculus, (ii) by rational coherence and conformity explicitly stated through formulas, (iii) by falsifiability that can be given by an abductive or inductive logic, and (iv) by stability and plasticity explicitly given through formulas.
Definition 4. An instrument is sufficient by a quality characterisation for internal quality, external quality and quality in use or through quality characteristics [20] such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

Definition 5. A well-formed instrument is called dependable if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

Definition 6. An instrument is called model if it is adequate and dependable. The adequacy and dependability of an instrument is based on a judgement made by the community of practice.

Definition 7. An instrument has a background consisting of an undisputable grounding from one side (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities) and of a disputable and adjustable basis from other side (assumptions, concepts, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense).

Definition 8. A model is used in a context such as discipline, a time, an infrastructure, and an application.

The model notion can be depicted in Figure 2 based on the following conceptions:

- a fundament or background
  - the grounding, and
  - the (meta-)basis,

- four governing directives given by
  - the artifacts or better origins to be represented by the model,
  - the deployment or profile of the model such as goal, purpose or functions,
  - the community of practice (CoP) acting in different roles on certain rights through some obligations, and
  - the context of time, discipline, application and scientific school,

- two pillars which provide
  - methods for development of the model, and
  - methods for utilisation of the model,

and finally

- the model utilisation scenario for the deployment of the model in the given application.

The model house in Figure 2 abstracted from its full version [24, 27] displays these different facets of the model. The house consists of a cellar (basis in Figure 2) and a fundament (grounding in Figure 2), two pillars (development resp. utilisation methods), four driving or governing forces (origins, purpose of function, community of practice, context), and finally the deployment roof (utilisation scenario). The grounding is typically implicitly assumed and not disputable. It contains paradigms, the culture in the given application area, the background, foundations and theories in the discipline, postulates, (juristic and other) restrictions, conventions, and the commonsense. The basis is the main part of the background and is typically disputable.
Definition 9. A fully-specified model is function-purpose-goal invariant if the model can be used instead of the origins in the given scenario and have the same goal, the same purpose, and the same function. A model is solution-faithful if the solution of the problem solved with the model is analogous in the world of the origins based on the analogy criterion that is used for stating adequacy.

2.2 The Conception Frame for the Model Notion

The model notion covers many different aspects. It might thus be of interest whether there is a guideline for development of models. Models are artifacts that can be specified within a W*H-frame [5] that extends the classical rhetorical frame introduced by Hermagoras of Temnos. Models are primarily characterised by W⁴: wherefore (purpose), whereof (origin), wherewith (carrier, e.g. language), and worthiness ((surplus) value). The secondary characterisation dimensions are given by: (1) stakeholder: by whom, to whom, whichever; (2) additional properties of the application domain: wherein, where, for what, wherefrom, whence, what; (3) solution: how, why, whereto, when, for which reason; and (4) context: whereat, whereabout, whither, when.

A practical guideline may just

---

⁴ Quis, quid, quando, ubi, cur, quem ad modum, quibus adminiculis (Who, what, when, where, why, in what way, by what means), The Zachman frame uses a simplification of this frame.
1. start with fixation of two directives: origins to be represented and community of practice that accepts this model;
2. restrict the model utilisation scenario and the usage model to those that are really necessary and thus derive the purpose and function of the instrument;
3. define adequateness and dependability criteria of the instrument within the decision set made so far;
4. explicitly describe the background of the model, i.e. its undisputable grounding and the selected basis; and
5. explicitly specify the context for utilisation of the model.

Fig. 3. Conception frame for systematic development of a model

The model development and utilisation depends in this case from: Judgements of some members of the CoP to deploy the instrument as a model for some origin based on an assessment (deployability, rigidity, modality, confidence) within a CoP, utilisation scenario, and within a context. Utilisation scenarios and use spectra accepted for the instrument with functions of the instrument in utilisation scenario, roles and deployment of the instrument in those scenario, and resulting purposes and goals for the utilisation.
The instrument as such with some appreciation
as a well-shaped instrument on the basis
– of some criteria in dependence on intended utilisation and criteria
for:
  • what is accepted in a CoP, and
  • what is syntactically, semantically, pragmatically well-shaped,
that fits to the intended use, and
is appropriate for the utilisation scenarios and the use spectra.

The orientation also reflect our understanding of a model as an instrument.

3 BPMN Diagrams as Models

The Business Process Modeling and Notation (BPMN) language [8, 14] is a
conceptual business process specification language and is standardized by the
Object Management Group (OMG). There are many different languages for
description of business stories (e.g. SiteLang), of business rules (e.g. business
use cases), and of workflows that are essentially specifications of business pro-
cesses, activities of participants, utilisation with resources, and of communi-
cation among the participants. Languages such as S-BPM, BPMN, and EPC
concentrate on different aspects of business processes, vary in scope and fo-
cus, use different abstraction levels, and are thus restricted in the capacity and
potential for modelling. Most of the existing languages evolved over their life-
span and extensionally added features, more features, and other features again.
BPMN is not an exception for this kind of overloading.

A business process consists of an ordered set of one or more activities (tasks)
which collectively realize a business objective or policy goal. A workflow is the
executable specification of a business process. It may describe all or some of
the five aspects of business processes [15]:

(1) control flow description for the partial order of the activities, events or
steps;
(2) organisation description with participants, theirs roles and plays within
the processes, their rights and obligations, their resources, and their assign-
ments;
(3) the data viewpoint description with an association to process elements
and access rights for participants;
(4) the functional description that specifies semantics, pragmatics, and be-
haviour of each element of the workflow, e.g. the operations to be per-
formed, pre- and postconditions, priority, triggers, and time frames for the
operations;
(5) the operational assignment of programs that support all elements of the
workflow.

The entire modelling process is based on a local-as-design perspective. A
holistic or global view on a diagram collection is the task of a designer and
becomes problematic in the case of specification evolution.
BPMN 2.0 defined four different kinds of diagrams for workflow specification. We shall briefly review these diagrams in the sequel. The diagram in Figure 4 combines these different aspects. It describes the accomplishment of requirements issued by a customer.

Fig. 4. Fulfillment of customer demands by vendors

3.1 Diagrams in BPMN

Process Diagrams. Process diagrams (also called orchestration diagrams) describe the stepwise task flow for one agent. A task flow might reflect different roles of an agent. These roles are separated by swimlanes. Processes are either public or private. Public processes can also be abstractions of private processes that represent the detailed control and task flow for a singleton agent. Main process elements are (a) atomic or complex activities for direct representation of stepwise actions of an agent, (b) gates for exclusive, non-exclusive, event-based or parallel splitting and joining of the control flow, (c) events for the start of a workflow, for the end of a workflow path, for the complete end of the entire workflow, and an intermediate event for representation of interaction events with agents outside the workflow, and (d) control flow arrows for representation of the order of process elements. Basic activities reflect abstract, service, send, receive, user, manual, business rule, or script tasks. Complex tasks reflect an entire sub-workflow, loops, parallel or sequential multiple executions, ad-hoc workflows, transactions, or specific exception handling workflows such as compensation. Interaction reflects message exchange, timer interaction, escalation enforcement, compensations, conditional interactions, links, and signals. Interaction can be sequential or parallel multiple. Interaction events may also
be boundary events for a complex activity. All elements can be explicitly annotated by comments, by consumed data, or by produced data objects or data stores.

**Choreography Diagrams.** Choreography diagrams describe the message exchange among agents with reference to sending and receiving events, the message issue, and the graph-based representation of the partial order among these messages.

**Collaboration Diagrams.** Collaboration use choreography diagrams and process diagrams for explicit binding of senders and receivers of messages to black-box abstractions of agent workflows and abstract from message issues.

**Conversation Diagrams.** Conversation diagrams survey communication flow among agents as a birds view. They allow to derive dependences among process diagrams of agents.

3.2 Capability of BPMN Diagrams

BPMN modelling becomes nowadays a standard for typical business applications. Therefore, the capability of processes must be specified and well understood. It is thus necessary to know what is the ability to achieve a good model through a set of controllable and measurable features.

BPMN diagrams require a work-around for a number of conceptions such as macro-state, history, and system architecture. There are redundancies in the language itself that lead to flavour- or taste-oriented programming due to the overwhelming number of elements, construct excess and overload, e.g. groups, pool and lane, transformations, off-page connectors. The structuring becomes unclear since activities can be itself a workflow or a collection of workflows. This rather specific kind of abstraction should not be mixed with abstraction in general. Exception handling is completely confusing and only partially defined.

BPMN diagrams can represent only 8 out of 43 workflow resource pattern [10]. The data aspect is provided through properties of tasks, processes, and sub-processes. Their interrelationship is left to the developer community. It is the task of the developers to keep in mind the entire picture of the BPMN diagram collection.

BPMN uses an informal approach to semantics description what has been a matter of confusion. A formal approach to BPMN semantics can however be developed [1–3].

Furthermore, there is no conception of well-formed diagrams. Decomposition and composition is left to the developer. BPMN does not properly support the aspects (2), (4), and (5). The data aspect (3) is partially represented.

3.3 Deficiencies of Diagrams and Diagrammatical Reasoning

Diagrams are not universal for modelling. It is often claimed that diagrams are simple to use, are easy to interpret, have an intuitive semantics, are unique within a user community and have thus a unique pragmatics, and are thus powerful instruments. We observe however a number of obstacles that must be resolved before accepting a diagram as a model, e.g. the following ones:
Habituation versus unfamiliarity: Diagrams should be familiar to their users, have a unique semantics and pragmatics without any learning effort. Readers of diagrams must be literal with them.

Ambiguity of interpretation versus well-formedness: Diagrams should not confuse by multiple interpretations (e.g. arrows), by instability and by context-dependence of form-content relations.

Incremental graphical construction: Diagrams should follow the same construction pattern as the origin and should concentrate on typicality.

Naturalness of local reasoning: Local-as-design approaches presuppose locality within the world of origins.

Unfamiliarity with non-linear behaviour: Users are mainly linearly reasoning. Non-linear reasoning should be supported in a specific form.

Additional and supplementary elements without meaning: Diagrams of use elements which do not have a unique or any meaning, e.g. colours, shapes, grid forms for lines etc.

Hidden dimensions within the diagram: Diagrams cannot reflect all aspects although there are essential ones, e.g. time.

Representation as fine and visual art: Finding a good representation is a difficult task and should be supported by a culture of modelling.

All these obstacles are observed in the case of BPMN diagrams [10].

Diagrams must be developed on the principles of visual communication, of visual cognition and of visual design [12]. The culture of diagramming is based on a clear and well-defined design, on visual features, on ordering, effect, and delivery, and on familiarity within a user community.

One of the main obstacles of diagrams is the missing abstraction. The simplest way to overcome it is the development of a model suite [19] consisting of a generic model and its refinement models where each of them is adequate and dependable. Generic models [31] reflect the best abstraction of all models within a model suite.

4 Evaluation of the BPMN Approach

BPMN is a powerful diagrammatic languages that uses more than 100 modelling elements. The same situation in the reality or the implemented system can be specified by a variety of diagrams. Since a theory of diagram equivalence is missing, [27] introduced seven evaluation methods for models:

- PURE – SMART – CLEAR evaluation for the goal-purpose-function evaluation of an instrument in a given application context, for given artifacts to be represented, for a given community of practice, and for a given profile (goal, purpose, and function) under consideration of the utilisation scenarios;
- PEST evaluation for assessment of internal, external, and quality of use;
- QUARZSAND evaluation for assessment of the model development, and
- SWOT – SCOPE evaluation for description of the potential of the model, i.e. the general properties of a given instrument or the modelling method.
Since we did not explore the directives in detail nor the adequateness and dependability of an instrument that is a candidate model, we concentrate on the last two methods in this section. The evaluation of adequacy and dependability has been developed in [28]. We concentrate here on the capacity and potential of the BPMN approach.

4.1 The Capacity of the BPMN Approach

Capacity is a strategic measure whereas the potential is a tactical one. The potential can be used to derive the added value of a utilisation of a model within a given scenario. The potential allows to reason on the significance of a model within a given context, within a given community of practice, for a given set of origins, and within the intended profile.

The capacity relates an instrument to utilisation scenarios or the usage spectra. We answer the questions whether the instrument functions well and beneficial in those scenarios, whether it is well-developed for the given goals and purposes, whether it can be properly, more focused, comfortably, simpler and intelligible applied in those scenarios instead of the origins, and whether the instrument can be adapted to changes in the utilisation. The answers to these questions determine the main content or cargo, the comprehensiveness, and the authority or general value of a model. Another important aspect is the solution-faithfulness of the instrument. The capacity is an essential element of the model cargo, especially of the main content of the model.

BPMN diagrams can be used in description, prescription, explanation, documentation, communication, negotiation, inspiration, exploration, definition, prognosis, reporting and other scenarios. We discover that communication, negotiation, and inspiration are supportable. Description, prescription, and definition can be supported if the BPMN diagrams are enhanced and a precise semantics of all BPMN elements is commonly used in all four kinds of diagrams. The adequacy and especially the analogy to the origins (i.e. storyboards or business processes) is assumed to be based on homomorphy what is rarely achieved. This homomorphy is suitable if all processes are completely and in detail specified and all variations and exceptions are consistent.

The general utility of BPMN diagrams becomes rather low if the specific background of the modelling approach is not taken into consideration. BPMN diagrams are process-oriented, based on an orthogonal separation of flow element into activities, gates, and events, differentiate actors within their roles, and support communication among actors based on message exchange. The execution semantics is based on a token interpretation of control flow. Actors are isolated in their execution if binding is not done through message exchange or implicit hidden resource conditions. Data and resource are however local. All processes are potentially executed in parallel. The local-as-design approach might be appropriate if business processes are not intertwined. The concentration on the same abstraction level restricts the applicability of BPMN modelling. Generic workflows [31] provide a solution to this limitation.
4.2 The Potential of the BPMN Approach

The potential describes the (in-)appropriateness of a modelling approach within the directives. The suitability of BPMN diagrams depends on whether the application and the context support the local-as-design approach, on whether the demands of the community of practice can be satisfied, on whether the instrument is adequate (analogous, focused, purposeful), on whether the goals can be achieved with the given instrument, on the fruitfulness of the instrument compared with other instruments, and on the threats and obstacles of utilisation of BPMN diagrams.

4.3 The SWOT Evaluation of the Potential

The SWOT analysis is a high-level method that allows to evaluate the general quality of an instrument and its general assumptions of deployment.

**Strengths.** The BPMN approach is standardised and uses a large variety of constructs. It thus allows development of detailed models. It has a high expressibility. Both intra- and inter-organisational aspects can be represented. The approach is well supported by tools.

**Weaknesses.** The large variety of competing elements is also a weakness. The complexity and integration of diagrams may cause solution-unfaithfulness. The language requires high learning efforts. Processes that are dynamic at runtime cannot be modelled. Exchange among tools is an open problem.

**Opportunities.** Most business processes can be adequately described due to the variety of elements. The standardisation provides at least a base semantics.

**Threats and Risks.** None-technical users might be unable to cope with diagrams. Work-arounds hinder comprehensibility. Vendors define their own extensions. The BPMN standard does not completely define the execution.

4.4 The SCOPE Evaluation of the Potential

The SCOPE analysis of a model embeds the model into the application context, refines the capacity evaluation of an instrument, and considers the community of practice and their specific needs.

**Situation.** BPMN diagram suites provide some kind of formalisation of business processes. Communication is specified to a certain degree. Control flow is well-represented.

**Competence.** BPMN diagrams must be combined with other models since the other four aspects (organisation, data flow, functions, operational assignment) are only partially reflected.

**Obstacles.** Typical challenges of BPMN modelling are the specification complexity, diagram coherence, exception handling, and the development of an execution semantics. There is no common agreement on well-formedness of diagrams.
**Prospects.** A separate BPMN diagram is easy to read and to interpret. 

**Expectations.** The BPMN approach can be combined with local-as-design-oriented conceptual data models, storyboards, business rule specification and other modelling approaches as one kind of models within a model suite.

### 4.5 The Resulting Potential of the BPMN Approach

The BPMN diagram has a high potential for communication and negotiation utilisation scenario. The potential for system construction within a description-prescription scenario is however rather limited due to missing co-design support. A similar inappropriateness can be stated for explanation, prognosis, exploration, definition, and reporting scenario. The potential within a documentation scenario is rather small. The highest potential of the BPMN approach can be however observed for inspiration scenario. The process, choreography, conversation, and collaboration diagrams are an appropriate means for an implementation plan based on inspiring diagrams.

Similar to SPICE assessments [7], we may rate maturation of a model and a modelling approach to: (0) ad-hoc, (1) informal, (2) systematic and managed, (3) standardised and well-understood, and (4) optimising and adaptable, and (5) continuously improvable styles. The evaluation shows that the BPMN approach has not yet reached level (2). This observation leads us to the conclusion that PURE-SMART-CLEAR and PEST evaluations are heavily dependent from the directives for BPMN diagram modelling.

A model must be of high utility, must have a high added value, and should have a high potential. These parameters also depend on the well-formedness of the instrument. The BPMN approach can be enhanced by criteria for well-formedness for syntactical, semantical and pragmatically well-shaped diagrams [28].

### 5 Towards a Theory of Modelling

#### 5.1 Models Burdened with Directives and Background

The directives and the background (see Figure 2) heavily influence the way how a model is constructed, what is taken into consideration and what not, which rigidity is applied, which basis and grounding is taken for granted, and which community of practice accepts this kind of model. The model incorporates these influences without marking them in an explicit form. The model is laden or burdened by these decisions. Additionally, models are composed of elements that are selected, changed and adapted within a development process. Figure 5 depicts elements of this burden and this development history.

#### 5.2 The Anti-Profile of an Instrument as a Model

We may now directly conclude that an instrument might or might not be adequate and dependable for any utilisation scenario due to its insufficiency to function in this scenario.
Definition 10. A **utilisation pattern** of an instrument describes the form of usage of an instrument, the discipline of usage, the applications in which the instrument might be used, and the conditions for its utilisation.

Definition 11. A **utilisation scenario** consists of a utilisation pattern and a number of functions a specific instrument might play in this utilisation pattern.

Definition 12. A **usage spectrum** consists of collection of utilisation scenarios. A **portfolio** of an instrument combines the usage spectra.

---

Fig. 5. Models are burdened by their development history, the background and the directives

Definition 13. A **profile of an instrument as a model** consists of the goals, the purposes, and the functions of the instrument within a portfolio.

We can now roughly describe an **anti-profile** of a model and resulting **utilisation proscriptions** of a an instrument as a model by answering the following questions:

- For which scenarios is the instrument useless?
In which of the following scenarios efficiency and effectiveness is not given for the instrument: description & prescription, realisation & coding, theory development, theory refinement, causality consideration, inexplicability, demonstration, prediction, explanation, mastering of complexity, understanding, or ...?

Are essential parameters of the origins missing? Are some of the essential parameters only represented via mediating or dependable parameters? Are there dummy or pseudo dependences among the parameters?

What cannot be adequately represented? Is the dependability really sufficient? In which case users need a special understanding and education? Which tacit knowledge is hidden in the instrument?

In which cases the instrument cannot be effectively used? What must a user respect and obviate before using the instrument?

Which biases and which background are palmed off? Which assumptions, postulates, paradigms, and schools of thought are hidden and not made explicit? Models might condition conclusions.

Since models are instruments their utilisation conditions conclusions and results. Therefore, it is appropriate to describe the anti-profile of a model as well.

5.3 Questions to Answer Before Using an Instrument as a Model

The rhetoric frame and its extension to the W*H frame [5] can now be used for derivation of questions one must answer before using the model:

What is the function of the model in which scenario? What are consequential purposes and goals? What are anti-goals and anti-purposes?

Which origins are going to be represented? Which are not considered? Does the model contain all typical, relevant and important features of the origins under consideration and only those?

Is the instrument adequate and dependable within the utilisation scenario? What are the parameters for adequacy and dependability? How purpose-invariance and solution-faithfulness is going to be defined?

What kind of reasoning is supported? What not? What are the limitations? Which pitfalls should be avoided?

Do you want to have a universal model that contains all and anything? Would it be better to use a model suite where each of the models represent some specific aspects? What about the nonessential aspects?

6 Conclusion

A general understanding of the notion of a model has been already started with development of Computer Science. Milestones are the papers and books by H. Stachowiak (1980ies and 1990ies), B. Mahr (2000ies until 2015), W. Steinmüller (1993), and R. Kaschek (since 1990ies) [9, 11, 17, 18, 21]. These notions treat models from a phenomenological point of view through properties that a model should have (e.g. as main properties: mapping or analogy, truncation, pragmatic...
properties). We need however also an explicit definition of the notion of model. Such general notion has been developed in a series of papers, e.g. [22, 24, 25, 27, 29].

The model notion is universal one and based on two parameter sets for adequateness and dependability. The parameter sets seem to be complex and need a methodological support. This paper develops such a support facility based on the notion of a conception frame. The practicality of the approach is demonstrated for the workflow specification language BPMN. BPMN shares the positive treatment with most other formal or informal languages in Computer Science. The capacity and thus the restrictions or obstacles are not explicitly communicated. We see however that the evaluation, capacity, potential, and capability can be explicitly provided based on our approach.

Since the model notion is a mathematical definition, it seems to be achievable to develop a theory of modelling in the sense of a theory. In this paper, we only discuss two components of such a theory: the explicit description of the background of models and the anti-profile. The conception frame for the model definition may also be used for derivation of question forms that a modeller can use before delivering an instrument as a model to a community of practice. The development of a full theory is however a research issue for the next decades.

References


Enhancing Entity-Relationship Schemata for Conceptual Database Structure Models

Bernhard Thalheim and Marina Tropmann-Frick

Department of Computer Science, Christian-Albrechts-University Kiel, 24098 Kiel, Germany

Abstract. The paper aims at development of well-founded notions of database structure models that are specified in entity-relationship modelling languages. These notions reflect the functions a model fulfills in utilisation scenarios.

1 Utilisation Scenarios of Conceptual Models

Conceptual models are used as an artifact in many utilisation scenarios. Design science research [4] and ER schema development methodologies (e.g. [3, 8, 11]) developed so far a good number of such scenarios.

Communication and negotiation scenario: The conceptual model is used for exchange of meanings through a common understanding of notations, signs and symbols within an application area. It can also be used in a back-and-forth process in which interested parties with different interests find a way to reconcile or compromise to come up with an agreement. The schema provides negotiable and debatable propositions about the understanding of the part of the reality but does not have well-developed justificatory explanations.

Conceptualisation scenario: The main application area for extended entity-relationship models is the conceptualisation of database applications. Conceptualisation is typically shuffled with discovery of phenomena of interest, analysis of main constructs and focus on relevant aspects within the application area. The specification incorporates concepts injected from the application domain.

Description scenario: In a description scenario, the model provides a specification how the part of the reality that is of interest is perceived and in which way augmentations of current reality are targeted. The model says what the structure of an envisioned database is and what it will be.

Prescription scenario: The conceptual model is used as a blueprint for or prescription of a database application, especially for prescribing the structures and constraints in such applications. The schema proposes what the structure of a database is on the one hand and how and where to construct the realisation on the other hand. ER schemata can be translated to relational, XML or other schemata based on transformation profiles [11] that incorporate properties of the target systems.
These scenarios are typically bundled into use spectra. For instance, design science uses three cycles: the relevance cycle based on the design cycle based on description and communication and negotiation scenario, and the rigor cycle based on a knowledge discovery and experience propagation scenario. Database development is mainly based on description, conceptualisation, and construction scenarios. The re-engineering and system maintenance use spectrum is based on combination of documentation scenarios with an explanation and discovery scenario from one side and communication and negotiation scenario from the other side. Models are also used for documentation scenarios, explanation and discovery scenarios for applications or systems, and for knowledge experience scenario. We concentrate here on the four scenarios.

Contribution of the Paper

The first main contribution of this paper is an analysis whether the an entity-relationship schema is suffices as a model for database structures. We realise that the four scenarios require additional elements for the ER schema in order to become a model. The second main contribution of this paper is a proposal for an enhancement of ER schemata which allows to consider the artifact as a model within the given four scenarios. The paper partially presupposes our research (esp. [14], see also other papers in [15]).

2 The General Notion of a Model

Science and technology widely uses models in a variety of in utilisation scenarios. Models function as an artifact in some utilization scenario. Their function in these scenarios is a combination of functions such as explanation, optimization-variation, validation-verification-testing, reflection-optimization, exploration, hypothetical investigation, documentation-visualization, and description-prescription as a mediator between a reality and an abstract reality that developers of a system intend to build. The model functions determine the purposes of the deployment of the model.

The following notion of the model has been developed [16] after an intensive discussion in workshops with researchers from disciplines such as Archeology, Arts, Biology, Chemistry, Computer Science, Economics, Electrotechnics, Environmental Sciences, Farming and Agriculture, Geosciences, Historical Sciences, Humanities, Languages and Semiotics, Mathematics, Medicine, Ocean Sciences, Pedagogical Science, Philosophy, Physics, Political Sciences, Sociology, and Sport Science.

**Definition 1.** A model is a well-formed, adequate, and dependable artifact that represents origins. Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilisation scenarios. As an artifact, a model is grounded in its community’s sub-discipline and is based on elements chosen from the sub-discipline.
This notion has been tested against the notions of a model that are typically used in these disciplines. We could state that these notions are covered by our notion. Origins of a model [7] are artifacts the model reflects. Adequacy of models has often been discussed, e.g. [6, 9, 10]. Dependability is only partially covered in research, e.g. [5].

Models have several essential properties that qualify an artifact as a model [14, 15]:

- An artifact is well-formed if it satisfies a well-formedness criterion.
- A well-formed artifact is adequate for a collection of origins if (i) it is analogous to the origins to be represented according to some analogy criterion, (ii) it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and (iii) it sufficient satisfies its purpose.
- Well-formedness enables an artifact to be justified: (i) by an empirical corroboration according to its objectives, supported by some argument calculus, (ii) by rational coherence and conformity explicitly stated through formulas, (iii) by falsifiability that can be given by an abductive or inductive logic, and (iv) by stability and plasticity explicitly given through formulas.
- The artifact is sufficient by its quality characterisation for internal quality, external quality and quality in use or through quality characteristics [13] such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).
- A well-formed artifact is called dependable if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.
- An artifact is called model if it is adequate and dependable. The adequacy and dependability of an artifact is based on a judgement made by the community of practice.
- An artifact has a background consisting of an undisputable grounding from one side (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities) and of a disputable and adjustable basis from other side (assumptions, concepts, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense).
- A model is used in a context such as discipline, a time, an infrastructure, and an application.

The Taxonomy of Conceptual Models

The starting point in our investigation was the observation that there is no unique and commonly agreeable notion of the conceptual database structure model as such. The model supports different purposes and has different functions in utilisation scenarios. Therefore, we must have different notion of the conceptual database structure model.

Due to space limitations we concentrate on the first four utilisation scenarios. The other four scenarios are supported by specific conceptual models in a similar form.
3 Conceptual Database Structure Models for Communication and Negotiation

Communication aims at exchange of meanings among interested parties. The model is used as a means for communication. It truly represents some aspects of the real world. It enables clearer communication and negotiation about those aspects of the real world. It has therefore potentially several meanings in dependence on the parties. Communication acts essentially follow rhetoric frames\(^1\), i.e. they are characterised through “who says what, when, where, why, in what way, by what means” (Quis, quid, quando, ubi, cur, quem ad modum, quibus adminiculis). In our case, the model ("what") incorporates the meaning of parties (semantical space; "who") during a discourse ("when") within some application with some purpose ("why") based on some modelling language.

Typically, artifacts used for communication and negotiation follow additional principles: Viewpoints and specific semantics of users are explicitly given. The artifact is completely logically independent from the platform for realisation. The name space is rather flexible. The model is functioning and effective if methods for reasoning, understanding, presentation, exploration, explanation, validation, appraisal and experimenting are attached.

Conceptual model for communication: The conceptual database structure model comprises the database schema, reflects viewpoints and perspectives of different involved parties \(\mathcal{U}\) and their perception models, and implicitly links to (namespaces or) concept fields of parties. Adequacy and dependability are based on the association of the perception models to viewpoints and of the viewpoints with the schema.

A partial communication model does not use a schema and does not associate viewpoints to schema elements.

Therefore, the model can be formally defined as a quintuple

\[
(S, \{ (V_i, \Phi_i) | i \in \mathcal{U} \}, \{ (P_i, \Psi_i) | i \in \mathcal{U} \}, A, D)
\]

that relates elements of the conceptual schema \(S\) to the perception model \(P_i\) of the given party \(i\). The perception model is reflected in the schema via viewpoints \(V_i\). It implicitly uses concept fields \(C_i\) of parties \(i\). The mapping \(\Psi_i : P_i \rightarrow V_i\) associates the perception model of a given party \(i\) to the agreed viewpoint. In the global-as-design approach, the viewpoint \(V_i\) is definable by some constructor \(\Phi_i\) defined on \(S\). The adequacy \(A\) is directly given by the second and third parts of the model. The justification \(J\) and the dependability \(D\) are extracted from the properties of \(\Phi_i\) and \(\Psi_i\).

The negotiation scenario can thus be understood as stepwise construction of the mappings, stepwise revision of the schema and the viewpoints, and analysis whether the schema represents the corresponding perception model.

\(^1\) It relates back to Hermagoras of Temnos or Cicero more than 2000 years ago.
4 Conceptual Database Structure Models for Conceptualisation

Conceptualisation is based on one or more concept or conception spaces of business users. Given a business user community \(U\) with their specific concept fields \(\{C_i\mid i \in U\}\). Let us assume that the concept fields can be harmonised or at least partially integrated into a common concept field of users \(C_U\) similar to construction approaches used for ontologies.

Conceptual model for conceptualisation: The conceptual database structure model comprises the database schema, a collection of views for support of business users, and a mapping for schema elements that associates these elements to the common concept field.

Therefore, the model can be formally defined as a quintuple
\[
(S, V, M, A, D)
\]
consisting of the conceptual schema \(S\) and a mapping \(M : S \rightarrow C_U\). The adequacy \(A\) is based on the mapping. The justification \(J\) and the dependability \(D\) are derived from the concept fields.

5 Conceptual Database Structure Models for Description and Prescription

An artifact that is used as a conceptual model for database system description can be either understood as a representation, refinement and amplification [13, 15] of situation or reality models or as a refinement and extension of the communication model. The main approach to conceptual modelling for system construction follows the first option. The second option would however be more effective but requires a harmonisation of the perception models. The first option may start with reality models that reflect the nature of the business in terms and in the language of the business. It includes also the top management view, a corporate overview, and a sketch of the environment. The reality models are reasonable complete, are described in terms of the business and use general categories that are convergent.

Conceptual model for description: The conceptual database structure model comprises the database schema, a collection of views for support of business users, a collection of a commonly accepted reality models that reflects perception or situation models with explicit association to views, and the declaration of model adequacy and dependability.

Therefore, the model can be formally defined as a quintuple
\[
(S, V, (R, \Psi), A, D)
\]
The conceptual model may be enhanced by an association \(\Psi\) of views to the schema. This enhancement is however optional.

Descriptive models adequately explicate main concepts [12] from the reality models and combine them into views. The descriptive model reflects the origins
and abstracts from reality by scoping the model to the ideal state of affairs.

Prescriptive models that are used for system construction are filled with anticipation of the envisioned system. They deliberately diverge from reality in order to simplify salient properties of interest, transforming them into artifacts that are easier to work with. They may follow also additional paradigms and assumption beyond the classical background of conceptual database structure models: Salami slicing of the schema by rigid separation of concern for all types; conformance to methods for simple (homomorphic) transformation; adequateness for direct incorporation; hierarchical architecture within the schema, e.g. for specialisation and generalisation of types; partial separation of syntax and semantics; tools with well-defined semantics; viewpoint derivation; componentisation and modularisation; integrity constraint formulation support; conformance to methods for integration; variations for the same schema for more flexible realisation etc.

Directives (or pragmas) [1] prescriptively specify properties for the realisation. Transformation parameters [11] for database realisation are, for instance, treatment of hierarchies, controlled redundancy, NULL marker support, constraint treatment, naming conventions, abbreviation rules, set or pointer semantics, handling of weak types, and translation options for complex attributes. Based on [2] we give an explicit specification of directives for the realisation. The prescription model also consists of a general description of a realisation style and tactics, of configuration parameters (coding, services, policies, handlers), of generic operations, of hints for realisation of the database, of performance expectations, of constraint enforcement policies, and of support features for the system realisation. These parameters are combined to the realisation template $T$. The realisation template can be extended by quantity matrix for database classes $Q$ and other performance constraints $C$ and by business tasks and their reflections through business data units $B$. Directives can be bound to one kind of platform and represent in this case a technological twist, e.g. by stating how data is layered out. They are typically however bound to several platforms in order to avoid evolution-proneness of models.

Conceptual model for prescription: The conceptual database structure model comprises the database schema, a collection of views for both support of business users and system operating, a realisation template, and the declaration of model adequacy and dependability.

Therefore, the model can be formally defined as a quintuple $(S, \mathcal{V}, T, A, D)$.

6 Conclusion

This paper shows that the ER schema is a central unit in a conceptual database structure model. The conceptual database structure model contains also other elements in dependence on its function in utilisation scenarios. As long as we use a global-as-design approach, the ER schema is essential and the kernel of such database structure models.
We may combine the conceptual models to description/prescription models 
\((S, \mathfrak{B}, (\mathfrak{G}, \psi), \tau, \mathcal{A}, \mathcal{D})\).

and to description/prescription models with conceptualisation 
\((S, \mathfrak{B}, (\mathfrak{G}, \psi), \mathcal{M}, \tau, \mathcal{A}, \mathcal{D})\).

The combination with communication/negotiation is more problematic since the corresponding models are based on divergent perception models that might represent the very personal viewpoint of business users in different context and work organisation.

The notion of the model for conceptual database structure models can be summarised in dependence on their utilisation scenario:

**Table 1.** Conceptual database structure models that extend the conceptual database schema in dependence on utilisation scenarios

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Model origin</th>
<th>Add-ons to the conceptual database schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Communication and negotiation</td>
<td>Perception (and situation) models</td>
<td>Views representing the viewpoint variety and associated with the perception models</td>
</tr>
<tr>
<td>Conceptualisation</td>
<td>Perception and reality models</td>
<td>Associations to concepts and conceptions, semantics and meanings, namespaces</td>
</tr>
<tr>
<td>Description</td>
<td>Reality model</td>
<td>View collection, associations to origins</td>
</tr>
<tr>
<td>Prescription</td>
<td>Reality (and situation) models</td>
<td>View collection, realisation template</td>
</tr>
</tbody>
</table>

**References**


Theories in Business and Information Systems Engineering

Martin Bichler · Ulrich Frank · David Avison · Julien Malaurent · Peter Fettke · Dirk Hovorka · Jan Krämer · Daniel Schnurr · Benjamin Müller · Leena Suhl · Bernhard Thalheim

© Springer Fachmedien Wiesbaden 2016

1 Theories in Business and Information Systems Engineering

1.1 Introduction

Even though the idea of science enjoys an impressive reputation, there seems to be no precise conception of science. On the one hand, there is no unified definition of the extension of activities subsumed under the notion of science. According to the narrow conception that is common in Anglo-Saxon countries, science is restricted to those disciplines that investigate nature and aim at explanation and prediction of natural phenomena. A wider conception that can be found in various European countries includes social sciences, the humanities and engineering. On the other hand and related to the first aspect, there is still no general consensus on the specific characteristics of scientific discoveries and scientific knowledge.

1.2 Theory and Science

The demarcation problem in the philosophy of science is how to distinguish between science and non-science. Some argue that the demarcation between science and non-science is a pseudo-problem that would best be replaced by focusing on the distinction between reliable and unreliable...
knowledge, without bothering to ask whether that knowledge is scientific or not. Nevertheless, there seems to be one answer to Kant’s question concerning the difference between scientific insights and the dreams of a ghost-viewer that is accepted by many: At its core, scientific knowledge is based on theories. Therefore, research should be aimed at the construction and testing of theories. However, this conclusion is satisfactory only at first sight, because the concept of theory itself lacks a unified and commonly accepted definition. There seem to be various reasons for this surprising lack of conceptual clarity at the foundation of an enterprise that is aimed at linguistic precision.

First, the term “theory” is used for different kinds of epistemological constructions. That makes it difficult to develop a satisfactory general conception. Philosophy of science does not provide us with an accepted concept of theory either (Godfrey-Smith 2003). Formal theories developed using the axiomatic method as it is subject of mathematics and logic are not necessarily motivated by observations from the empirical world. Their truth can be proved, i.e., they can be verified with respect to the underlying axioms. Theories in the empirical sciences usually aim at gaining reliable descriptions of reality. Therefore, their justification will depend on some form of confrontation with a conception of reality which is coined by underlying epistemological and ontological assumptions. In the case of (neo)positivist approaches, this kind of justification is based on the correspondence theory of truth, which in turn has its background in a (critical) realist view of the world. Some philosophers of science aim at a (partially) formalized conception of empirical theories. The semantic view (Suppe 1989) regards theories as being comprised of sets of mathematical models and sets of models with an empirical claim. (Testable) hypotheses then serve to link both kinds of models. The ‘non-statement view’ of theories aims at specifying a formal structure, also called an “architectonic”, which should be suited to represent the “essential” features of empirical knowledge ....” (Balzer et al. 1987, xvii). The formal structure comprises a set of so called potential models (interpretations) of the underlying conceptual framework. Hermeneutic approaches which are rather based on different forms of constructivism or idealism make use of the coherence or the consensus theory of truth. In addition to that it is questionable whether truth is always the only justification criterion (Frank 2006).

Second, the actual use of the term is not only ambiguous but also ambivalent. A clear distinction between scientific (theoretical) and non-scientific knowledge is not trivial, if not impossible (Laudan 1983). Furthermore, studies in sociology of science show that scientific knowledge contributions are not independent from external factors such as incentives, expected reputation or power games (Feyerabend 1993; Kuhn 1964; Latour and Woolgar 1986). Sometimes it may seem that a theory is the result of a social construction – somebody has named it as such and his proposal was legitimized by being published in a top tier journal – rather than an epistemological distinction.

1.3 Theories in Our Field

The lack of a satisfactory conception of theory is especially critical in Information Systems or Business and Information Systems Engineering (BISE), respective. The wide range of research topics in our field comprises not only empirical theories, but also formal theories and the design of elaborate artifacts. At the same time, leading journals emphasize the need for theories, thereby creating a situation that is suited to create confusion. Various publications are aimed at targeting this problem.

Especially Gregor (2006) helps clarifying the use of theories in Information Systems. However, her work is mainly restricted to (neo-)positivist ideas of theory (Popper, Hempel/Oppenheim) and does not account for the peculiarities of formal theories or those conceptions of theory found in our neighboring disciplines economics, informatics, and management science, and also of those in several sub-communities of BISE. Frank (2006) suggests a meta conception of scientific knowledge that covers empirical, formal and design contributions, but does not provide a correspondingly wide conception of theory.

The situation is even worse when it comes to criteria that help assessing the quality of theories – especially with respect to the epistemological value of probabilistic propositions that are used by the majority of theoretical contributions in our field (Lim et al. 2009) – and that Popper refused to accept as proper theories. The problems caused by an ambiguous conception of theory in our field have been known for some time. In a recent debate that was triggered by Avison and Malaurant (2014) who question what they call the “theory fetish in information systems”, (Markus 2014, p. 342) comes to the conclusion “... that conflicting notions of theory and theoretical contribution, rather than sheer overemphasis on theory, may lie at the heart of the problem that Avison and Malaurant identified.”
A close look at theories relevant for our field results in a wide range of examples that are substantially different. For example, in informatics theoretical foundations such as automata theory, computability theory, complexity theory, or computational learning theory, which are typically based on the axiomatic method, constitute foundations for engineering sub-disciplines such as data engineering, data mining, and operations research. In those fields that focus on human behavior and action systems, many researchers follow a neo-positivist research paradigm with a concept of theory that leans on that common in the natural sciences. However, some researchers in these fields prefer hermeneutic approaches, e.g., for conducting case studies. Respective research methods do not only replace the idea of scientific objectivity with subjectivity, they sometimes deny the need for generalization.

The neo-positivist conception is challenged by a further principal concern that is directly related to a current subject of our research: the digital transformation. It is questionable whether research can provide an orientation for change if it is focused on actual or past patterns of developing and using IT. Instead, it may be more appropriate to emphasize the notion of theory (“theoría”): to transcend the “factual” world by contemplation. For us that means to look beyond current patterns of developing and using IT or – in other words: to develop justified (!) models of possible future worlds (Rorty 1999; Frank 2006) that serve those who live the future as an inspiration and a meaningful orientation. Respective constructions cannot be validated by confronting them with reality, since they are on purpose different from it.

Fields that make heavy use of formal models and methods are arguably very important for our discipline. They emphasize the power of mathematics and logic for representing scientific knowledge. While respective constructions come with obvious advantages as they allow for computing and proving, they come with the problem how to decide whether there is a valid empirical interpretation of socio-economic systems and whether actors can be expected to follow the rules of logic.

On the other hand, there are researchers that follow a more empiricist agenda, but aim to reconstruct their theories with formal models. This is particularly important in our field as human behavior cannot easily be characterized by a simple set of axioms. Empirical models of behavior can then be used to contrast axioms as they are used in theory. For example, independence of irrelevant alternatives is an axiom typically used in social choice theory. However, experimental research has found that human subjects often change their preferences over two alternatives if faced with an extended set of alternatives.

1.4 Theory and BISE Identity

The theoretical foundation of a scientific discipline has a substantial impact on its identity, and the identity of the IS discipline has led to significant discussion in the past. Some colleagues see themselves in the tradition of computer science and operations research, and they heavily draw on certain branches of mathematics, theoretical computer science (in particular algorithms and complexity theory), and statistics. Some colleagues are closer to economics and draw on economic theory, most notably microeconomics and industrial economics. Finally, the work of many colleagues is rooted in psychology and sociology, in particular when it comes to user perception and adoption of information systems.

Of course, the underlying theory has a substantial influence on the research being done and the criteria used to evaluate research. Some argue that IS needs to develop its own theories, which are distinct from reference disciplines. After all, it is not even easy to characterize what constitutes a theory, and the understanding of this is different in all of these reference disciplines. In any case, the current state of the discussion on theory in IS appears unsatisfactory.

Due to the fact that IT plays a role in more and more aspects of our lives, IS academics have looked into an ever growing number of subjects and IT-driven phenomena. Sometimes these phenomena are related to finance (e.g., crowd funding), sometimes to marketing (e.g., online shopping behavior), sometimes to systems engineering (e.g., enterprise architecture management), and sometimes to labor economics (e.g., online job markets). Nowadays research topics in BISE are largely interdisciplinary. While it is important to analyze all of these topics, our community is not the only one looking at these phenomena. It is important that we bring certain methods and theories to the table – a particular point of view that adds to the work of others in a valuable way. This is one, but of course not the only reason why it is important to be aware of the theoretical foundations of our work.

While some may regard a discussion of theories a mere philosophical exercise, we are convinced that a reflection on the foundations of our work – and its intended outcome – is essential. Without considering the existing variety of theory conceptions in our discipline, we cannot develop elaborate ideas of the ultimate goals of our work, of the justification and evaluation of research, of scientific progress and of proper ways to document scientific knowledge.

1.5 Contributions

We have collected the views of colleagues on the importance and nature of theories in their field. This was
intended to not only lead to a summary of different theoretical streams relevant to our research, but it might also influence the discussion about curricula in our field. We asked them to account for the following questions:

- Which conception of theory is central to your area of research?
- How do you evaluate progress in your field and what would you describe as long-term goal?
- In which way does theory guide design and engineering in your field and how does it impact practice?
- How do you evaluate the quality of theories in your field?

The contributions we received confirm that a debate on theory in our field is both challenging and inspiring. It is challenging because there is a variety of clearly different perspectives on the subject that indicates not only that we lack a common conception of theory, but that it might even be illusive to aim at one. At the same time, such a debate promises that “the object of our thought becomes progressively clearer” (Berger and Luckmann 1966) through the multitude of perspectives on it.

David Avison and Julien Malaurent used the opportunity to comment on their contribution to a debate on theory they had organized earlier (Avison and Malaurent 2014). There they questioned the “theory fetish” they observed in IS research and suggested that research would benefit from a more relaxed notion of theory, which they referred to as “theory light”. In their present contribution they emphasize that they did not mean to give up the quest for theory in IS research, but that there should be the opportunity for publishing ideas without referring to a rigorous notion of theory. Avison and Malaurent seem to assume that there is a common conception of theory in IS, since they do not discuss the conception of theory as such.

Peter Fettke focuses on particularities of research in Business and Information Systems Engineering (BISE) compared to IS. He argues that IS follows a model of research that has matured in the natural sciences, while BISE is rooted in engineering. While he regards referring to theories as a common, if not mandatory part of research in IS, he suggests that there are conceptual frameworks in BISE that are not called theory, but might as well qualify as such. While Fettke is reluctant to offer a definition of theory, he has a clear preference for a concept of theory that emphasizes the identification of cause–effect-relationships.

Dirk Hovorka proposes an inspiring relativist view on theory. He criticizes the common idea that a theory is a static linguistic structure that enables problem solving as misleading. Instead, he proposes a more dynamic view. Theories, as well as the conception of theory, are in a state of flux, they are representations of the ongoing discourse that constitutes the idea of science. Since such a discourse may stress a multiplicity of different perspectives on the subject of thought, theories may possess different forms and serve different purposes. Therefore, according to Hovorka, it would be inappropriate to aim at a common or integrated conception of theory. At the same time, such a view on theory implies giving up the common idea of scientific progress, because it denies the existence of criteria that would allow a clear discrimination of competing contributions to a common knowledge base.

In their research, Jan Krämer and Daniel Schurr follow a micro-economic paradigm that makes heavy use of mathematical models. Therefore, it does not come as a surprise that the conception of theory they suggest shows clear similarities to the notion of theory in mathematics. They regard models as interpretations of formal theories that help mediating between abstract structures and reality. To serve this purpose, models need to be designed with assumptions about the targeted domain in mind, which in turn requires some sort of empirical analysis. Hence, they claim that models serve as an instrument to develop appropriate formal theories that can be turned into theories with an empirical claim. They do not, however, advocate a pure realist conception of models. Instead, they regard models as analytical tools that may on purpose deviate from factual properties of reality.

Benjamin Müller distinguishes between positivist and non-positivist conceptions of theory and poses the question which one is more appropriate. He argues that scientific progress is likely to result from integrating and consolidating findings that are brought about by different research methods and paradigms. Consequently, he proposes that accounting for multiple perspectives should be a pivotal criterion for evaluating the quality of theories. He also advocates the conduction of research on post-adoption, that is to go beyond simplified models of technology adoption and focus on new patterns of (inter-) action that may emerge after the adoption of new technologies.

Leena Suhl’s view on theories reflects her work in operations research. She argues that operations research calls for enriching formal theories with empirical theories from the targeted domains, especially from economics, but also from fields such as manufacturing or marketing. Suhl suggests that the use of different types of theories contributes to the strength of the field, because it requires looking at the research subject from different perspectives. Therefore, she advises against aiming for a common conception of theory or even a comprehensive unified theory in Business and Information Systems Engineering. Instead, she suggests building and maintaining a common repository of relevant theories and methods that foster reuse.

Bernhard Thalheim argues that conceptual models are indispensable instruments of research in our field.
Therefore, he proposes a general model theory that is suited to guide the more reflected construction, use and evaluation of models. For this purpose, he suggests a conception of model and discusses its relationship to the concept of theory. Since he regards models as primary subjects of scientific thought, he recommends supplementing a general model theory with a theory of reasoning that would include foundational elements of reasoning about the construction, analysis, and use of models.

Prof. Dr. Martin Bichler
Technical University of Munich

Prof. Dr. Ulrich Frank
University of Duisburg-Essen

2 A Call for ‘Theory Light’ Papers

In our original paper published in *Journal of Information Technology* (Avison and Malaurant 2014), we argued that papers in our top journals need not only emphasize theoretical contributions, but could also, for example, emphasize new arguments, facts, patterns and relationships and thereby be ‘theory light’ and yet still make a major contribution to the discipline of information systems (IS). We gave some examples of such papers from IS and other management disciplines. We also provided several reasons for our concern about the present stress on theory in our journals, giving full explanations in that original paper:

1. Authors may be tempted to revert to ‘ideal types’ in our understanding process to make sense of the data within a theoretical framework.
2. Authors may be tempted to distort the description of the research setting so that it fits better to the chosen theory or theories.
3. There is no ‘recipe’ to help authors somehow fit the data to a theory and too few reflective accounts about how any potential gap between theory and data can be addressed, so that authors may be tempted to choose only those data that fit the story.
4. Authors may be tempted to choose theories that might be more related to ‘fashion’ or the fact that a theory developed in another discipline has yet to be ‘borrowed’ into IS, in order to provide an ‘original’ theoretical contribution, rather than to select a theory on the grounds of suitability considerations.
5. The requirement to emphasize theory in all our published papers has an opportunity cost as authors loose the opportunity to make other valuable contributions fully because of space issues. To move into ‘unexplored territories and arguments’ requires supporting explanations etc. to make the contributions convincing.
6. The requirement of a theoretical contribution in every paper makes some of these ‘contributions’ somewhat trivial. Many papers may contain ‘theoretical filling’ rather than making a substantial theoretical contribution. It is this ‘window dressing’ which downplays theory as it does not give theory the weight it deserves and suggests that IS is ‘weak theoretically’. Thus IS papers that do stress theory should deepen IS theory rather than simply ‘add to the mass’.

As we stated in our original paper, all these concerns are not about appropriate emphasis on theory, but about the danger of inappropriate emphasis or inappropriate use of theory or theoretical frameworks. We therefore argued for (and provided examples of) some papers being ‘theory light’ where theory plays (or pretends to play) no significant part in the paper and the contribution lies elsewhere.

We are particularly concerned that too few papers published in the top journals of our discipline impact practice. Articles published are often posteriori interpretations of cases or datasets and the connections between academic IS researchers and practitioners remain too limited and uncertain. For this reason we have been particularly keen to promote the use of action research (Avison et al. 2016).

Our paper has had the impact to lead, for example, to six rich commentaries published in the same issue of *Journal of Information Technology*, but it has also sometimes been misinterpreted and misrepresented. For that reason we now emphasize what we did not say! For example:

1. We did not argue for a theoretical or theory-free research. This suggests an anti-theoretical stance that we do not share. We argue for papers to be accepted in our top journals that either make an excellent theoretical contribution or that make an excellent contribution elsewhere.
2. Our position is not the same as that of a grounded theorist who might start from a tentative theory-free stance but when making sense of the data is expected to create theory. Therefore papers based on the grounded theory approach are expected to discuss theoretical contributions of the research.
3. We did not argue that theory should not be a key element of doctoral studies. Doctoral students should have a thorough grasp of theory. They need to demonstrate knowledge and use of theory as part of their qualification.
4. We did not suggest that ‘anything goes’ in ‘theory light’ papers. Indeed, we suggested that authors and reviewers ask themselves ten questions which might apply to all qualitative papers, but are especially important in ‘theory light’ papers. These questions are: (1) Is it interesting? (2) Is it original? (3) Is it rigorous?
(4) Is it authentic? (5) Is it plausible? (6) Does it show criticality? (7) Is there access to the original data? (8) Is the approach appropriate? (9) Is it done well? (10) Is it timely? Again, each of these questions is discussed in the paper.

5. We do not regard writing ‘theory light’ papers to be easier to research or write, nor did we imply a less rigorous reviewing process, a lowering of standards for our leading journals, or an easier read. On the contrary, responding positively to our ten questions above suggests that these contributions need to be especially good ones.

The acid test for any paper (including ‘theory light’ ones) is the following high barrier: Is it probable that the paper will stimulate future research that will substantially alter IS theory and/or practice? Following this path we should see more papers in our leading journals that are truly original, challenging, and exciting, and less – dare we say – formulaic.

Dr. David Avison
Dr. Julien Malaurent
ESSEC Business School

3 Towards a Coherent View on Information Systems

Scientists have odious manners, except when you prop up their theory; then you can borrow money of them. – Mark Twain

3.1 Business Informatics as an Academic Field of Inquiry

Talking about theories depends on the underlying notion of theory. First, I would like to point out that academic fields of inquiry have developed very different understandings of what science and an acceptable theory are. It is impossible to give a complete overview of all answers. However, I would like to open the discourse and make some important preliminary remarks.

Table 1 shows four triples of corresponding words in English, French and German. This synopsis clearly shows that for the English word “science” different terms are used in German and French (McCloskey 1984). This fact is of major importance because it makes indisputably clear that the terms “science” and “Wissenschaft” are not interchangeable in all sentences without altering the truth value of statements. Hence, speakers from different language communities, particularly from English and German speaking ones, have different conceptions in mind when talking about science or Wissenschaft. According to (McCloskey 1984, p. 97), while in German and French the science word “merely means ‘disciplined inquiry,’ as distinct from... journalism or common sense”, in English, the “august word connotes of numbers, laboratory coats, and decisive experiments publicly observed”. In fact, whenever German speakers use the term “Wissenschaft” in the sense of Geisteswissenschaft or Ingenieurwissenschaft, English speakers do not use the term “science” at all.

Therefore, if we talk about Information Systems or Business and Information Systems Engineering (BISE) as a science, our understanding of science has to be clarified. While Information Systems is strongly rooted in science, BISE has its origin in engineering. In the following, I use the term “Business Informatics” – in analogy to Bioinformatics or Health Informatics – as an umbrella term for Information Systems and BISE. Table 2 summarizes the foci of different academic disciplines studying information systems.

3.2 What is a Theory in Business Informatics?

Analyzing the usage of the term “theory” in different communities is one approach to answer the question what a theory is in Business Informatics. Table 3 aggregates the results of two quantitative literature reviews conducted by Lim et al. (2009) (with a focus on Information Systems) and Houy et al. (2014) (with a focus on BISE).

| Table 2 Focus of different academic disciplines studying information systems |
|-----------------------------|-------------|--------|-----------------------------|
|                           | Natural sciences | Social sciences | Humanities | Engineering |
| Information systems        |              |             |              |              |
| Business and Information Systems Engineering |
| Business informatics        |              |             |              |              |

Table 1 Synopsis of terms denoting academic fields of inquiry in different languages (based on McCloskey 1984)

<table>
<thead>
<tr>
<th></th>
<th>English</th>
<th>French</th>
<th>German</th>
</tr>
</thead>
<tbody>
<tr>
<td>Natural sciences</td>
<td></td>
<td>Les sciences naturelles</td>
<td>Die Naturwissenschaften</td>
</tr>
<tr>
<td>Social sciences</td>
<td></td>
<td>Les sciences sociales</td>
<td>Die Sozialwissenschaften</td>
</tr>
<tr>
<td>Humanities</td>
<td></td>
<td>Les sciences humaines</td>
<td>Die Geisteswissenschaften</td>
</tr>
<tr>
<td>Engineering</td>
<td></td>
<td>L’ingénierie</td>
<td>Die Ingenieurwissenschaften</td>
</tr>
</tbody>
</table>
These results show:

- Pluralistic orientation: Table 3 only depicts the most cited theories in Business Informatics, in total, more than 200 theories were identified. This result shows that there exists no clear and distinct theoretical research paradigm in the sense of Kuhn (1996). Although there are some competing theories (e.g., resource-based view versus market-based view), most theories have different application areas and can be seen as complementary.

- Theory as an umbrella term: Sometimes the term “theory” is used as an umbrella term for different theoretical approaches, e.g., organization theory, decision theory or systems theory include very different theoretical approaches.

- Different reference disciplines: Theories used in Business Informatics are rooted in different academic fields of inquiry, e.g., microeconomics (game theory), strategic management (resource-based view), or organizational sciences (organizational theory).

- Mathematical and empirical theories: Some theories have an empirical content, e.g., transaction cost theory. The empirical content of other theories is debatable, e.g., systems theory or game theory. Other theories, e.g., graph theory, do not have any empirical content at all.

- Descriptive and normative theories: The term “theory” is used in a descriptive as well as a normative sense. For instance, it is well-known that decision theory has two different branches, normative/rational decision theory and descriptive decision theory.

Although such quantitative literature analyses can give important and interesting insights into the usage of the term “theory” in Business Informatics, it is also clear that such results should be critically reflected: (1) The presented analysis is based on the premise that a theory is present wherever the term “theory” is used. Although the idea that the meaning of a word is given by its usage is appealing, it should be remarked that it would be a classical logical fallacy to derive a normative notion of what a theory is solely from a descriptive analysis. (2) Since the term “theory” is used very differently, it is prima facie plausible that there exists not only one conception of the idea “theory”. My following contribution relies on the premise that the term “theory” can be explicated differently.

### Table 3: Most cited theories in Business Informatics

<table>
<thead>
<tr>
<th>Theory</th>
<th>Lim et al. (2009)</th>
<th>Houy et al. (2014)</th>
<th>Ranking points</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technology acceptance model</td>
<td>1</td>
<td>3</td>
<td>2.0</td>
</tr>
<tr>
<td>Game theory</td>
<td>4</td>
<td>1</td>
<td>2.5</td>
</tr>
<tr>
<td>Transaction cost theory</td>
<td>5</td>
<td>2</td>
<td>3.5</td>
</tr>
<tr>
<td>Resource-based view</td>
<td>2</td>
<td>6</td>
<td>4.0</td>
</tr>
<tr>
<td>Systems theory</td>
<td>–</td>
<td>4</td>
<td>4.0</td>
</tr>
<tr>
<td>Organizational theory</td>
<td>–</td>
<td>5</td>
<td>5.0</td>
</tr>
<tr>
<td>Diffusion of innovations</td>
<td>6</td>
<td>9</td>
<td>7.5</td>
</tr>
<tr>
<td>Graph theory</td>
<td>–</td>
<td>8</td>
<td>8.0</td>
</tr>
<tr>
<td>Theory of planned behavior</td>
<td>6</td>
<td>11</td>
<td>8.5</td>
</tr>
<tr>
<td>Theory of reasoned action</td>
<td>3</td>
<td>18</td>
<td>10.5</td>
</tr>
<tr>
<td>Decision theory</td>
<td>16</td>
<td>6</td>
<td>11.0</td>
</tr>
<tr>
<td>Principal agent theory</td>
<td>21</td>
<td>10</td>
<td>15.5</td>
</tr>
<tr>
<td>Organizational learning theory</td>
<td>8</td>
<td>28</td>
<td>18.0</td>
</tr>
<tr>
<td>Social cognitive theory</td>
<td>10</td>
<td>44</td>
<td>27.0</td>
</tr>
<tr>
<td>Dynamic capabilities</td>
<td>8</td>
<td>89</td>
<td>48.5</td>
</tr>
</tbody>
</table>

3.3 Two Major Design Theories in Business Informatics

The analysis above shows that design theories are clearly underrepresented in the top Business Informatics theories (Gregor 2006). However, it cannot be concluded from this result that there are no important design theories in Business Informatics. Note that there are many important theories in other branches of academic inquiry which do not carry the term “theory” in their name, e.g., geometry, thermodynamics or evolution. In fact, some very important research results in Business Informatics are not labeled as theory at all. Let me introduce two examples which have major influence within the German Business Informatics community:

- Model of Integrated Information Systems (IIV) developed by Mertens (2012): The work on this model started in the late 1960s and was further developed for more than 40 years. This model shows how different application systems in the manufacturing industry are conceptually integrated.
• Architecture of Integrated Information Systems (ARIS) developed by Scheer (1994): Scheer developed the ARIS as an instrument to systematize different aspects to describe and develop information systems. For each aspect and layer particular instruments are introduced and integrated. This model was developed in the late 1980s and is still used in different versions.

Although both works can easily be criticized for several reasons (e.g., bias towards manufacturing industry, not every construction step is explicated), the mentioned examples are two major instances of design theories. This is not merely my opinion; the statement can easily be substantiated by taking a look on the history of these contributions (work of Mertens developed up to the 18th edition, Scheer’s major work on ARIS is translated into English, Chinese, Russian and other languages). There are numerous examples of dissertations and research articles which are based on the design theories developed by Mertens and Scheer, although the literature analysis shows that they are not explicitly labeled as theory. Furthermore, at many German-speaking universities, these works provide the classical textbook for an introductory course into Business Informatics.

To summarize, although both design theories are not explicitly called “theories” and therefore do not appear in the above mentioned literature analysis, it would be a mistake not to subsume this work under the umbrella term “(design) theories” of Business Informatics.

3.4 Theoretical Progress: A Multi-Perspective Understanding of Theory

At large, there are good arguments to question the idea of scientific progress in general (Kuhn 1996). However, when understanding academic inquiry as a problem solving activity by following a particular research paradigm, I think it is possible to see some important developments which can be called progress. With respect to different research traditions, such a progress can have very different roots and epistemic qualities (Hacking 1983). Figure 1 provides an overview of four main perspectives.

• Business Informatics as mathematics: From the perspective of mathematics, the formal structure of information systems is of major importance. Empirical insights are out of scope of this perspective. As a primary method, a formal proof is used. Progress is achieved by formalizing general ideas and proving interesting statements. Example: Seminal paper by Kindler (2006) introduces and formalizes a framework for formal execution semantics for Event-driven Process Chains (EPC). The significant progress of this work is a mathematically sound definition of the non-local behavior of EPC.

• Business Informatics as a science: Real phenomena are described, explained, understood and often generalized by using a theory about these phenomena. Experiments are the scientific method par excellence. From this perspective, there are different areas for improvement, mainly a theoretical progress [finding a new theory explaining a phenomenon], an empirical progress (identifying or describing a (new) phenomenon] and a methodological progress (improving an existent or inventing a new method). Example: Seminal Paper by Davis (1989) explaining the acceptance of information technology. Davis shows that perceived ease of use and perceived usefulness are high predictors for user acceptance of information technology (theoretical progress). Additionally, he develops and validates measurement instruments for all introduced constructs (methodological progress).

• Business Informatics as engineering: New, more powerful and astonishing information technologies are created in academic or industrial laboratories and ultimately tested in reality. Research and development respectively prototyping are primary research methods. Example: Seminal work by Scheer (1994) on the Architecture of Information Systems (ARIS). The significant contribution of Scheer’s work is a comprehensive framework for describing and developing business information systems. Furthermore, a powerful software package was developed which demonstrates the feasibility and usefulness of this innovative approach. The experiences with this prototype provides the foundation for the development of the ARIS Platform which later became the market-leading system for business process management.

• Business Informatics as a philosophy: Developing new ideas and perspectives and criticizing well-known approaches is important for the philosophy of information systems. Speculation, discourse, analysis,
argument and debate are the major elements of methods used from this perspective. Example: Wand and Weber (1988) present the idea of using ontology as a foundation of information systems research and set the philosophical starting point and foundation for a broad research stream (Fettke 2006). Another example on the meta-level of research on Business Informatics is the seminal work by Hevner et al. (2004) who explicitly discuss the importance of design science research in information systems. Both works mentioned offer very fresh and fruitful views for and on research in Business Informatics. The significant contribution of Wand and Weber is a completely new fundament for conducting research. Hevner et al. introduce clear guidelines for conducting design science.

Again, I would like to point out that the different perspectives often stress different aspects of progress. However, the ultimate goal is to provide a coherent view on information systems. Identified contradictions in practice or theory are an important sign of a lack of coherence and call for more research. Furthermore, different perspectives on information systems have to be integrated. Such an integration provides a richer picture of how information systems are, can be, or should be.

As I stated before, different academic fields of inquiry have developed different understandings of what a theory is. However, I would like to mention that there exists a standard view on theory in the philosophy of science, which I would like to discuss in more detail in the following.

3.5 A Narrower View on Theory: The Standard View in Philosophy of Science

If you talk about what a theory is, there are of course different answers to this question (Fettke and Loos 2004). In the broadest sense, a theory is the result of an academic inquiry. As such it can be understood as justified true beliefs which are framed and often specifically named. However, the term “theory” is often used in a narrower sense. For example, compare the five theory types described by Gregor (2006), namely theory for: (I) analysis, (II) explaining, (III) predicting, (IV) explaining and predicting and (V) design & action.

Compared to the concept of theory introduced by Gregor, the standard view of philosophy of science is much narrower (Bunge 1998b; Ladyman 2001). According to the standard view, a theory is a cumulative point of scientific endeavor. A theory is a hypothetical-deductive system which contains presumptions and at least one scientific law statement covering a cause–effect-relationship (formalized as $A \rightarrow B$). The Euclidian geometry theory was for a long time the ideal formulation of a theory. However, in the meantime it is well known that Euclid’s geometry does not fit together with the real world, other geometry theories have been developed. Furthermore, Newtonian mechanics is an example of another theory in this sense. However, we know that this theory is still successfully applied in everyday reasoning, although it is not correct when very large velocities or very big masses are involved. Under this assumption, relativity theory must be used for correct reasoning.

From my point of view, there are good reasons to identify cause–effect-relationships at the core of an academic discipline or theory (Note that this statement is not a contradiction to my preliminary remarks as long as you accept the unproblematic premise that there are different conceptions of what theory is.). However, as an application-oriented discipline, solely quarrying for cause–effect-relationships is not sufficient. Business Informatics should not only be interested in cause–effect-relationships, but should also research means–end-relationships (Bunge 1998a; Chmielewicz 1994; Zelewski 1995).

3.6 The Importance and Foundation of Technological Rules

Business Informatics investigates information systems. Such investigations aim at representing and explaining existing information systems. According to the standard view of theory, a scientific law constitutes the core of a theory. In contrast, an application-oriented discipline such as Business Informatics is not only interested in scientific laws but in technological rules [formally: “B per A!”], (Bunge 1998a; Maaß and Storey 2015). In other words, Business Informatics works on new, possible information systems [Frank (2006); Müller (1990), p. 8]. Two design types can be distinguished. First, a new system can be described (“to-be system”). Although not every time explicitly mentioned, the modus of description is: “It is possible that ...”. Such a description represents an information system as it could or should be. Second, a new process can be described (“to-be process”). A planned process describes an action plan of how a possible system can be implemented or how an objective can be achieved.

Technological rules do not represent existing systems; they guide the development of new information systems. It is impossible to assign truth values to statements about possible systems by comparing the stated possibility with actual reality. Instead, one can only ask whether it is possible to implement or to realize such designs or whether it is desirable to make a planned system reality.

Typical examples for technological rules are (Fettke 2008):
The most important question is how such technological rules can be justified. Or, more generally: What is the interdependence between theories (in the narrower sense) and technological rules?

Often, from the perspective of pure science, it is argued that engineering is only an application of such law statements. Although some renowned proponents, e.g., Popper (1957), formulate the idea that theories can easily be transformed into technological rules by so-called tautological transformations, I believe the interrelationship between both concepts is much more complex (Houy et al. 2010, 2015). For example, the following aspects must be taken into account: (1) “Man has known how to make children without having the remotest idea about the reproduction process” (Bunge 1998a, p. 143). (2) Theories are sometimes still used for design purposes even when it is widely accepted that they are not true, e.g., Newtonian mechanics is still used for the calculating satellite orbits. (3) Not every law statement can effectively be used by a technological law statement, e.g., if one has no means to make the antecedent of the law true, it is impossible to use the law by a simple tautological transformation. Nevertheless, knowing the law might be useful for technological purposes. (4) Particularly in Business Informatics it is questionable whether all known empirically identified patterns or regularities qualify as causal relationships. For example, it is debatable whether the construct “perceived ease-of-use” of the Technology Acceptance Model has a causal effect on system acceptance. (5) Social systems engineers have to deal with self-fulfilling or self-defeating predictions.

To conclude, from an application-oriented perspective it does make sense to conduct academic inquires which are not theory-grounded (in the narrower sense) but practically successful.

### 3.7 On the Quality of Theories in Business Informatics

Lack of cumulative research, following short-lived fads and missing long-term, ambitious research goals are well-known shortcomings of our field which many others have criticized before (Hirschheim and Klein 2003; Steininger et al. 2009). Instead of repeating these still relevant deficits, I would like to put more emphasis on another aspect.

In his contribution to this discussion, Dirk Horvoka already referenced Kuhn’s concept of the disciplinary matrix which constitutes not only the identity of discipline but also the values of a research community. In other words, it is interesting to have a more detailed look on our disciplinary matrix in order to elaborate on the quality of theories in our field.

The textbooks of a discipline are one important factor constituting the disciplinary matrix. First, textbooks are major sources for introducing students to a field and demonstrating what is well-known and well-accepted in that discipline. Second, textbooks are also useful for practitioners as points of references to most significant results. Metaphorically speaking, they are symbols for the body of knowledge of a discipline.

A few years ago, some colleagues conducted a detailed analysis of Business Informatics textbooks and obtained remarkable and thought-provoking results (Frank and Lange 2004; Schauer and Strecker 2007). I do not want to recapitulate and update this analysis here. Instead, I would like to pose the following question: How do our textbooks deal with theories?

Without conducting a detailed analysis of how theories are referenced and described in our textbooks, I conjecture that the theories mentioned before do not play a central role in these introductory texts. This might have different reasons, e.g., it might take some time until a theory that is newly introduced by a major research outlet is included in a textbook.

As said before, there are also well-established theories in Business Informatics (e.g., Technology Acceptance Model and the two design theories by Mertens and Scheer mentioned above). I know there are some textbooks which adequately cover these theories. However, other textbooks do not describe or even mention these well-known theories at all. What can be the reason for this omission?

If we exclude the explanation that these textbooks do not represent the disciplinary matrix adequately, one explanation may be that the authors of these textbooks do not identify the mentioned theories as part of the disciplinary matrix of our discipline. If my assumption is true, then it can be concluded that our disciplinary matrix is not coherent anymore, but might be cracked.

### 3.8 Conclusion

When discussing what theory is and its role in academic inquiry, it must be clear that different fields of inquiry have very different answers to these questions. From the wider perspective of scientific progress, it can be argued that this situation can be harmful but also very productive. However, it is necessary that different fields of knowledge create a coherent view of what information systems are.

According to the standard view of theory in philosophy of science, a theory is a set of statements with at least one
nomological law. Such statements are of major importance for the understanding and design of information systems. Although there are some candidates for such statements in the context of Business Informatics, it is clear that there are very few examples which are able to constitute the core of our discipline. However, there exist well-known examples for (design) theories which can be seen as the core of Business Informatics.

In the future, it is necessary to develop a more coherent picture of different approaches to information systems. I propose to distinguish between two types of approaches, namely black box and white box theorizing. In a black box approach, technology is viewed as a black box whose inner components are invisible to the theory; they are abstracted. Typical examples for black box theories are the Technology Acceptance Model or studies on success factors of ERP systems. Such an approach to theorizing has its strengths. It provides a higher level of abstraction because the concrete implementation is not regarded as important for the theory. Furthermore, the complexity of real information systems is effectively reduced.

However, black box approaches are established on the premise that technology is simply given. Such approaches are blind with respect to design decisions inside the black box, which might have a huge impact on theorizing about it. Per definitionem, they do not generate knowledge about the inner structure and functions of technology. What our discipline needs are more white box theories providing a coherent view on information systems and its inner components.

Prof. Dr. Peter Fettke
German Research Center for Artificial Intelligence (DFKI)
and Saarland University

4 Science as Practice: Theory-as-Discourse

4.1 Introduction

When Latour’s climate scientist explains why his own claims and not those of the climate-change deniers should be believed, he does not invoke theory or models. He does not summon explanatory power or predictive accuracy. Nor does he retreat to an argument about instruments or data or simulations. Rather he responds, “If people don’t trust the institution of science, we’re in serious trouble” (Latour 2013, p. 3). He appeals to the fragile and ill-defined institution that engages a specific form of discourse. It is the discourse of science this essay highlights, and the disciplinary context in which the concept of theory makes any sense at all.

The assertions that theory is the pinnacle of research (Gregor 2006; Straub 2009), that scientific knowledge is based on theories, and that the primary contribution to research is theory have become IS folklore and are only rarely contested [for examples see: Avison and Malaurant (2014), Hambrick (2007)]. The claim that “conflicting notions of theory and theoretical contribution, rather than sheer overemphasis on theory” (Markus 2014, p. 342) is the cause of problems for the field assumes that a unitary view of theory is desirable. Further, it obscures the differences among the discursive, material, and instrumental contexts in which theory makes sense. Many authors discuss theory as a thing-in-itself, as an isolated entity to be reified, bounded and celebrated above all else. This preoccupation diminishes the other disciplinary research contributions that are required for a theory to be cogent (Hovorka and Boell 2015). Certainly theory is important and requires attention, but it is critical to position our understanding of theory within the distinctive disciplinary contexts through which theory, as a discourse, is created, critiqued, evolved, and adjudicated.

Through historical analysis, Kuhn captures this discourse in his original sense of paradigm, a term he subsequently abandoned for the broader concept of disciplinary matrix. This matrix is composed, at least in part, of symbolic generalizations, models, exemplars, instruments, and values (e.g., precision, prediction, generalizability, design). While Kuhn acknowledges that the list is incomplete, its components illustrate some of the shared commitments of a scientific practice.

It is noteworthy that in Kuhn’s extensive writing theory is not prioritized as a defining component of disciplinary integrity or legitimacy. Instead, disciplines are characterized by their paradigm or disciplinary matrix. The primary meaning of paradigm (and a key component of the disciplinary matrix) is the exemplar: the texts, teaching cases, and narratives which “contain not only the key theories and laws, but also...the applications of those theories in the solution of important problems, along with the new experimental or mathematical techniques (such as the chemical balance in Traité élémentaire de chimie and the calculus in Principia Mathematica) employed in those applications.” (Bird 2011). Theory and models are important but not “king” or the primary contribution of research. The elevation of theory as the premier contribution in scientific practice and the basis of knowledge misrepresents the role of theory in the broader discourse of scientific inquiry.

In Kuhn’s normal science, scientists are occupied with matching facts and observations to extant theory, and with articulating what is implicit with theory. Scientists must “premise current theory as the rules of the game. His objective is to solve a puzzle... at which others have failed
and current theory is required to define the puzzle...” (Kuhn 1965). Theory becomes fixed as a reified entity used to solve specific problems. Discussion in IS frequently focuses on the normal science image of theory as a reified object with essential characteristics. But during revolutionary science, in which the fundamentals of a disciplinary matrix change, Kuhn reveals fluidity in the conception of theory among practicing scientists who share the same commitments. The interpretation of a theory and even what it means to be a theory, is subject to situated contestation and revision.

Kuhn’s normal-revolutionary science distinction reveals that there is no clean separation of a theory from the disciplinary matrix, the discourse, in which it is embedded. As communities develop and change, theories are contested, supported/rejected, critiqued, expanded or simplified. Accounts of revolutionary science reveal an image of contestation, where ontological perspectives, theories of instruments and measurement, observations, ideas, things, marks, practices, and truth vie for recognition.

From this we can see that theory cannot be cleanly separated from the discourse regarding observations, instruments, measurements, methods, and the values by which scientific activity is evaluated. Every theory is a discourse composed of the individual papers which, taken together, present argumentation for a specific account of a phenomenon. This account is only understood by the community based on disciplinary matrix which the community shares and within which the theory is grounded.

The introduction to this special section and some of the contributors acknowledge that IS, BISE, informatics, management science, and other specializations are overlapping, yet distinctive, fields of inquiry. As new research communities and subspecialties proliferate over time (e.g., Big Data, Q-BISE, DSR) there will perforce be many theory discourses between and within disciplines. Within each community, what counts as factual, as a construct, as valid, or as explanatory also changes. The set of publications, conference talks, teaching materials – the discourse – becomes an intellectual space where ideas clash. The theory-as-discourse is an area defined by what we know, but it is also a zone of contestation, not of revolution, but of ideas competing against each other to disclose what worlds are created by theory.

The consequence of conceptualizing theory as an ongoing discursive-instrumental argument rather than a category used to include/exclude specific instances is that there is no essential characteristic form or function of theory. One of Kuhn’s central contributions was the recognition that practicing scientists do not follow a set of rules that enable coordinated research activity. Rather, the shared disciplinary matrix of each community is exhibited in the exemplars used to enroll researchers into the practice. Theory and models are only a part of the community’s exemplars and are embedded in the discourse in each community. Thus theory-as-discourse takes on a multiplicity of forms and functions including:

- An aspiration – what we wish we knew.
- A condensation – what we think we know.
- A compounding – (nothing accumulates in an unaltered form).
- A guide – what is worthy of our time.
- A value – what is worthy of knowing.

4.2 Reflections on this Special Section

The variety in conceptions of theory as exhibited in this special section evidence the primary argument I have put forward. In summary, different intellectual communities articulate theory in a variety of ways. Theory is viewed: (a) as a law-like cause-effect relationship that may be used to develop practical technological rules (Thalheim, in this section), (b) as a set of models, which are themselves simplified abstractions of reality (Kraemer and Schnurr, in this section), and (c) as a foundation for specific domain-oriented sub-disciplines (Suhl, in this section). There is some agreement among these papers that theory differs among disciplines (Avison and Malaurent as well as Fettke in this section). In addition, Mueller (in this section) notes the relationship between different onto-epistemologies that disclose different phenomenon, and the theorizing that identifies and accounts for those phenomenon. For example, the phenomenon of IS use, which is grounded in a Cartesian separation of user and object (Weber 2012), is de-centered in a non-dualist ontology (Barad 1996; Riemer and Johnston 2012).

These different conceptions do not present a compelling argument that IS/BISE and design communities should search for a unifying conceptual ground upon which to construct “theory for everyone,” or for an integrated conception of theory across communities. Rather they evidence the position argued in this essay that different conceptions of theory are not only inevitable, but are essential, for the different communities within IS/BISE, design and engineering to progress. It is not possible or desirable to reconcile or to integrate the many descriptions of theory such that every science community would agree on a single set of normative criteria. For example, IS is composed of multiple intellectual communities (Larsen et al. 2008). These communities have differing goals and values, and their different ontological foundations disclose different phenomena. Some communities in IS and BISE focus on explaining and predicting known phenomena. Recognizing the multiple forms and interpretations of...
instruments struggle to make things work (Pickering 1992). Revealing a realm in which the researcher and their material phenomena themselves resist and push back, values are challenged, supported critiqued, and evolve. The instruments, symbolic generalizations, models and ground upon which communities adjudicate the quality of other reconfiguration to overcome resistance (Pickering 1999) renders Gregor’s (2006) theory types equivocal in that the development and assessment of explanatory or predictive theories differs depending on the specific form of explanation or prediction implicated in the theory discourse. IS design- and engineering-oriented communities are more like architectural practice (Lee 1991) in their focus on creating new realities and emergent phenomenon rather than retrospective explanation or specific future predictions. But they are different practices and consider theory quite differently. In each community the resulting theory-as-discourse has different criteria for development, for contribution, for progress, and for adjudication of quality. In some communities, increasing the absolute accuracy of prediction is valuable. In other communities, increasing the business utility of prediction indicates progress. For some the creation of novel or problem-solving artifacts constitutes contribution and intellectual progress. But often progress can only be judged in retrospect as technologies or new processes derived from scientific inquiry come to dominate the landscape. Broadly, there are multiple distinctions for progress, including increasing correspondence of representations to observed phenomenon, of coherence of a set of beliefs held to be true, and of pragmatism. These adjudications further illustrate the inevitability of different theory discourses within and among the IS/BISE and design communities as each community enacts theory-as-discourse in relation to its own shared commitments to knowing the world.

A flexible and many-valent theory-as-discourse does not lead to arbitrary or relativistic conceptions of theory. The instrumental and discursive theory-as-discourse proposed here is implied by Pickering’s “mangle” (Pickering 1995) and by the “motley of science” of Hacking (1992). The dialectic of resistance and accommodation in scientific practice provides severe criteria for objectivity at both community and individual levels. These may include demands for falsifiability, avoidance of post-hoc and ad-hoc modifications, and the preference for theory which predicts new phenomenon over theories that explain what is already known (Pickering 1995). These, and other shared commitments of the institution of science are the backdrop upon which communities adjudicate the quality of each theory-as-discourse. As scientific practice is enacted, the instruments, symbolic generalizations, models and values are challenged, supported critiqued, and evolve. The material phenomena themselves resist and push back, revealing a realm in which the researcher and their instruments struggle to make things work (Pickering 1992). Material reality resists capture by experiments, denies measurement, and confounds instruments. Accommodation occurs when researchers enact conceptual, instrumental or other reconfiguration to overcome resistance (Pickering 1995). The dialectic of resistance and accommodation thus results in further changes in the theory discourse. When material resistance becomes extreme, a theory-as-discourse will longer elaborate “a distinct realm of facts, phenomena, and understandings of the world” (Pickering 1995, p. 202), and it is abandoned. For example, Wegener’s theory of continental drift (Wegener 1966), first published in 1915, was dismissed as being eccentric, footloose, preposterous, and improbable. But new instruments (e.g., sonar, magnetometers), disclosure of new phenomenon (e.g., ocean ridges and trenches, earthquake zones), new theory (e.g., sea-floor spreading, magnetic field reversal), and new models (e.g., continental drift; lithosphere dynamics) entered the theory-as-discourse resulting in the abandonment of contracting-earth theory and the broad acceptance of Plate Tectonics – albeit 50 years later.

The theoretical discourse culminating in Plate Tectonics illustrates that the phenomenon itself changed as symbolic generalizations, instruments, models and new exemplars become part of the disciplinary matrix. It is only within this discourse, in its entirety, that Plate Tectonics theory makes the world comprehensible. Theory-as-discourse acknowledges the variety of contributions composing a community’s disciplinary matrix and contextualizes the social-political-material-discursive practice of scientific institutions. This position liberates us from an unresolvable debate on what theory is or should be. In rejuvenating the discussion of the full spectrum of potential research contributions which constitute a disciplinary matrix, we may restore theory to an appropriate position and regain confidence in the institution of science itself.

Dr. Dirk Hovorka
University of Sydney

5 Microeconomically Founded Information Systems Research

5.1 Introduction

It is our fundamental understanding that the main purpose of IS research, like most other research disciplines, should be the development of robust theories, which can then inform us about the likely answers to our research questions. What is notable, although not unique about IS research is that the research questions we pursue are not only concerned with the understanding, explanation and possibly prediction of real world phenomena, but also with how we can shape the institutions (North 1991; Roth 2002) that govern these phenomena in order to achieve a certain goal (cf. Gregor 2006). In this regard, IS research takes a theory-guided engineering perspective.
Consider the domain of electronic markets, for example. IS research may be interested in why an observed (e.g., technology induced) market behavior occurs, which market outcomes are likely under a given scenario, but also how markets should be designed in order to achieve a desirable outcome.

In the following we will develop and discuss what we call an idealized microeconomically founded IS research process cycle, depicted in Fig. 2, which reflects our view that fruitful IS theories can be built upon formal, analytic models. Such models are in turn founded upon both, stylized facts that are derived from empirical regularities observed in reality, as well as the existing body of knowledge stemming from robust theories. With reality, we denote the object and processes of investigation that research intents to describe or understand. Scientific inquiries are either concerned with realizations of the past or with potential future states. Researchers perceive reality through empirical observation and data gathering, which is naturally constrained and imperfect. Models, which in themselves are the foundation of theory, can then be used to explain, predict and design instances of the real world. Finally, models, and thus also theory, are evaluated and refined with respect to their ability to inform us about past or future real world phenomena. This can be achieved in field or laboratory studies either by validating or falsifying theory-guided hypotheses, comparing a theory’s predictions with actual future outcomes or by evaluating the success of theory-informed design proposals and engineering approaches in actual applications.

The herein described research paradigm is more specific than (but not contradictory to) more general IS research paradigms (cf. Frank 2006), such as design science (cf., e.g., Hevner et al. 2004). Nevertheless, we will argue that theories developed under this framework are suitable to pursue all four fundamental goals of IS research, namely analysis, explanation, prediction, and prescription/design (cf. Gregor 2006). It is not our intention, however, to evaluate or judge different IS research approaches, but rather to motivate why we believe that the proposed microeconomically founded research paradigm is one of several appropriate means to rigorously develop relevant IS theories.

5.2 The Building Blocks of Microeconomically Founded Theory Development

5.2.1 Theory as a Set of Models

In general, theory has been characterized as the “basic aim of science” (Kerlinger 1986, p. 8) and is often referred to as “the answer to queries of why” (Kaplan and Merton cited by Sutton and Staw (1995), p. 378). According to (Weick 2005, p. 396) a theory may be measured in its success to “explain, predict, and delight”. In explaining our precise understanding of “theory”, we start from the premise that the main task of theory is the integration of findings of individual studies into a modular, but coherent body of knowledge that connects research agendas based on a shared terminology and which provides a microfoundation. Revision and extension of theory is achieved in iterative steps through new or modified models that may either re-investigate central assumptions, thus deepening theory’s microfoundation, or create meta-

![Fig. 2 Idealized microeconomically founded IS research process cycle](image-url)
models by further abstraction based on the existing body of knowledge. By this means, a microfounded theory serves as an anchor (Dasgupta 2002) and provides building blocks for new research projects and further theory-building.

In our view, robust theories are the result of deduction and induction from a host of formal models. Therefore, theory can be viewed as a classified set or series of models (Morgan and Knuuttila 2012). In philosophy of science this integral role of models as a part of the structure of theory has been supported by the Semantic View and has been further emphasized by the Pragmatic View (Winther 2015). Consequently, a clear distinction between theory and its models is difficult in general, and even more so if the analysis of theoretical models is deemed as the central part of scientific activity.

At the extreme, a single model can already be the foundation of a theory, although probably not a very robust one. In this regard, the understanding of a robust theory in the social sciences may differ from the understanding of a robust theory in the natural sciences, because theory in the social sciences can be very context dependent, as subjectivity of decision makers, i.e., their beliefs, information, and view of the world substantially shape their choices and actions (Hausman 2013). For example, (Dasgupta 2002, p. 63) noted that “the physicist, Steven Weinberg, once remarked that when you have ‘seen’ one electron, you have seen them all. [...] When you have observed one transaction, you have not observed them all. More tellingly, when you have met one human being, you have by no means met them all”. This is why a robust theory in the social sciences should regularly be built upon a set of models, each of which takes a different perspective on a particular issue and explores a slightly different set of assumptions, such that the boundaries of the theory become transparent.

5.2.2 Models as the Mediator Between Theory and Reality

This understanding of theory shifts our attention to the development of suitable models. Models as idealizations (Morgan and Knuuttila 2012) serve as representations of reality that are obtained by simplification, abstraction (see, e.g., the work of Cartwright 2005; Hausman 1990) and/or isolation (Mäki 1992, 2012). But they may also be created as pure constructions, i.e., exaggerated caricatures (Gibbard and Varian 1978), fictional constructs (Sugden 2000), or heuristic devices that “mimic [...] some stylized features of the real system” (Morgan and Knuuttila 2012, p. 64). Gilboa et al. (2014) suggested that economic models serve as analogies that allow for case-based reasoning and contribute to the body of knowledge through inductive inference rather than through deductive, rule-based reasoning. We advocate the use of formal, analytic models in this context, because such models allow to make the assumptions transparent that may lead to a proposition and possibly a normative statement upon which a robust theory, and ultimately a robust explanation or prediction can be built. Note that mathematical formalization is a sufficient, but not a necessary prerequisite to develop a formal model, because it allows to precisely formulate its subject domain, making it an “exact science” (Griesemer 2013, p. 299). Moreover, (Dasgupta 2002, p. 70f.) argued that in building a theory “prior intuition is often of little help. That is why mathematical modeling has proved to be indispensable”. The analytic approach provides researchers with a toolbox to deal with especially hard and complex problems. By the means of logical verification, propositions can be shown to be internally true with regard to the underlying assumption.

In general, the goal of a model is to “capture only those core causal factors, capacities or the essentials of a causal mechanism that bring about a certain target phenomenon” (Morgan and Knuuttila 2012, p. 53). Such an abstraction is the prerequisite for conducting a deductive analysis within a particular scenario of interest. What we consider to be particularly important in order to develop relevant models is that a model’s microfoundation should contain elements of both theory and reality. On the one hand, a model’s assumptions should reflect stylized empirical facts that are well grounded in observed empirical regularities or relevant future scenarios. Such empirical facts can be derived directly from gathered data (most likely with measurement error), may already be the result of extended data analysis, e.g., in the form of detected patterns or correlations, or may be identified by means of a literature review (Houy et al. 2015). However, stylized empirical facts need not (yet) be supported by any theory. This enables us also to incorporate insights of theory-free empirical analysis [particularly (big) data analytics or machine learning] into formal models, which may then lead to a theory that can explain the empirical regularities. On the other hand, a model’s assumptions may also be derived from the existing body of knowledge, i.e., from theory. This exemplifies the dual view on the relationship between models and theory: Although models are used to advance theory, theory is also used to produce and inform models.

1 In this context, it is worth mentioning that although data analytics may be able to predict what will happen in a specific context, similar to a theory, it is still theory-free, because it is generally not able to explain why it happens. Without theory, however, it must remain unknown whether these predictions can be generalized and to what extent they are robust to other application scenarios. Therefore, data analytics differs from the traditional paradigm of empirical analysis, which centers around the falsification or validation of hypotheses, which again requires a theory (although not necessarily in the same sense as proposed here – see, e.g., Diesing (2008) for a more elaborate discussion of the relationship between empirical and formal theory) from which these hypotheses are derived in the first place.
A main line of attack against analytic models is to argue that they are not realistic and thus, model-driven theory is useless, because there is nothing to learn about reality. This criticism is amplified in the field of social science, where models are context dependent, as argued above. This naive understanding, however, falls short. First, as we have just mentioned, good models should be grounded in stylized empirical facts. Second, there is an inherent trade-off between accuracy and generality, achieved through simplicity (Gilboa et al. 2014). Scholars experienced in the domain of modeling generally agree on the fact, that too much complexity in fact impedes the explanatory power and the interpretability of models. For example, (Schwab et al. 2011, p. 1115) stated that in order “to formulate useful generalizations, researchers need to focus on the most fundamental, pervasive, and inertial causal relations. To guide human action, researchers need to develop parsimonious, and simple models that humans understand”. In the words of (Lucas 1980, p. 697) “a 'good' model [...] will not be exactly more ‘real’ than a poor one, but will provide better imitations”. In this context, the statistician George Box coined the famous phrase that “all models are wrong, but some are useful” (Box 1979, p. 2), clarifying that a model must inherently be unrealistic in a dogmatic sense (see Mäki 2012 for a discussion), but that models in fact enable us to understand real phenomena by abstracting from the complexity of reality. To exemplify this, (Robinson 1962, p. 33) argued that “a model which took account of all the variegation of reality would be of no more use than a map at the scale of one to one”. Of course, an interesting model must also exceed a pure tautology, i.e., the results that can be deduced from its assumptions are usually not a priori clear, but may represent surprising results (Koopmans 1957; Morgan and Knuuttila 2012). This requirement can be paraphrased by a quote that is supposedly due to Einstein: “Everything should be made as simple as possible, but not simpler”.

Furthermore, we wish to emphasize that over and beyond the explanatory function of formal models, the modeling process itself may prove to exhibit value for understanding a particular scenario. Moreover, a model is an instrument to express an individuals’ perception of a problem and may therefore serve as a communication device. (Gibbard and Varian 1978, p. 669) stated that “perhaps, it is initially unclear what is to be explained, and a model provides a means of formulation”.

5.2.3 Empirical Analyses as the Means to Evaluate Theory

According to our theory-centric research view, empirical analysis serves two core functions: (1) As described above, empirical analysis is a means to derive stylized facts in order to motivate model assumptions, or likewise, to evaluate the plausibility of proposed assumptions. (2) As will be described next, empirical analysis is also a means to evaluate the quality of a theory as a whole. In the context of IS research, we conceive three main ways in which evaluation of theory can be done.

First, empirical analysis, foremost field and laboratory studies, can be employed in order to falsify [in the spirit of Lakatos and Popper (Hausman 2013; Backhouse 2012)], and more ambitiously to validate, theoretically derived hypotheses. While field studies have the advantage of high external validity, they can be generally challenged on the premises that it is difficult to establish causal effects due to problems of (unobserved) confounding variables and endogeneity. At a fundamental level, this gives rise to doubts whether empirical observations are able to falsify (a fortiori validate) theory at all. These concerns are magnified due to the context-specific nature of field studies and a lack of control over the environment that encompasses investigations. Laboratory experiments may be able to mitigate some of these concerns through systematic variation of treatment conditions, randomization of subjects and augmented control of the researcher. Based on a high internal validity, although at the cost of lack of external validity, isolation of causal relationships is facilitated and falsification of theoretical propositions is more easily justifiable (Guala 2005). Furthermore, laboratory experiments facilitate the process of de-idealization (Morgan and Knuuttila 2012), i.e., the generalization of the model context beyond its well-defined assumptions by successively relaxing the assumptions until the theory’s established hypotheses begin to break down. Ultimately, however, laboratory and field studies are complementary means to a similar end.

Second, empirical analysis can evaluate the accuracy of theory-driven predictions over time. Although hypotheses may also be regarded as model predictions, the focus here lies less on falsification of suggested causal relationships, but more on the correct qualitative assessment of the impact of future scenarios. With regard to its ability to predict future states of reality [in the sense of Friedman 1953], a microfounded theory draws from its ability to explain observations at the macro level, based on an understanding of the underlying mechanisms and the necessary conditions. By this means, theory-driven predictions are likely to be more robust to changes of real systems as underlying causes can be identified and theory can be modified accordingly (Dasgupta 2002). Moreover, formal analysis allows for experimentation and evaluation of counterfactuals. Two remarks should be made in this context: First, it must be noted that there exists an inherent trade-off between a theory’s simplicity and its predictive accuracy. While a simple model or theory may apply more generally and is able to make more robust qualitative
predictions, it will also almost certainly be too simple to make accurate quantitative predictions. In turn, the reverse holds true for complex models. This is akin to what is known as the bias-variance-trade-off in statistics (cf. Hastie et al. 2009). Second, even if a theory’s prediction may be accurate, this does not “prove” in a deductive sense that it is valid. We may only apply what is known as abductive inference here, that is we can infer that a theory was sufficient to predict the phenomenon of interest, but not that it was necessary, i.e., the only possible theory to be sufficient.

Third, and possibly most interesting in the context of IS research, empirical studies can serve as a testbed for theory-driven design proposals. In this context, laboratory experiments can be seen as an intermediate economic engineering step, similar to a wind tunnel in traditional engineering, where the design proposals (e.g., a proposed market design or regulatory institution) can be evaluated under idealized conditions that mirror those assumptions under which the theory was developed. If the proposed design performs well (relative to the intended goal) in the laboratory then it should be taken to the field for further evaluation. If, however, the proposed design already fails to perform in the laboratory, then there is little reason to believe that it would perform well in the field (Plott 1987). Consequently, the design, and most probably also the underlying theory, would need revision already at this stage.

5.3 Conclusions

Recently, several scholars in the fields of management (Locke 2007; Hambrick 2007) and IS (Avison and Malaurant 2014), among others, have criticized excessive adherence to theory and argue that a scientific contribution can also be made without the need for theory. While we are sympathetic with this view, we strongly believe that the development of robust theories is at the core of scientific endeavor. However, we also believe that these models and theories should be both, (1) well grounded in stylized empirical facts that are the result of inductive research efforts, as well as (2) evaluated and refined through empirical analyses based on field studies and laboratory experiments. To this end, we have motivated and discussed a microeconomically founded IS research paradigm that we deem suitable to develop theories in our field that are rigorous and relevant. In this spirit, we deem the long term goal of microeconomically founded IS research to be the development of robust and stable theories that have been developed and refined through several repetitions of the depicted research process cycle.

Prof. Dr. Jan Krämer  
Daniel Schurr, M.Sc.  
Universität Passau

6 Theory in the Age of Post-Adoption

6.1 Introduction

To put first things first: I think of theory and theorizing as the key task of any science and feel that our discipline’s attention is increasingly shifting in that direction. This is evidenced by seminal contributions (e.g., Burton-Jones et al. 2015; Gregor 2006; Weber 2012), special sections in key journals (e.g., MISQ and JAIS), and dedicated conference tracks (esp. at ICIS, ECIS, and HICSS). In my opinion, this is a welcome shift from methods to theories – or from how to what we research – that brings a dormant discussion to the center stage: what is theory?

This shift also comes with controversy: While I personally don’t agree to the “theory fetish” Avison and Malaurant (2014) diagnose, I think they do our discipline a great service by recognizing this discussion. However, I believe that this issue’s editorial points in the right direction when it refers to Markus’ (2014, p. 342) observation that “conflicting notions of theory and theoretical contribution, rather than sheer overemphasis on theory, may lie at the heart of the problem […]” In light of an increasing recognition of the debate about what theory is, it comes as no surprise that Becker et al. (2015) find that “rethinking the theoretical foundations of the IS discipline” is among the top three grand challenges in our discipline’s future development – both in terms of relevance and impact.

6.2 The Field of Post-Adoption

One arena I believe this challenge to be particularly true for is post-adoption. As a response to criticism of simple models of technology adoption, the post-adoption research community is shaping up to develop more elaborate models for what happens across multiple levels once technology starts to interact with individuals’ actions and larger organizational, market, and societal structures. The resultant research opportunities resonate with the German Informatics Society’s grand challenge of omnipresent human-computer interaction, and socio-technical issues, broadly speaking, are among the key issues in the BISE community as well (Becker et al. 2015). Outside of academia, post-adoption research comes at a time when many organizations are thinking about how to engage in digital transformation in order to leverage modern information and communication technologies.

Of course, this is not a new issue. Its roots date back to the 1970s (esp. Bostrom and Heinen 1977a, b) and beyond (e.g., Emery and Trist 1960; Woodward 1958). Recently, however, post-adoption research has mainly been characterized by an intense ontological and epistemological
debate and a resultant fragmentation of its results – that is, its theories.

(1) Which conception of theory is central to your area of research?

The two main contestants in this debate come with different conceptions of theory: those advocating ontological separability of social and material aspects on one side, and those promoting ontological inseparability on the other (Mueller et al. 2012). Recently, these camps have begun to rally under new banners such as “critical realism” versus “agential realism” (Leonardi 2013) or “weak socio-materiality” versus “strong sociomateriality” (Jones 2014) respectively.

While the interested reader can find more elaborate explanations of these camps in Leonardi (2013) and Jones (2014), the camps’ assumptions about ontology and epistemology are central to the debate on theory. On the one hand, the separability camp subscribes to a realist ontology and a mostly representational epistemology. For them, material and social aspects exist independently of any actor and the theorist’s key job is to determine which is which and how they interact once they meet in practice. Works by Mutch (2010, 2013) and Mingers (2000) – who strongly draw on Bhaskar (1979) – investigate how such a paradigmatic setup can facilitate the study of technology in social systems, and papers by Burton-Jones and Grange (2013) or Volkoff, Strong, and colleagues (e.g., Strong and Volkoff 2010; Volkoff et al. 2007) deliver excellent exemplars of how this philosophical position helps develop theoretical models of post-adoptions mechanisms and processes.

On the other hand, the inseparability camp grants ontological equality of all entities involved in a phenomenon. These entities, however, do not depend on any objective reality nor are they an attribute of human (or, for that matter, non-human) agency. They rather emerge within entanglements through material-discursive practices. Such an entanglement, or phenomenon, is the ontological entity that is sociomaterial. This means that, ontologically speaking, all phenomena are inseparably social and material and that any attempt to separate the two is an arbitrary decision by an agent – be it an actor in one of our studies or the researcher. This stresses a deviation from the representational epistemology discussed above and suggests a shift towards performative (and diffractive) thinking. This camp, rooted in Barad’s (2003) work, was made popular in IS by Orlikowski and Scott (esp. Orlikowski 2010; Orlikowski and Scott 2008) and studies by Scott and Orlikowski (2014) themselves and by Schultze (2011) illustrate the tenets of this paradigmatic position and its conception of theory.

While my own thinking increasingly gravitates towards the realist position (e.g., Lauterbach et al. 2014) – mainly because I find respective field studies easier to design – I strongly believe that both positions should not be seen as fundamentally irreconcilable opposites. Rather, I would like to think that there is a level beyond the current discussion on which we could explore how insights from these two perspectives complement each other. However, the (seeming) opposition between the camps creates a key challenge to this (perhaps naive) belief: Are the mostly positivistic conceptions of theory and theorizing still useful (let alone valid) in the neo-positivist world of the critical realists or in the non-positivist world of the agential realists or – particularly – in a world that seeks to move beyond their distinction?

(2) How evaluate progress in your field? What is a long-term goal?

It is this challenge that also drives progress: For the last five years, progress in this domain is probably best described by the emergence of new theoretical perspectives and our discipline’s increasing command of the underlying paradigmatic positions. While the former is evidenced by a growing number of studies employing some form of sociomaterial thinking (e.g., Hultin and Mähring 2014; Introna and Hayes 2011; Johri 2011; Jones 2014), the latter is underlined by the various attempts to better structure the debate’s philosophical roots (e.g., Jones 2014; Leonardi 2013).

However, a challenge I see in this is the fact that many seem to have been motivated by some instance of paradigmatic inconvenience to develop an own variant of the ontological and epistemological foundations. Looking at the larger body of sociomaterial studies published recently, irreconcilable differences seem to hamper our discipline’s ability to integrate and synthesize theoretical findings I argued for above – an essential prerequisite for the development of a cumulative tradition and a competition of theories to retain the most powerful explanations (Weick 1989).

Consequently, a long-term goal I think worthy of exploration is to turn away from a theory for every one towards a theory for everyone – even if we may have to stop calling it theory then. That is, carefully discussing if and how paradigmatic differences influence our findings, what we mean when we talk of theory, and our ability to compare, contrast, and combine insights into the interplay of technology, social structures, and individuals’ behaviors. Hovorka (this section) makes an excellent observation when he points out that the different communities involved in such an integration effort will likely also realize differences in what they mean by theory and how they judge its progress and quality. Nevertheless, I feel that this plurality of perspectives still gravitates around the interplay of technology, social structures, and individuals’ behaviors as a common phenomenon. Wouldn’t it thus seem logical to
try to learn from each other? To me, this thought resonates with the debate between Avison and Malaurant (2014) and Markus (2014) as much as it seems to be on Barad’s (2003) mind. Also, a debate seeking to transcend philosophical differences seems a promising approach to not simply reproduce the philosophical discussions from outside the BISE community, but to actually contribute to advancing these debates – a concern for this domain that can be traced back as far as Williams’ and Edge’s (1996) seminal paper. Consequently, in order to help the post-adoption domain and its theories grow, revisiting paradigmatic assumptions to explore options for complementarity of findings is an essential prerequisite for integrating and consolidating our various findings towards a shared understanding.

(3) How is theory guiding design and engineering and how does it impact practice?

While much of the debate in this field might seem esoteric, I see three important links between this paradigmatic debate and practice. First, I believe that our research in this domain enables managers to better express their experiences. This is inspired by a steering committee meeting I attended three years ago in which I pitched the post-adoption research my team and I intended to do to a potential host company. While the team and I expected that the philosophical aspects might be ill-matched to the audience, the participating executives quickly adopted the concepts presented to them and retold their experiences in this newfound language. The ensuing discussion allowed them to make sense of each other’s experiences, pinpoint problems, and devise solutions – and resulted in exciting insights for research.

Second, I see important links to the design and engineering of future systems. Insights from this domain of IS research are beginning to shed light on how people interact with technology, make sense of it, and transform what they do through it (e.g., Burton-Jones and Grange 2013; Liang et al. 2015) as well as on how we design the projects that introduce these technologies (e.g., Strong et al. 2014; Wagner et al. 2010). While not yet prominent, some IS research hints towards this research’s impact on how we design technologies and their interfaces, particularly when recognizing material properties and their impact on resultant practices (e.g., Jones 2014; Leonardi 2012). I like the thought Brynjolfsson and McAfee (2014) introduce: Increasingly, we will have to think of technology and how we design it not (only) as a potential replacement for human work, but as a meaningful augmentation that complements human work. This will lead to new forms of technology and interface design just as much as to new patterns of interaction between humans and technology. In the long run, this understanding will inform the development of truly intelligent and self-adapting technologies.

Third, on a more abstract but all the more important level, better understanding of what technology is, how we relate to it, and how it shapes our lives also has an ethical dimension. While underexplored in our field thus far, technology is in the process of fundamentally reshaping our life and how we live it.

Taking these three together, advanced sensemaking and expression will allow for expanded description, analysis, and explanation of the interplay of technology, social structures, and individuals’ behaviors. Such an improved understanding of post-adoption research’s key phenomenon will transform technologies, behaviors, and social structures. Thus there seems to be nothing quite so practical as a sound understanding of what technology means for us, how we relate to it, and how it influences our behaviors; all of which needs to ground on a sound paradigmatic understanding of the theories we develop to help explain these issues.

(4) How do you evaluate the quality of theories in your field?

Much like elsewhere, the basic evaluation of theories in the post-adoption field is conducted through a social process towards consensus among a panel of reviewers, editors, and authors. The key tenet of this process to me, especially for conceptual pieces mostly focused on theory and theorizing, is to see if a new theory proposed succeeds in convincing peers. To this end, its power to transform our thinking is one of the key aspects I believe to be important in new theoretical contributions. This resonates strongly with DiMaggio’s (1995) idea of theory as narrative with a touch of enlightenment as well as with my own steering committee experience I shared above.

As such, the question of whether a new theoretical perspective helps to make sense of things we observe in practice, but cannot quite explain so far, seems like a key aspect of a theory’s quality. For this, Popper (1980) develops the metaphor of theories as “[...] nets cast to catch what we call ‘the world’; to rationalize, to explain and to master it” (p. 59). Again, DiMaggio (1995) offers a brilliant perspective on theory as being constructed “post hoc,” which to me suggests that many theories might best not be evaluated by any quantitative indicator, but by their potential to inspire and transform thinking.

This also alerts us to the fact that no theory should be looked at in isolation. Beyond any one single theory alone, a good theory also engages in a detailed discussion of rivalry explanations, boundary spanning constructs, and its own boundaries. While often neglected in complex manuscripts already pressured for space, this engagement with what else we know is essential to link any theoretical insight back to the larger discourse and its attempt to build a cumulative core of knowledge on the phenomenon we study. Based on own experiences (e.g., Mueller and Raeth
2012). I particularly appreciate multi-paradigmatic and multi-theoretical work that consciously compares and contrasts what we can see from one perspective with what we would see from another. In the long run, such comparative working will contribute to what Weick (1989) calls disciplined imagination, that is, theorizing as a process of variation, selection, and retention.

Of course the ability to do so depends on understanding the underlying paradigmatic assumptions and on being willing to focus on commonalities and overlaps rather than differences. Above, I hinted towards my belief that the post-adoption community is not yet at a point where such a synthesis is possible. The last five years rather seem to inspire the metaphor of the “Tower of Babel” instead of letting us hope for the coming of a “Babelfish” for theories and insights (as borrowed from Douglas Adams’ bestselling “Hitchhiker’s Guide to the Galaxy” series).

6.3 Challenges on the Way Ahead

In the next five years, however, I am confident that this domain will witness a tremendous discussion and – hopefully – advance of theory and theorizing. Regardless of which of the above mentioned camps researchers subscribe to, both will likely be united in their quest for post-positivist theories; neo-positivist, realist scholars on one side and non-positivist scholars on the other. This will come with a shift away from the conceptual monopoly positivistic, representational constructions of theory have held in the discourse so far. In fact, the upcoming working conference of the IFIP working group 8.2 to be held this December just before ICIS has set out to explore “new encounters with technology and organization” that go “beyond Interpretivism” (from the call for papers) and I am excited to see what this will produce.

Future debates like this will have to address a wide spectrum of issues: from the redefinition of basic theory taxonomy (e.g., is the term “construct” also applicable to describe theories that do not follow a realist ontology and a representational epistemology?) to quite practical concerns (e.g., means of representation; Gregor 2006). This will also lead to an intense debate on what theory really is and new quality criteria that theories have to live up to, preferably also across paradigmatic positions (see, e.g., Burton-Jones et al. 2015 or Lee 2014 for notable early contributions). Reading Hovorka’s contribution to this section, I feel that the post-adoption community is on the brink of realizing and discussing its theories-as-discourses – both in terms of their contents (immediate theories) as well as on a philosophical level (meta-theoretical considerations). While the current fragmentation of these discourses seems to hamper the integration of our various understandings of the post-adoption phenomenon, its heterogeneity must not be seen as something evil per se. Quite to the contrary, I join Scott and Orlikowski (2013) in appreciating the plurality of current studies and also think that Lyttinen and King (2004) make an excellent point when they advocate plurality as a driver of innovation that makes sure that a discipline stays current and maintains a reasonable level of plasticity to adapt to changes in the phenomena it studies.

At the end of the day, all research in this domain strives to better understand the interplay (or intraplay) of technology, social structures, and individuals’ behaviors. In the years ahead, I personally hope that the focus will not only be on the content (i.e., the theory itself), but also on two equally important aspects: First, the meaning of theory – or what comes beyond theory – in order to help integrate what we learn about post-adoption. Second, the process of theorizing in order to help aspiring theorist – like myself – hone the skills and crafts of writing and reasoning that are theorizing.

Dr. Benjamin Müller
University of Groningen

7 Business and Decision Analytics in BISE: How much Theory do we Need?

As a scientific discipline, BISE is based on a theoretical foundation that includes different theories depending on the focus and perspective of a given subcommunity. The BISE subcommunity, due to its focus on analytical methods and decision support systems, uses quantitative methods to build and analyze descriptive, predictive and prescriptive models that support decision makers in practice. Here we use the term “Business and Decision Analytics” for this subarea. The quantitative methods draw from a rich theoretical basis in mathematics, statistics, computer science, and operations research, among others. It is not a main goal of BISE researchers to develop new theories in mathematics or operations research, but they need understanding of theory in order to be able to select a right solution approach for each problem and research task. As a generalization and abstraction, new theoretical findings can be established based on BISE research in this area.

Theories in statistics, artificial intelligence, and data modeling form the basis of business and decision analytics, and researchers develop new models and methods to analyze data and compute various indicators to guide business decisions. Mathematics, algorithm theory, and software engineering are important to guide business analysts and software developers in building optimization systems to compute optimal or near-optimal solutions for complex decision problems in business applications.
The models that represent decision problems from practice tend to be quite large and difficult, so that solution methods are needed which can cope with large models and can scale these according to the needs from practice. Knowledge of complexity theory helps researchers to classify algorithmic solution methods and be able to judge their suitability for a given decision problem. It is not a main goal of a BISE researcher to prove worst-case complexity of an algorithm, but rather to assess which methods are able to generate best possible solutions that can be realized in practice with today’s technologies.

Fuzzy set theory or alternative uncertainty theories, including stochasticity, can be the basis for modeling approaches with respect to preference elicitation and optimization, when the data available is uncertain. Discrete event simulation traditionally uses stochastic distributions to model uncertain data. Decision theory can be used as a basis for designing systems for multicriteria decision support. Some decision support approaches can be built using game theory to represent autonomous actors in agent-based systems.

Modeling is a very important step in developing solutions for decision situations. The best modeling approach should be selected based on the structure and goals of the decision problem. Optimization models, simulation models, data mining models and multicriteria decision models, among others, have their own application areas, and each modeling technology requires a certain structure of the decision problem. A unified modeling theory is still missing and would be helpful for selecting a suitable modeling approach (see Thalheim, in this section).

A main challenge the business and decision analytics subcommunity faces today is the increasing complexity of decisions in the progressively dynamic environment of today’s business, especially in supply, manufacturing and service networks (see Fink et al. 2015; Mertens et al. 2015). The increasing interaction of various entities in complex business networks is not yet well understood. Simultaneously today’s powerful information technology allows for the use of large amounts of structured digital data for decision-making. “Big data” together with cloud technologies provide much more opportunities to analyze and generate supporting information for decision makers than has been realized until now.

A main research goal of the business and decision analytics subcommunity is to develop new models, methods and systems to be able to model and analyze the complex networks and interactions of their entities. New approaches are needed that include uncertainties and consider robustness aspects, thus providing support to help practitioners improve decision making. To achieve this goal, an interdisciplinary approach is necessary. We need expertise in modeling, algorithms, software engineering, and business theories.

Long-time research goal of the business and decision analytics subcommunity is thus to develop and improve models and methods that help to understand and analyze the dynamic environment of today’s business. Evaluation of research progress should therefore assess to what extent new decision models cover relevant areas in business that have not been fully understood until now, as well as how good the methods are which have been proposed to solve and analyze the models. The models and methods developed should be evaluated considering problem structure and needs from the business world, and the same should be done simultaneously with the scientific state-of-the-art and relevant theory. The natural goal is thus to combine rigor and relevance and to produce relevant research results on a high level of scientific rigor.

An expert in research and/or practice of business and decision analytics needs interdisciplinary skills and usually combines knowledge of several disciplines such as information systems, mathematical models and methods, business processes, computer science, software engineering, and data science with decision support techniques. In these disciplines theories have been developed that build a theoretical foundation and thus establish the discipline as a scientific research area. Some of the relevant theories are domain-specific and focus on a given application domain, such as ERP, revenue management or recommender systems, and others are of general nature, such as graph theory or complexity theory.

Besides theoretical knowledge, a business and decision analytics professional needs awareness of all competences necessary to complete modeling and system development projects that provide support for business decision makers and processes. Typically, the following competencies are needed:

- To understand the domain and the specific decision problem.
- To select a suitable modeling approach: simulation, optimization, MCDM, data analysis etc.
- To set up a correct model, combining domain knowledge with modeling knowledge and experience.
- To select the right solution approach, its implementation, and configuration.
- If necessary, to develop and test new solution methods.
- To integrate new quantitative models into an existing business information system, incl. design of database interfaces, user interfaces, communication networks, etc.
- To interpret the solution for the decision makers.

Typical textbooks for decision support systems and operations research contain most of the relevant areas (see for
ex. Turban et al. 2014), however, they mostly focus on methodical aspects and ignore many areas that are important from the information systems point of view.

The question arises whether the subarea business and decision analytics in BISE involves or needs its own theories, or if it is sufficient to be based on theories of neighboring disciplines, the combination and integration of which no doubt is a very challenging task in every single project. To my understanding it does not seem promising to try to develop one unified comprehensive theory for the complete subcommunity, it would simply be too multifaceted as well as constantly evolving and without sharp boundaries. Its basis would be many theories from the neighboring disciplines, and an expert should have an understanding of the most important ones and be able to combine various aspects of them in each single research and development project.

However, it might be possible and helpful to develop a classification or taxonomy of business and decision analytics that could be called a theory. Such a structured and comprehensive view (though not necessarily covering all aspects) would help to understand the area and to select the right approach and right methods for a given problem.

Individual researchers and practitioners have collected a lot of experience and established strict rules as well as heuristic thumb rules that help structuring certain decision problems, selecting the right models and methods, and embedding the system components into an existing IS environment. This knowledge and experience may build the basis for a theory in the sense of classification, taxonomy and/or rule system. Such a taxonomy would ideally involve aspects such as application areas, modeling and solving methods, decision support components, as well as integration into business information and communication systems (see Table 4).

A comprehensive taxonomy would be helpful in introducing the area to students and professionals and in communicating the concepts of business and decision analytics. In practice, many objects can be assigned to two and more classes. However, the classification would help assigning an object and selecting the right approach to solve a given business decision task.

Prof. Dr. Leena Suhl
University of Paderborn

8 Towards a Theory of (Conceptual) Models

8.1 Introduction

A theory is in general any systematic and coherent collection of ideas that relate to a specific subject. The notion of theory varies in dependence on scientific disciplines (Kondakov 1974; Seiffert and Radnitzky 1992; Thiel 2004).

1. A theory can be understood as a practice-oriented apprenticeship, as a counterpart of acting and of practice, as a systematic generalization of experience, and a system of main ideas.

2. A (scientific) theory is a “systematic ideational structure of broad scope, conceived by the human imagination, that encompasses a family of empirical (experiential) laws regarding regularities existing in objects and events, both observed and posited. A scientific theory is a structure suggested by these laws and is devised to explain them in a scientifically rational manner. In attempting to explain things and events, the scientist employs (1) careful observation or experiments, (2) reports of regularities, and (3)...
systematic explanatory schemes (theories).” (Bosco et al. 2015).

3. A theory can also be understood as an offer, i.e., a scientific, an explicit and systematic discussion of foundations and methods, with critical reflection, and as a system of assured conceptions providing a holistic understanding. Many scientific and engineering disciplines use this constructive understanding of the notion of theory. A constructive theory is a collection of settled instruction conceptions (e.g., concepts, rules, laws, conditions) for (system) development within practical (technical) and quality (esthetic) norms, according to the goals of construction, and guided by some background. A theory is understood as the underpinning of engineering similar to architecture theory (Semper 1851) and the approaches by Vitruvius and L. B. Alberti. Constructive theories in Computer Science and Business Informatics use as their sources four kinds of methods: systematic (deductive mathematical or inductive logical), engineering-oriented abductive or compositional, application-driven, and electronics-oriented component methods.

A theory in the third sense combines explicative and prognostic functions. It is applicative, explicate, exploitative, expiative, explorative, and implicative from the one side, and it is preindicating, prognosticative, and predictive from the other side. Gregor (2006) associates models with construction-oriented theories for the area of information systems. She distinguishes (1) theories for analyzing, (2) theories for explaining, (3) theories for predicting, (4) theories for explaining and predicting, and (5) theories for design and action. Her main attitude is, however, construction models for analysis, explanation, prediction, and construction.

8.2 Models – The Third Dimension of Science

Models are one of the – if not the – central elements of Computer Science and Business Informatics. The research in these disciplines considers models as artifacts that are constructed in a certain way and prepared for their utilization. Models might also be mental models and thought concepts. Models are used in utilization scenarios such as construction of systems, verification, optimization, explanation, and documentation. In these scenarios they function as instruments.

Given the utilization scenarios, we may use models as perception models, mental models, situation models, experimentation models, formal model, mathematical models, conceptual models, computational models, inspiration models, physical models, visualization models, representation models, diagrammatic models, exploration models, heuristic models, informative models, instructive models, etc. They are a means for some purpose (or better: function within a certain utilization scenario), are often volatile after having been used, are useful inside and often useless outside the utilization scenario.

8.2.1 Elements of a General Modeling Theory

A general theory of model should provide answers to questions such as: What is a model? What are its essential elements? Which kinds of models reflect which task and support a solution of which problems? Which methods must be provided for a proper use of the model? Which methods support development and modernization of models? In which cases is the model adequate? What are the limits and where should this model not be used? In which case we can rely on a model? What are good models? Which models are effective? Which properties can be proven for models? How can models be integrated and composed? What are the correct activities for modeling? What is the added value of a model? Who can use the model how? What are the background theories of modeling? Why should this model be used where it is used? In what way? And by what means?

A general modeling theory generalizes the variety of model notions. In this case language matters, e.g., it enables or disables. The theory allows for managing a complexity of models and methods. Model development methods and model utilization methods should be defined in a similar way as in natural sciences. The theory should also refer to good utilization stories and to best practices.

8.2.2 Models Within the Dichotomy of Theory and State of Affairs

Classical science and also Computer Science and Business Informatics consider models to reflect a certain state of affairs, a certain part of reality, or certain observations. They might also depict parts and pieces of a theory. So, models seem to be placed between the state of affairs and theories. Figure 3 shows the classical understanding of this dichotomy.

This two-dimensional reasoning seems, however, too simple. Models form a further and orthogonal means and are different from theories and also different from the state of affairs.

8.2.3 The Development of Sciences

Disciplines often use a combination of empirical research that mainly describes natural phenomena, of theory-

---

2 An instrument is among others (1) a means whereby something is achieved, performed, or furthered; (2) one used by another as a means or aid or tool (Safra et al. 2003).
oriented research that develops concept worlds, of computational research that simulates complex phenomena, and of data exploration research that unifies theory, experiment, and simulation (Gray 2007). Thus Fig. 4 distinguishes four generations of sciences.

Models are a main instrument in all four generations. Their function, however, is different as illustrated in Fig. 5.

8.2.4 Extending the Two-Dimension of the Dichotomy by a Third Dimension

The classical dichotomy of reality and theories should be extended by a third dimension. Theories explain the state of affairs. They are results of explorations of the reality. Models provide an understanding of a theory and illustrate the reality. For Computer Science and Business Informatics, the relationship is similar. We might, for instance, use schemata as models. The theory behind could be, for instance, a concept theory.

Models are therefore the third dimension of science (Thalheim and Nissen 2015a)3. Figure 6 depicts this understanding.

8.3 The Conception of the (Conceptual) Model

A model is a well-formed, adequate, and dependable instrument that represents origins.

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of

---

3 The title of the book (Chadarevian and Hopwood 2004) has inspired this observation.
practice within some context and correspond to the functions that a model fulfills in utilization scenarios.

The model should be well-formed according to specific well-formedness criteria. As an instrument or more specifically an artifact, a model comes with its background, e.g., with paradigms, assumptions, postulates, language, thought community, etc. The background is often given only in an implicit form.

A well-formed instrument is adequate for a collection of origins if it is analogous to the origins to be represented according to specific analogy criteria, it is more focused (e.g., simpler, truncated, more abstract or reduced) than the origins being modeled, and if it sufficiently satisfies its purpose.

Well-formedness enables an instrument to be justified by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through formulas, by falsifiability, and by stability and plasticity.

The instrument is sufficient by its quality characterization for internal quality, external quality and quality in use or through quality characteristics (Thalheim 2010) such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

A well-formed instrument is called dependable if it is sufficient and justified for some of the justification properties and some of the sufficiency characteristics.

### 8.3.1 Scenarios and Functions of a Model

Models function as an instrument in some usage scenarios and a given usage spectrum. Their function in these scenarios is a combination of functions such as explanation, optimization-variation, validation-verification-testing, reflection-optimization, exploration, hypothetical investigation, documentation-visualization, and description-prescription functions. The model functions effectively in some of the scenarios and less effectively in others. The function determines the purpose and the objective (or goal) of the model. Functioning of models is supported by methods. Such methods support tasks such as defining, constructing, exploring, communicating, understanding, replacing, substituting, documenting, negotiating, replacing, optimizing, validating, verifying, testing, reporting, and accounting. A model is effective if it can be deployed according to its objectives.

### 8.3.2 Conceptual Models

An information systems or database model is typically a schematic description of a system, theory, or phenomenon of an origin that accounts for known or inferred properties of the origin and may be used for further study of the origin’s characteristics.

Conceptual models are models enhanced by concepts and integrated into a space of conceptions\(^4\). Conceptual modeling is modeling with associations to concepts and conceptions. A conceptual model incorporates concepts into the model. Hence, Fig. 6 can now be revisited for this case and we arrive at Fig. 7.

### 8.3.3 Reasoning Theory within a Theory of Models

A general theory of reasoning must therefore cover many different aspects. We may structure these aspects by a pattern for specification of reasoning support for modeling acts or steps as follows (Thalheim 2011, 2012b, 2014; Thalheim and Nissen 2015b):

- the modeling acts with its specifics (Thalheim 2010);
- the foundation for the modeling acts with the theory that is going to support this act, the technics that can be used for the start, completion and for the support of the modeling act, and the reasoning techniques that can be applied for each step (Thalheim 2012a);
- the partner involved with their obligations, permissions, and restrictions, with their roles and rights, and with their play;
- the aspects that are under consideration for the current modeling acts;
- the consumed and produced elements of the instrument that are under consideration during work;
- the resources that must be obtained, that can be used or that are going to be modified during a modeling act.

Consider, for instance, the reasoning that aims at realization objectives. It includes specific facets such as

- to command, to require, to compel, and to make someone do something by means of supporting acts such as communicating, requesting, bespeaking, ordering, forbidding, prohibiting, interdicting, proscribing;
- to ask, to expect, to consider obligatory, to request and expect by means of specific supporting acts such as transmitting, communicating, calling for, demanding;
- to want, to need, to require by means of supporting acts of wanting, needing, requiring;

\(^4\) White (1994) distinguishes two different meanings of the word ‘concept’: (1) Concepts are general categories and thing of interest that are used for classification. Concepts thus have fuzzy boundaries. Additionally, classification depends on the context and deployment. (2) Concepts are all the knowledge that the person has, and associates with, the concept’s name. They are reasonable complete in terms of the business. Murphy (2001) and Thalheim (2007) define concepts in a more sophisticated form. According to White (1994), conceptions are systems of explanation.
• to necessitate, to ask, to postulate, to need, to take, to involve, to call for, to demand, to require as useful, to just, or to proper.

The reasoning that is geared towards operating, relevant properties, model objectives, the model itself, towards construction and assessment and guarantees can be characterized in a similar form.

8.4 Theories and (Conceptual) Models

Thalheim and Nissen (2015a) distinguish between ‘models’ (models as representations or artifacts), ‘to model’ (methods of model development and model utilization), and ‘modeling’ (systematic and well-founded matured model development and model utilization; abbreviated as MMM).

8.4.1 Art, Science, and Culture of Modeling

Art (in the broader sense, e.g., used in D.E. Knuth’s “Art of Programming”) is based on creative skills and imagination in the MMM community and produces models as instruments for an easy and simple way of utilization in given scenarios. It requires conscious development of well-formed models. It intends to be contemplated or appreciated as adequate and dependable. We claim that an MMM art has already been developed but is not yet compiled into a holistic body of knowledge.

However, engineering requires a creative application of scientific principles to the design or development and utilization of models, to forecast the effect of model application, and to effectively handle co-evolution of systems and models according to the function of models in utilization scenarios. It requires an MMM science and culture.

An MMM science additionally contains methodologies, matured guidelines for modeling practice, well-founded algorithms and methods for development and utilization of models beyond MMM theories. Culture is “a system of shared values, which distinguishes members of one group or category of people from those of another group; culture is therefore intrinsic in the mind of individuals and it can be measured” (Hofstede et al. 2010). An MMM culture is the collective programming of the mind in one MMM community of practice. It will be different in different areas of Computer Science and Business Informatics.

8.4.2 The MMM Theory as a Lacuna of CS and BI Research

Hartmann and Frigg (2014) consider models and modeling as one of the lacunas in modern research: “Models play an important role in science. But despite the fact that they have generated considerable interest among philosophers, there remain significant lacunas in our understanding of what models are and of how they work.” The book of Thalheim and Nissen (2015a) tries to close this gap on the basis of surveys of models, of approaches to the modeling activities, and of modeling in various sciences (archeology, arts, biology, business informatics, chemistry, computer science, economics, electrotechnics, environmental sciences, farming, geosciences, historical sciences, languages, marine science, mathematics, medicine, ocean sciences, pedagogical science, philosophy, philology, physics, political sciences, sociology, and sports). An MMM theory is still one of the difficult research topics in Computer Science and Business Informatics. The development of a settled conception of models is the first step. The next step is the treatment of modelling activities and of modeling. An MMM culture seems to constitute the task of the next decade.

Bernhard Thalheim
Christian-Albrechts-Universität zu Kiel

References


Bhaskar R (1979) The possibility of naturalism, 1st edn. Harvester, Sussex


Brooks FP (1975) The mythical man-month – essays on software engineering. Addison-Wesley, Reading


Bunge M (1998a) Philosophy of science – from explanation to justification, Rev edn. Transaction, New Brunswick

Bunge M (1998b) Philosophy of science – from problem to theory, Rev edn. Transaction, New Brunswick


Frank U (2006) Towards a pluralistic conception of research methods in information systems research. ICB Research Reports No. 7, Institute for Computer Science and Business Information Systems, University Duisburg-Essen, ISSN 1860-2770


Pickering A (1992) From science as knowledge to science as practice. Sci Prac Culture 4


Schultze U (2011) The avatar as sociomaterial entanglement: a performative perspective on identity, agency and world-making in virtual worlds. In: Galletta DF, Liang TP (eds) 32 international conference on information systems (ICIS 2011), Shanghai


Semper G (1851) Die vier Elemente der Baukunst. Braunschweig


A Conceptual Model for Services

Bernhard Thalheim\textsuperscript{1(✉)} and Ajantha Dahanayake\textsuperscript{2}

\textsuperscript{1} Department of Computer Science, Christian Albrechts University Kiel, 24098 Kiel, Germany
thalheim@is.informatik.uni-kiel.de

\textsuperscript{2} Department of Computer Information Science, Prince Sultan University, Riyadh, Kingdom of Saudi Arabia
adahanayake@pscw.psu.edu.sa

Abstract. Models are a mainstay of every scientific and engineering discipline. Models are typically more accessible to study than the systems. Models are instruments that are effectively functioning within a scenario. The effectiveness is based on an associated set of methods and satisfies requirements of usage of the model. A typical usage of a model is explanation, informed selection, and appropriation of an opportunity. This usage is declared through information and directions for usage or more specifically through an informative model in the case of a service model.

1 Services and the Conception of a Service

Today, the service has gained recognition as the more realistic concept for dealing with complexities of cross-disciplinary systems engineering extending its validity beyond the classical information systems design and development realm [4]. In this respect the service concept combines and integrates the value created in different design contexts such as person-to-person encounters, technology enabled self-service, computational services, multi-channel, multi-device, location-based and context-aware, and smart services [13]. Therefore, the service concept reveals the intrinsic design challenges of the information required to perform a service, and emphasizes the design choices that allocate the responsibility to provide this information between the service provider and service consumer.

1.1 Some Well-Known Service Notions

The service is being defined using different abstraction models with varying applications representing multitude of definitions of the service concept [7]. The increasing interests in services have introduced service concept’s abstraction into levels such as; business services, web services, software-as-a-service (SaaS), platform-as-a-services, and infrastructure-as-a-service [2]. Service architectures are proposed as means to methodically structure systems [1,5,16].

There are number of service notations available in the in the literature, and research has looked into the service mainly from two perspectives, (a) from the
low-level technological point of view and (b) from the higher abstract business point of view. These two categories of service descriptions have derived number of service notations. Some of those main stream service notations are:

The REA (Resource-Event-Agent) ontology [8,11] uses as core concepts resources, economic event, and agent. The RSS (Resource-Service-Systems) model [12] is an adaptation of REA ontology stressing that REA is a conceptual model of economic exchange and uses a Service-Dominant Logic (SDL) [21]. The model of the three perspectives of services uses abstraction, restriction, and co-creation. It concentrates on the use and offering of resources [2]. The perspectives addressed by this model are: service as a means for abstraction; service as means for providing restricted access to resources; and service as a means for co-creation of value. The logics behind is the Goods Dominant Logic (GDL) model [22].

Web service description languages concentrate on Service-Orientated Architectures (SOAs) for web service domain. Software systems are decomposed into independent collaborating units [14]. Named services interact with one another through message exchanges. The seven contexts of service design [6,9,13] combine person-to-person encounters, technology-enhanced encounters, self-service, computational services, multi-channel, multi-device, and location-based and context-aware services description.

1.2 The Explanation, Selection, and Appropriation

Explanation, understanding and informed selection of a tool is one of the main usage scenarios for a software models. People want to solve some problems. Services provide solutions to these problems and require a context, e.g. skills of people, an infrastructure, a specific way of work, a specific background, and a specific kind of collaboration. In order to select the right service, a model of the service is used as an instrument for explanation and quick shallow understanding which service might be a good candidate, what are the strengths and weaknesses of the service under consideration, which service meets the needs, and what are the opportunities and risks while deploying such a service.

The best and simplest instrument in such usage scenario is the instruction leaflet or more generally as a specification of the information and directions on the basis of the informative model. We shall show in the sequel that this model of a service extends the cargo dimension [10] to the general notion of the informative model. Such models of a service enable people in directed, purposeful, rewarding, realistic, and trackable deployment of a service within a given usage scenario, i.e. use according to the qualities of the model [4]. After informed selection of a service, it might be used in the creation of new work order based on the assimilation of the service into the given context, i.e. appropriation of the service.

1.3 Developing a Service Model Based on the W*H Frame

Systems are typically characterised by a combination of large information content with the need of different stakeholders to understand at least some system aspects. People need models to assist them in understanding the context of their
own work and the requirements on it. We concentrate in this paper on the support provided by models to understand how a system works, how it can be used or should not be used, and what would be the benefit of such a model. We illustrate this utilisation of models for services.

We develop a novel service model based on the W*H specification frame [4]. The W*H model [4] provides a high-level and conceptual reflection and reflects on the variety of aspects that separates concerns such as service as a product, service as an offer, service request, service delivery, service application, service record, service log or archive and also service exception, which allows and supports a general characterization of services by their ends, their stakeholders, their application domain, their purpose and their context.

2 The Notion of a Model

The theory of models is the body of knowledge that concerns with the fundamental nature, function, development and utilisation of models in science and engineering, e.g. in Computer Science. In its most general sense, a model is a proxy and is used to represent some system for a well-defined purpose. Changes in the structure and behaviour of a model are easier to implement, to isolate, to understand and to communicate to others. In this section we review the notion of the model that has been developed in [18–20].

2.1 Artifacts that Are Models

A model is a well-formed, adequate, and dependable artifact that represents origins. Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilisation scenarios.

The model should be well-formed according to some well-formedness criterion. As an instrument or more specifically an artefact a model comes with its background, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form. A model is used in a context such as discipline, a time, an infrastructure, and an application.

Models function as an instrument in some usage scenarios and a given usage spectrum. Their function in these scenarios is a combination of functions such as explanation, optimization-variation, validation-verification-testing, reflection-optimization, exploration, hypothetical investigation, documentation-visualisation, and description-prescription functions. The model functions effectively in some of the scenarios and less effectively in others. The function determines the purpose and the objective (or goal) of the model. Functioning of models is supported by methods. Such methods support tasks such as defining, constructing, exploring, communicating, understanding, replacing, substituting, documenting, negotiating, replacing, optimizing, validating, verifying, testing, reporting, and accounting. A model is effective if it can be deployed according to its objectives.
Models have several essential properties that qualify an artifact as a model. An well-formed artifact is adequate for a collection of origins if it is analogous to the origins to be represented according to some analogy criterion, it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its purpose.

Well-formedness enables an artifact to be justified by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through formulas, by falsifiability, and by stability and plasticity.

The artifact is sufficient by its quality characterisation for internal quality, external quality and quality in use or through quality characteristics [17] such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

A well-formed artifact is called dependable if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

2.2 Artifacts as Instruments in Some Usage Scenario

Models will be used, i.e. there is some usage scenario, some reason for its use, some goal and purpose for its usage and deployment, and finally some function that the model has to play in a given usage scenario. A typical usage scenario is problem solving. We first describe a problem, then specify the requirements for its solutions, focus on a context, describe the community of practices and more specifically the skills needed for the collaborative solution of the problem, and scope on those origins that must be considered. Next we develop a model and use this model as an instrument in the problem solving process. This instrument provides a utility for the solution of the problem. The solution developed within the model setting is then used for derivation of a solution for the given problem in the origin setting.

A similar use of models is given for models of services. Service models might be used for the development of a service system. They might be used for assessment of services, for optimisation and variation of services, for validation-verification-testing, for investigation, and for documentation-visualization. In this paper we concentrate on the explanation, informed selection, and appropriation use of a service model. It must provide a high level description of the service itself. This usage is typical for a process of determining whether a service is of high utility in an application. Such usage is based on specific usage pattern or more specifically on a special model that is the usage model of an instrument as a model.

2.3 Conceptional Modelling: Modelling Enhanced by Concepts

An information systems model is typically a schematic description of a system, theory, or phenomenon of an origin that accounts for known or inferred properties of the origin and may be used for further study of characteristics of the
origin. *Conceptional modelling*\(^1\) aims to create an abstract representation of the situation under investigation, or more precisely, the way users think about it. *Conceptual models* enhance models with concepts that are commonly shared within a community or at least within the community of practice in a given usage scenario. Concepts specify our knowledge what things are there and what properties things have. Their definition can be given in a narrative informal form, in a formal way, by reference to some other definitions, etc. We may use a large variety of semantics [15], e.g., lexical or ontological, logical, or reflective.

### 2.4 Adequacy and Dependability of Informative Models

Models are used in *explanation, informed selection, and appropriation* scenarios. We call such models *informative models*. Their main aim of is to inform the user according to his/her information demand and according to the profile and portfolio. The instrument steers and directs its users which are typically proactive. It supplies information that is desired or needed. Users may examine and check the content provided. Typical methods of such instruments are communication, orientation, combination, survey, and feedback methods.

Users have to get informed what is the issue that can be solved with the instrument, what are the main ingredients of the instrument and how they are used, what is the main background behind this instrument, and why they should use this instrument. They need a quick shallow understanding how simple, how meaningful, how adequate, how realistic, and how trackable is the instrument (*SMART*). They must be enabled to select the most appropriate instrument, i.e. they should know the strengths, weaknesses, opportunities, and threats of the given instrument (*SWOT*).

The SWOT and SMART evaluation is the basis for adequateness and dependability of informative models. The informative model must be analogous in structure and function to its origins. It is far simpler than the origin and thus more focussed. Its purpose is to explain the origin in such a way than a user can choose this instrument because of its properties since all demanded properties are satisfied. The selection and appropriation of an instrument by the user depends on the explanatory statement on the profile and the portfolio of the given instrument, on coherence to the typical norms and standards accepted by the community of practice, on a statement on applicability and added value of the instrument, and the relative stability of the description given. The instrument usage becomes then justified. Furthermore, the instrument must suffice the demands of such scenarios. The quality in use depends on understandability and parsimony of description, worthiness and eligibility of presented origins, and the added value

---

\(^1\) The words ‘conceptual’ and ‘conceptional’ are often considered to be synonyms. The word ‘conceptual’ is linked to concepts and conceptions. ‘Conceptual’ means that a thing - e.g. an instrument or artifact - is characterised by concepts or conceptions. The word ‘conceptional’ associates a thing as being or of the nature of a notion or concept. Conceptional modelling is modelling with associations to concepts. A conceptual model incorporates concepts into the model.
it has for the given utilisation scenarios. The external quality is mainly based on its required exactness and validation. The internal quality must support these qualities. The quality evaluation and the quality safeguard is an explicit statement of these qualities according to the usage scenarios, to the context, to the origins that are represented, and to community of practice.

2.5 The Cargo of a Model

The cargo of any instrument is typically a very general instrument insert like the package insert in pharmacy or an enclosed label. It describes the instrument, the main functions, the forbidden usages, the specific values of the instrument, and the context for the usage model. Following [10, 20] we describe the cargo by a description of the mission of the instrument in the usage scenarios, the determination of the instrument, an abstract declaration of the meaning of the instrument, and a narrative explanation of the identity of the instrument.

The mission of a model consists of functions (and anti-functions or forbidden ones) that the model has in different usage scenarios, the purposes of the usages of the model, and a description of the potential and of the capacity of the model. The determination contains the basic ideas, features, particularities, and the usage model of the given instrument. The meaning contains the main semantic and pragmatic statements about the model and describes the value of the instrument according to its functions in the usage scenarios, and the importance within the given settings. Each instrument has its identity, i.e. the actual or obvious identity, the communicated identity, the identity accepted in the community of practice, the ideal identity as a promise, and the desired identity in the eyes of the users of the instrument.

2.6 The Informative Model

The informative model consists of the cargo, the description of its adequacy and dependability, and the SMART and SWOT statements. It informs a potential user through bringing facts to somebody’s attention, provides these facts in an appropriate form according their information demand, guides them by steering and directing, and leads them by changing the information stage and level. Based on the informative model, the user selects the origin for usage with full informed consent or refuses to use it. It is similar to an instruction leaflet provided with instruments we use. The informative model is semantically characterized by: objectivity; functional information; official information; explanation; association to something in future; different representational media and presenters; degree of extraction from open to hidden; variety of styles such as short content description, long pertinent explanation, or long event-based description.

In the case of a service model, the informative model must state positively and in an understandable form what is the service, must describe what is the reward of a service, and must allow to reason about the rewards of the service, i.e. put the functions and purposes in a wider context (PURE). Informative models of a service are based on a presentation that is easy-to-survey and to understand,
that is given in the right formatting and form, that supports elaboration and
surveying, that avoids learning efforts for their users, that provides the inner
content semantics and its inner associations, that might be based on icons and
pictographs, and that presents the annotation and meta-information including
also ownership and usability.
We shall now explore in the sequel what are the ingredients of such informa-
tive instruments in the case of a service model.

3 Service Specifications

3.1 Scenarios and Functions of Service Specifications

To capture the scenarios and functions of service specification we introduce \( W^*H \)
model in Fig. 1 that is a novel conceptual model for service modelling.

<table>
<thead>
<tr>
<th>Service</th>
<th>Service Name</th>
<th>Wherefore?</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concept</td>
<td>Ends</td>
<td>Why?</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Where to?</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>For When?</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>For Which reason?</td>
<td></td>
</tr>
<tr>
<td>Content</td>
<td>Supporting means</td>
<td>Wherewith?</td>
<td>Application are</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Wherein?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Application case</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Wherefrom?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Problem</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>For What</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Organizational unit</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Where</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Triggering Event</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Whence</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>IT</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>What</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>How</td>
</tr>
<tr>
<td>Annotation</td>
<td>Source</td>
<td>Where of?</td>
<td>Party</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Supplier</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>By whom?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Consumer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>To whom?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Producer</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Whichever?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Activity</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Input</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>What in?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Output</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>What out?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Added Value</th>
<th>Surplus Value</th>
<th>Worthiness?</th>
<th>Context</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Systems Context</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Where at?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Story Context</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Where about?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Coexistence Context</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Wither?</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Time Context</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>When?</td>
</tr>
</tbody>
</table>

Fig. 1. The W*H Specification Frame for the Conceptual Model of a Service

The W*H model in Fig. 1 fulfills the conceptual definition of the service
color concept composing the need to serve the following purposes:
The composition of the W*H model consisting of content space, concept space, annotation space, and add value space as orthogonal dimensions that captures the fundamental elements for developing services.

It reflects number of aspects neglected in other service models, such as the handling of the service as a collection of offering, a proper annotation facility, a model to describe the service concept, and the specification of added value. It handles those requirements at the same time.

It helps capturing and organizing the discrete functions contained in (business) applications comprised of underlying business process or workflows into interoperable, (standards-based) services.

The model accommodates the services to be abstracted from implementations representing natural fundamental building blocks that can synchronize the functional requirements and IT implementations perspective.

It considers by definition that the services to be combined, evolved and/or reused quickly to meet business needs.

Finally, it represents an abstraction level independent of underlying technology.

In addition, the W*H model in Fig. 1 also serves the following purposes:

- The inquiry through simple and structured questions according to the primary dimension on wherefore, whereof, wherewith, and worthiness further leading to secondary and additional questions along the concept, annotation, content, add value or surplus value space that covers usefulness, usage, and usability requirements in totality.
- The powerful inquiring questions are a product of the conceptual underpinning of W*H grounded within the conceptional modelling tradition in the Concept-Content-Annotation triptych extended with the Added Value dimension and further integration and extension with the inquiry system of Hermagoras of Temnos frames.
- The W*H model is comprise of 24 questions in total that cover the complete spectrum of questions addressing the service description; (W5 + W4 + W10H+W4) and H stands for how.
- The models compactness helps to validate domain knowledge during solution modelling discussions with the stakeholders with high demanding work schedules.
- The comprehensibility of the W*H model became the main contributor to the understanding of the domain’s services and requirements.
- The model contributes as the primary input model leading to the IT-service systems projection on solution modelling.
- It contribute as the primary input model leading to the IT-service systems projection on the evaluations criteria of systems functioning on its trustworthiness, flexibility to change, and efficient manageability and maintainability.

3.2 Dimensions of Service Specification

The Content Dimension: Services as a Collection of Offerings. The service defines the what, how, and who on what basis of service innovation, design,
and development, and helps mediate between customer or consumer needs and an organization's strategic intent. When extended above the generalized business and technological abstraction levels, the content of the service concept composes the need to serve the following purposes:

- Fundamental elements for developing applications;
- Organizing the discrete functions contained in (business) applications comprised of underlying business process or workflows into interoperable, (standards-based) services;
- Services abstracted from implementations representing natural fundamental building blocks that can synchronize the functional requirements and IT implementations perspective;
- Services to be combined, evolved and/or reused quickly to meet business needs; Represent an abstraction level independent of underlying technology.

The abstraction of the notion of a service system within an organization's strategic intent emphasizes those purposes given above allow us to define the content description of services as a collection of offers that are given by companies, by vendors, by people and by automatic software tools [3]. Thus the content of a service system is a collection of service offerings.

The service offering reflects the supporting means in terms of with what means the service's content is represented in the application domain. It corresponds to identification and specification of the problem within an application area. The problem is a specific application case that resides with an organizational unit. Those problems are subject to events that produce triggers needing attention. Those triggering events have enormous importance for service descriptions. They couple to the solution at hand that is associated with how and what of a required IT solution.

The Annotation Dimension. According to [14], annotation with respect to arbitrary ontologies implies general purpose reasoning supported by the system. Their reasoning approaches suffer from high computational complexities. As a solution for dealing with high worst-case complexities the solution recommends a small size input data. Unfortunately, it is contradicting the impressibility of ontologies and define content as complex structured macro data. It is therefore, necessary to concentrate on the conceptualization of content for a given context considering annotations with respect to organizations intentions, motivations, profiles and tasks, thus we need at the same time sophisticated annotation facilities far beyond ontologies. Annotation thus must link the stakeholders or parties involved and activities; the sources to the content and concept.

The Concept Dimension. Conceptional modelling aims at creation of an abstract representation of the situation under investigation, or more precisely, the way users think about it. Conceptual models enhance models with concepts that are commonly shared within a community or at least between the stakeholders involved in the modelling process.

According to the general definition of concept as given in [19], Concepts specify our knowledge what things are there and what properties things have.
Concepts are used in everyday life as a communication vehicle and as a reasoning chunk. Concept definition can be given in a narrative informal form, in a formal way, by reference to some other definitions etc. We may use a large variety of semantics, e.g., lexical or ontological, logical, or reflective.

Conceptualisation aims at collection of concepts that are assumed to exist in some area of interest and the relationships that hold them together. It is thus an abstract, simplified view or description of the world that we wish to represent. Conceptualisation extends the model by a number of concepts that are the basis for an understanding of the model and for the explanation of the model to the user.

The definition of the ends or purpose of the service is represented by the concept dimension. It is the curial part that governs the service’s characterization. The purpose defines in which cases a service has a usefulness, usage, and usability. They define the potential and the capability of the service.

The Added Value Dimension. The added value of a service to a business user or stakeholder is in the definition of surplus value during the service execution. It defines the context in which the service systems exists, the story line associated within the context, which systems must coexist under which context definitions prevailing to time. Surplus value defines the worthiness of the service in terms of time and labor that provide the Return of Investment (ROI).

4 Conclusion

There are many other usage models for services. This paper elaborated the explanation, informed selection, and appropriation usage model for a service. Other usage models of an instrument as a model are, for instance, optimization-variation, validation-verification-testing, understanding, extension and evolution, reflection-optimization, exploration, documentation-visualization, integration, hypothetical investigation, and description-prescription usage models. We introduced in this paper a general notion of the model and showed what makes description or specification a service to be become a model of the service.

References

Wherefore Models are Used and Accepted?
The Model Functions as a Quality Instrument in Utilisation Scenarios

Marina Tropmann-Frick*, Bernhard Thalheim*
*Christian Albrechts University Kiel, Department of Computer Science, D-24098 Kiel, Germany
mtr | thalheim@is.informatik.uni-kiel.de

Abstract. Science and technology widely uses models in a variety of utilisation scenarios. Models function as a representation of origins in some of these utilisation scenario, e.g. they function for explanation, for optimization-variation, for validation-verification-testing, for exploration, for hypothetical investigation, for documentation-visualization, and finally for description-prescription as a mediator between a reality and an augmented reality that developers of a system intend to build. The functions of a model determine the purposes, the goals, and the kind of the model. The model effect as an instrument in these utilisation scenarios. Its qualities determine whether a model is acceptable, whether it is of high utility, why and for which reason a model is successfully used, and what are the convincing properties. These qualities can be derived from the utilisation scenario in which a model functions as an instrument.

1 The Mission of Models in Computer Science and Engineering

Computer science and engineering widely uses models. At the same time, the notion of the model is still not commonly agreed. There are many different application cases for models. Models are representations\(^1\) of a collection of origins or originals. Origins can be material goods, systems, software, reality, augmented reality, imaginations of a person, etc. Following H. Stachowiak Stachowiak (1973, 1992), a model is often defined in a phenomenalistic way based on three properties:

1. Mapping property: the model has an origin and can be based on a mapping from the origin to the instrument.

\(^1\) Computer science and engineering uses the word ‘artefact’ for a representation. An artefact has beyond the meaning “any object made by human beings, especially with a view to subsequent use” also other meanings such as:
- any mass-produced, usually inexpensive object reflecting contemporary society or popular culture;
- a substance or structure not naturally present in the matter being observed but formed by artificial means;
- a spurious observation or result arising from preparatory or investigative procedures;
- a structure seen in tissue after death, fixation, staining, etc.
Wherefore Models are Used and Accepted?

(2) Truncation (reduction) property: the model lacks some of the ascriptions made to the origin.

(3) Pragmatic property: the model use is only justified for particular model users, the tools of investigation, and the period of time.

We observe however that these properties do not qualify a representation as a model. The mapping and truncation properties are far too strict and need further investigation. We might use representations that are not images of mappings such as a Turing machine, a system architecture, or development strategies. Furthermore, we might use representations that are not reducts of origins such as (conceptual) information system models for the variety of viewpoints users of databases might have. So, we need a better definition of the notion of a model.

Models are developed by a community of practice for utilisation by a community of practice and in a context. The utilisation depends on the intentions of users and their context. So, we observe that the utilisation of models determines (a) the kind of model, (b) the governing purposes or goals of utilisation of the model, (c) the properties of a model, (d) the amplification a model provides with extensions, (e) the idealisation by scoping the model to the ideal state of affairs, (f) the divergence by deliberately diverging from reality in order to simplify salient properties of interest, and (g) the added value of a model. The seven additional statements are combined in the mission a model has. The mission clarifies how the model functions well within its intended scenarios of usage according to its capacity and potential. The mission must be coherent with the context, the determination or specific basis of conduct or utilisation of the model, and must be acceptable for the users or – more concrete – the community of practice. Therefore, the mission clarifies the functions (and anti-functions or forbidden ones), purposes and goals of the utilisation, the potential and the capacity of the model.

The mission is combined in the cargo Mahr (2008); Thalheim (2015); Thalheim and Nissen (2015b) with the determination of the representation (basic ideas, features, particularities, and the utilisation), an abstract declaration of the meaning (main semantic and pragmatic statements about the model; description of value of the representation according to its functions in the utilisation scenarios; its importance within the given settings), and a narrative explanation of the identity within the five kinds of identity: actual, communicated, accepted, ideal, desired identity.

The Storyline of the Paper. Questions that must be answered with any model are: Why, for what cause or for which reason, on what account a model is utilised? What is the intention underlying this utilisation? Why a model is acceptable within a certain scenario? What are the characteristics of a model that convince and persuade users to utilise this model?

In this paper, we start with reminding the conception of the model. Models are utilised as an instrument in some scenarios. These scenarios determine the functions the model has to play. Functions determine the purposes and goals we aim at. At the same time, any model has also limitations that restrict its utilisation. Whether an instrument can function as a model depends on its adequacy and dependability. Adequacy is well-considered in many publications. Dependability needs however clarification and deeper investigation. It combines the justification of utilisation of the given instrument and the sufficiency of the instrument in the given scenarios. Sufficiency can be based on quality characteristics and on evaluation procedures for these characteristics. We thus may derive maturity statements of a given model. We finally apply this quality evaluation to model used for description and prescription scenarios, e.g. information system models.
2 The Conception of the Model

A model is a well-formed, adequate, and dependable instrument that represents origins. Embley and Thalheim (2011); Thalheim (2014a,b)

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilisation scenarios.

The model should be well-formed according to some well-formedness criterion. An well-formed instrument is adequate for a collection of origins if it is analogous to the origins to be represented according to some analogy criterion, it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its purpose.

Well-formedness enables an instrument to be justified by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through formulas, by falsifiability, and by stability and plasticity.

The instrument is sufficient by its quality characterisation for internal quality, external quality and quality in use or through quality characteristics Thalheim (2010) such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

A well-formed instrument is called dependable if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

Figure 1 depicts dependability and adequacy properties of a representation that is used as an instrument in some utilisation scenarios. We explore therefore next models in their utilisation scenarios. Adequacy and justification has already been defined and considered in detail in Embley and Thalheim (2011); Thalheim (2010, 2014a); Thalheim and Dahanayake (2015); Thalheim and Tropmann-Frick (2015, 2016).

The model has a background consisting of an undisputable grounding from one side (e.g. paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities) and of a disputable and adjustable basis from other side (e.g. assumptions, concepts, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense). The background its often given only in an implicit form.

Models function as instruments or tools. Typically, instruments come in a variety of forms and fulfill many different functions. Instruments are partially independent or autonomous of the thing they operate on. Models are however special instruments. They are used with a specific intention within a utilisation scenario. The quality of a model becomes apparent in the context of this scenario.

Lightweight models typically cut off background and context. They assume per default some utilisation scenario and reduce the functions of the model to the main function. The purpose is then driven by this function. Often the community of practice is set to some standard community that uses a specific kind of justification. In this case, the sufficiency criteria are often related to well-formedness criteria Cherfi et al. (2002, 2007), e.g. syntactic ones. We notice that the mapping, truncation and pragmatic properties become simpler. We may extend this kind of scoping to generic models Thalheim et al. (2014) that are particular or idealised models for a specific community of practice with a specific background, within a specific
Wherefore Models are Used and Accepted?

class, and for representation of a specific world of origins under consideration. Generic models can be calibrated to specific models through a process of data or situation calibration, refinement, concretisation, context enhancement, or instantiation.

3 Models and Utilisation Scenarios

Models function as an instrument in some usage scenarios and a given usage spectrum. Their function in these scenarios is a combination of functions (explanation, optimization-variation, validation-verification-testing, reflection-optimization, exploration, hypothetical investigation, documentation-visualisation, description-prescription). The model functions effectively in some of the scenarios and less effectively in others. The function determines the purpose and the objective (or goal) of the model. Functioning of models is supported by methods. Such methods support tasks such as defining, constructing, exploring, communicating, understanding, replacing, substituting, documenting, negotiating, replacing, optimizing, validating, verifying, testing, reporting, and accounting. A model is **effective** if it can be deployed according to its objectives.

Model adequacy

- purposeful, e.g. pragmatic
- more focused, e.g. truncated, abstract
- analogous, e.g. mapping
- well-shaped, well-formed, well-defined

Model utilisation

Function $F_A$ Function $F_B$

Scenario A Scenario B

Model dependability

**FIG. 1 – The model as an instrument**

(1) that is used in utilisation scenarios in certain functions,
(2) that is adequate for representation of origins, and
(3) that is dependable for its utilisation.

Figure 1 present model configuration as a three-dimension characterisation based on utilisation and related adequacy and dependability properties. The specific form of characterisation varies.
a lot in dependence of the kind of model. Its mathematical rigidity can be rather Fuzzy or based on certain calculi (deductive, inductive, abductive, argumentation, etc.).

4 Functions and Kinds of Models

In general, a model is a representation of origins (‘what’) and in some cases a model for new origins Mahr (2015). It is used by users that form together with the modelers the community of practice (‘who’) within their context. The users have their specific ‘stereotype’ or pattern of using the model, i.e. the model has a function within these utilisation scenario (‘where’, ‘whereat’). We can thus consider the origins, the community of practice, the utilisation, and the context as four governing directives for a model.

A model is thus an instrument that functions within utilisation scenarios (‘Gebrauchsspiel’ (deployment story), ‘Sprachspiel’ (language game, Wittgenstein (1958))) in different roles with different rigidity, modality and confidence (‘how’). Each science and engineering discipline uses specific scenarios. The role and function of a model may vary. Typical functions of a model are: explanation, optimization-variation, validation-verification-testing, reflection-optimization, exploration, hypothetical investigation, documentation-visualization, and description-prescription. The last function uses models as a mediator between a reality and an augmented reality that developers of a system intend to build.

The different functions of models are supported by different kinds of models. Models are in general used as perception models (reflection of one party’s current understanding of world; for understanding the application domain), situation models (reflection of a given state of affairs), conceptual models (based on formal concepts and conceptions), experimentation models (as a guideline and basis for experimentation), formal models (based some formalism within a well-based formal language), mathematical models (in the language of mathematics), computational models (based on some (semi-)algorithm), physical models (as physical artifact), visualisation models (for representation using some visualisation), representation models (for representation of some other notion), diagrammatic models (using a specific language, e.g. UML), exploration models (for property discovery), and heuristic models (based on some Fuzzyness, probability, plausibility, correlation), etc. The large variety of notions for a model (e.g. see Thalheim and Nissen (2015a) for models used in science and engineering or Embley and Thalheim (2011); Thalheim (2014a,b) for conceptual models for database structuring) mainly reflects these different kinds.

5 Capacity, Potential, and Maturity of a Model

Capacity is a strategic measure whereas the potential is a tactical one. The potential can be used to derive the added value of a utilisation of a model within a given scenario. The potential allows to reason on the significance of a model within a given context, within a given community of practice, for a given set of origins, and within the intended profile.

The capacity relates an instrument to utilisation scenarios or the usage spectra. We answer the questions whether the instrument functions well and beneficial in those scenarios, whether it is well-developed for the given goals and purposes, whether it can be properly, more focused, comfortably, simpler and intelligible applied in those scenarios instead of the origins,
and whether the instrument can be adapted to changes in the utilisation. The answers to these questions determine the main content or cargo, the comprehensiveness, and the authority or general value of a model. Another important aspect is the solution-faithfulness of the instrument. The capacity is an essential element of the model cargo, especially of the main content of the model.

Similar to SPICE assessments ISO/IEC (2006); Fiedler et al. (2009); Jaakkola et al. (2005), we may rate maturation of a model and a modelling approach to:

1. A model is defined in an ad-hoc manner.
2. A model is mainly defined in an informal form.
3. Model development is systematic and managed and the model is of high quality.
4. Models are based on standards and are well-understood.
5. Modelling follows a continuously improvable, supporting evolution and migration style.

The maturity ladder and modelling experience we have observed for three decades direct us to the hypothesis that most Computer Science modelling approaches have not yet reached level (2).

We use now the collection on the Kiel modelling approach Thalheim and Nissen (2015a) for an exploration of quality characteristics of models.

6 Sufficiency of Models: Quality Characteristics

Models have a function within their utilisation scenario. There are different utilisation scenarios such as construction, exploration, and explanation. According to Thalheim (2010), we may distinguish between

- internal quality characteristics (accuracy, suitability, interoperability, coherence, stability, generality, robustness, flexibility, self-contained, independence, minimality, language quality, compositionality, uniformity, changeability, documentation),
- (development or) external quality characteristics (correctness, pervasiveness, analysability, changeability, stability, testability, privacy of the models, ubiquity, expressiveness, generalisability, existence of refinement and abstraction hierarchies, traceability, adaptability, maturity, fault tolerance, recoverability, reliability compliance, configuratatability, resource utilisation, scalability, testability, maintainability, stability, portability, reusability, replaceability), and
- quality of use characteristics (understandability, learnability, usefulness, comprehensibility, parsimony, operability, attractiveness, appropriatedness, availability, efficacy, efficiency, functionality, utility, usability, use, dependability, performance, fitness, productivity, safety, trust, satisfaction).

These quality characteristics are static. We may however also consider dynamic ones such as executability, refinement quality, scope restriction, effect preservation, context explicitness, and completion tracking.

We may use the refined categorisation for internal characteristics in Fieber et al. (2008):

- The inner quality of a model is given by well-formedness characteristics (representation, syntactic well-formedness, semantic well-formedness, and pragmatic well-formedness,
modularity, controlled redundancy, clarity, style conventions, according to the rules), by
syntactic characteristics (precision, syntactic simplicity, syntactic adequacy, level of de-
tail), by semantic characteristics (universality, semantic simplicity, semantic adequacy,
consistency, accuracy, degree of formalisation), and by pragmatic characteristics (con-
ceptual integrity and uniformity, conformity, variety of representations, consistency with
people or organisations).
– The outer quality relates the model to other models or to their origins: cohesion, cor-
rectness, completeness, traceability, changeability, validity, generality.

Another categorisation is given by Prat and Cherfi (2003); Cherfi et al. (2002); Akoka et al.
(2007); Prat et al. (2014): The system dimension evaluates the goal, the environment, the
structure, the activity and the evolution. It is supported by certain evaluation criteria. Software
engineering uses metrics for evaluation. Another evaluation procedure is the one in Jaakkola
and Thalheim (2010) where semantic and pragmatic calculi are used. The large variety of
quality characteristics makes evaluation complex.

We prefer thus the approach used in Cherfi et al. (2007); Jaakkola and Thalheim (2010). A
quality evaluation stereotype is a selection of quality characteristics that are driven by
– the functions of a model in utilisation scenarios that are considered, the resulting pur-
poses and goals, and quality characteristics that are essential for the profile sufficiency,
– the kind of the model, and
– its capacity and potential.

Let us now develop one quality evaluation stereotype for the construction of systems.

7 The Description-Prescription Scenario for Structural Models of Information Systems

The construction scenario is one of the central modelling scenario for information and soft-
ware system development. The model functions as a mediating specification for all viewpoints
business users might have, as a blueprint for software development, and as an assessment arti-
fact for the system developed accordingly.

Let us consider now quality characteristics for models that are used for development of in-
formation systems. We noticed already Thalheim and Tropmann-Frick (2015) that models are
used (1) for communication and negotiation, (2) for conceptualisation, and for (3) realisation
of a system. These models are different. Their primary quality characteristics vary as well.

We follow the design science approach Dahanayake and Thalheim (2011); Prat et al. (2014)
and separate the construction process into three stages: relevance stage, modelling stage, and
realisation stage. These stages may also be considered as the description phase followed by
the prescription phase, i.e. first a system is described by a model and second the model is used
as a prescription or blueprint for the realisation. This approach uses stepwise development of
different artifacts according to Mahr (2009): first, the scope is set to a set of origins \( O \); second,
the relevant and necessary properties \( \Phi(O) \) of \( O \) are elicited; third, these properties are mapped
to objectives \( \Psi(M) \) of the model \( M \) to be developed; fourth, the model is developed; fifth, we
extract essential, relevant and necessary properties \( \Phi(M) \) of \( M \) and map those to objectives
\( \Psi(Y) \) for the system \( Y \) under development. Finally, we may assess the system based on the
properties \( \Phi(Y) \) the system has.

Ingenierie Management des Systemes D’Information, 2016
Wherefore Models are Used and Accepted?

We observe that support for modelling results in a wide variety of reasoning activities. For instance, reasoning about properties of a model is also based on an explicit consideration of the notion of an analogy between the model and the application domain origins or the model and its reflection in theories and constructions. Reasoning on objectives of realisations includes detection of requirements a system must satisfy. Figure 2 displays the different ways of working during an information systems development.

![Diagram](image)

**Fig. 2 – Two of the three design science stages of information system modelling: description followed by prescription and corresponding kinds of quality characterisations**

The scenario uses models in three different functions during information and software systems development. These function are well-supported if $O$, $\Phi(O)$, $M$, $\Phi(M)$, $Y$, and $\Phi(Y)$ obey the following properties:

**Origins capture quality:** Primary characteristics are: suitability, clarity, stability, plasticity, usability, and refinement & abstraction quality. We can also consider secondary characteristics such as: generality, controlled redundancy, usefulness, use, efficacy, and generality.

**Comprehension quality:** Primary characteristics are: adequacy, justification, generality, consistency, correctness, appropriateness, fitness, and trust. We can also consider secondary characteristics such as: coherence, robustness, universality, and flexibility.

**Requirements quality:** Primary characteristics are: completeness, utility, accuracy, compositionality, understandability, and validity. We can also consider secondary characteristics...
such as: language quality, precision, generalisability, variety of representations, and clarity.

**Model quality:** Primary characteristics are: syntactic and semantic well-formedness, minimality, completeness, correctness, and usefulness. We can also consider secondary characteristics such as: modularity, uniformity, learnability, parsimony, simplicity, and degree of formalisation.

**Blueprint quality:** Primary characteristics are: accuracy, refinement & abstraction quality, correctness, and conformity. We can also consider secondary characteristics such as: coherence, robustness, and flexibility.

**Specification quality:** Primary characteristics are: adaptibility, completeness, and understandability. We can also consider secondary characteristics such as: well-formedness, syntactic characteristics, stability, flexibility, self-contained, uniformity, changeability and adaptability, and configuratability.

**Realisation quality:** Primary characteristics are: changeability, maintainability, dependability, configuratability, resource utilisation, and efficiency. We can also consider secondary characteristics such as: documentation, traceability, productivity, fault tolerance, reliability compliance, and scalability.

**Assessment quality:** Primary characteristics are: analysability, and testability. We can also consider secondary characteristics such as: configuratability, efficicacy, and trust.

The lists of quality characteristics are not complete. They show however that different characteristics are important during description and prescription. We observe that most of the quality characteristics must be semantically or syntactically defined Jaakkola and Thalheim (2010) and can only partially represented through metrics.

These quality characteristics are still too manyfold and become impractical. The lists become far shorter if we consider models in dependence on their main utilisation scenario. For instance, the description scenario uses models for communication and negotiation goals. The prescription scenario uses models according to the image, design and action, blueprint and realisation goals. The lists can also be calibrated according to the methodology that is used for information system development. The lists become condensed if the stages or their corresponding phases already use models, e.g. if high quality perception models and high quality situation models are already known at the phases of the relevance stage. A similar shaping can be developed for conceptualisation scenarios for information systems.

Let us now consider structure-driven information system development based on extended entity-relationship modelling language Thalheim (2000) that uses a prototyping methodology and a given concept space $C$ in a well-known application area for straightforward description directly followed by prescription. This development uses a specific quality evaluation stereotype

\[ \text{structureIS} \]

which adornment is given by the scenarios, the methodology, the outcome, and the background of the model:

**Driving quality concerns** are the following ones:

- Functions of the model are quick communication (and negotiation) and prototypical system realisation. Resulting purposes are thus design and action, reasoning on consolidation, derivation of problematic elements, and realisation of a running system.
Wherefore Models are Used and Accepted?

The goal is the development of at least one realisation as a proof-of-concept. The prototype may later be revised or used for proper development.

- The model is a mediating, guiding, and inspiring instrument. It is the basis for prototypical realisation, for refinement of the approach taken, and for experimentation with the realisation. The model itself is informal or semi-formal. It incorporates visualisation features. In our case, it is tightly bundled with its diagrammatic sub-model in the extended entity-relationship modelling language.

- The capacity of the model describes the general properties of a model, e.g. its completeness, its non-functional and functional properties, its integrateability or interoperability, its advantages and disadvantages considering missing part, the development of appropriate functionality, provided reflections for user viewpoint, and its realisability of proper high-quality system. The potential characterises the performance of a model, e.g. the model capability, its fruitfulness, its restrictions and boundaries of the realisation, and corresponding justification considerations. The maturity of prototype-oriented models is typically on level 1 only.

Primary quality characteristics are controlled conceptual completeness, controlled conceptual correctness, syntactic correctness, syntactic completeness, flexibility, analysability, comprehensibility, potential operability, potential functionality, changeability, and application awareness through meaningful representation of the real world.

Secondary quality characteristics are conceptual minimality, coherence, traceability, configuratability, syntactic and semantic well-formedness, semantic adequacy, usefulness, efficacy, and use.

We observe that both primary and secondary quality characteristics are used at one stage but irrelevant almost all other stages. So, the list of quality characteristics becomes manageable for quality evaluation of $O$, $\Phi(O)$, $\Psi(M)$, $M$, $\Phi(M)$, $\Psi(Y)$, $Y$, and $\Phi(Y)$. For instance, assessment quality is given via analysability, potential functionality, changeability, configuratability, and efficacy. These five quality characteristics are partially derived from the quality characteristics for $\Psi(Y)$ and $Y$.

8 Principles of Model Development and Utilisation

We summarise our observations to principles of quality model design for description-prescription scenarios similar to Chen et al. (1999); Thalheim (2010):

**Community principle:** The model must support the entire community of practice and match to their understanding, their focus and scope, and their reasoning abilities in a non-disinterpretable form. The model can be used efficiently and comfortably and with a minimum of fatigue.

**Scenario principle:** The model accommodates a wide range of its application within its utilisation scenarios based on the specific utilisation scenarios of their users.

**Adequacy principle:** The model must be as focused as only possible and must be understandable regardless of experience, knowledge, and skills of users that accept the same grounding and tolerate the basis. Unnecessary complexity is avoided. It is consistent with expectations and experience. It accommodates a wider range of bases.
Justification principle: All elements of the model are justified by a corroboration that relates them to origins, by coherence and conformity criteria, by an explicit statement on scope and focus, and by stability considerations against the potential set of origins.

Effectivity principle: The model delivers necessary elements effectively to their users, regardless of the user’s skills and of ambient conditions. A model supports different representation models. The model provides a clear line of sight of its elements for any user. It accommodates effective utilisation for its profile and provides variation features for similar functions.

Robustness principle: The model minimises hazards and the adverse consequences of accidental and unintended utilisations that are not supported by the profile of the model.

9 Concluding: The Model Functions as a Quality Instrument in Utilisation Scenarios

This paper is a contribution to a general theory of models and modelling in Computer Science. Typical deficiencies of modelling in Computer Science are: ad-hoc modelling, modelling in the small, limited reuse of models, models are not understood as some kind of programs, and rigid separation into sub-disciplines without development of a common understanding and culture. Models are considered to be the third dimension of science Thalheim and Nissen (2015a). Modelling is one of the four central paradigms of Computer Science beside structures (in the small and large), evolution or transformation (in the small and large), and collaboration (based on communication, cooperation, and coordination).

Models function in utilisation scenarios as instruments. As such models should be of high or at least sufficient quality. Many quality characteristics are known and partially of importance and relevance. Typically, qualities form a parameter space. Quality characteristics can be categorised into main or primary parameters and secondary, tertiary etc. ones in dependence on the function that a model has in a given utilisation scenario.

References


Wherefore Models are Used and Accepted?


Thalheim, B. (2014b). Models, to model, and modelling - Towards a theory of conceptual models and modelling - Towards a notion of the model. Collection of recent papers,
Résumé

Models as adequate and dependable representations of origins. They function in utilisation scenarios and are thus instruments. Their dependability can be given on the basis of a justification and evaluation for some main quality characteristics which are selected according to the functions a model has in a given utilisation scenario. The quality characterisation can be defined as a manyfold of quality parameters in which some are main and others are not relevant or distinctive. Which quality characteristics is considered to be primary is determined by the utilisation scenarios and the functions the model has.
Abstract. A model functions in a utilisation scenario as an instrument. It is well-formed, adequate and dependable. It represents or deputes origins. This conception of the model is a very general one. Computer engineering uses models for description of development intentions and for prescription of the system to be build. It typically uses a number of models depending on the layer of abstraction, the scope, the context, the community of practice, and the artefacts to be represented. Model-based development is one of key success factors for development of database systems. This paper thus develops foundations for model-based engineering. Database system development is used as the illustration example for this investigation.

1 Models in Computer Science and Computer Engineering

Models are a kernel element of Computer Science and Computer Engineering (CS&CE). They are used sometimes without any definition or with an intuitive understanding. We know, however, a large variety of model notions (e.g. the 46 notions in [46]). A general theory, technology, art, science, and culture of modelling remain to be one of the research lacunas.

1.1 The Model

A model is a well-formed, adequate, and dependable instrument that represents origins. [11, 44, 45]

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilisation scenarios.

The model should be well-formed according to some well-formedness criterion. As an instrument or more specifically an artifact a model comes with its background, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form.
1.2 Multi-Model Modelling

Most sciences use coexisting models as a coherent holistic representation of their understanding, their perception, and their theories. For instance, medical research [8] typically considers medical models as experimentum, practicale, ratio, speculativum, and theoreticalis. These models are developed with different scale, precision, variability, vision, veracity, views, viewpoints, volume, and variation.

Each of the models has some functions in utilisation scenarios, for instance, communication, negotiation, construction, and representation and depiction functions. Depending on these functions, the model may be considered to be adequate and dependable. If we use several models then coherence of these models becomes an issue. We may explicitly represent coherence of models through model suites [7, 43]. We may also layer models based on their abstraction and scale, e.g. [17]. UML [32] uses ensembles of models that are loosely coupled.

A model suite consists of a set of models, an explicit association or collaboration schema among the models, controllers that maintain consistency or coherence of the model suite, application schemata for explicit maintenance and evolution of the model suite, and tracers for the establishment of the coherence.

A specific model suite is used for co-design of information systems that is based on models for structuring, for functionality, for interactivity, and for distribution [39]. This model suite uses the structure model as the lead model for functionality specification. Views are based on both models. They are one kernel element for interactivity specification. Distribution models are additionally based on collaboration models, e.g. [38]. Co-design includes coherence and maintenance of coexistence and co-evolution. Models might be complementary or completing or reversing or opposing each other, e.g. static and dynamic models in the HERM and BPMN languages. At the same time, a model suite integrates models and thus forms a new and more complex model which may convey totally different meanings. Models share than purposes, responsibilities, and meanings. A model suite may consist of two or more models (bi-models or diptych-models, triptych-models etc.). The association among models in a model suite is based on association styles and patterns such as master-slave, proxy, or publish-subscribe. Since models can exclusively serve some purpose the remaining models may be latent or inactive or of non-interest as long as the given purpose is of interest.

A specific model suite consists of two models which share most of their background, context, community of practice, their application scenario, and thus also function within these scenarios. These models coexist together, are interdependent, and are correlated to each other. We call such models co-model. Co-models form a diptyph1.

They can be coalesced into one model with two different sub-models or they may depend from each other (see, for instance, the Königsberg bridge models in [28] with the topographical, topological and graph-theoretic models). Origins are often also models and thus form together with their model a co-model. The

---

1 A diptych is work made of of two parts. So, we might call co-models also di-models or diptych models.
orgin $M_1$ thus conditions its model $M_2$, i.e. $M_2/M_1$ is a conditional. Modern CS&CE is full of examples of such co-models, e.g. [2, 9, 10, 13, 16, 25, 26, 31, 32, 34, 36, 41]. A model in a co-model also often inherits adequacy and dependability of the other model. Sometimes, they follow however also different backgrounds. For instance, eER-based conceptual modelling uses a global-as-design paradigm. BPMN-based conceptual models are based on a local-as-design approach with an orientation of actors with their roles.

1.3 Science and Engineering

Science and engineering are two rather different activities. According to the Encyclopedia Britannica [35], science is (1) the state of knowing, (2a) a department of systematized knowledge as an object of study, (2b) something (as a sport or technique) that may be studied or learned like systematized knowledge, (3a) knowledge or a system of knowledge covering general truths or the operation of general laws especially as obtained and tested through scientific method, (3b) such knowledge or such a system of knowledge concerned with the physical world and its phenomena alike in natural sciences, and (4) a system or method reconciling practical ends with scientific laws.

Engineering is nowadays performed in a systematic and well-understood form [1]. It also well supported in software engineering, e.g. CMM or SPICE [18]. Engineering is the art of building with completely different success criteria (see [37]): “Scientists look at things that are and ask ‘why’; engineers dream of things that never were and ask ‘why not’.” (Theodore von Karman) “Engineers use materials, whose properties they do not properly understand, to form them into shapes, whose geometries they cannot properly analyse, to resist forces they cannot properly assess, in such a way that the public at large has no reason to suspect the extent of their ignorance.” (John Ure 1998)).

S. Oudrhiri [33] considers four elements of matured engineering: “(a) the technological know-how, (b) a set of established practices, (c) a scientific approach for defining the underlying principles of these practices, and (d) an economical model to explain the implications of such practices in terms of value delivered (effectiveness) and resources consumed (efficiency)”. Engineering is inherently concerned with failures of construction, with incompleteness both in specification and in coverage of the application domain, with compromises for all quality dimensions, and with problems of technologies currently at hand. [48] distinguishes eight stages of engineering: inquire, investigate, vision, analyse, qualify, plan, apply, and report.

1.4 Co-Models and Model Suites in CS&CE

CS&CE often uses direct associations of models, i.e. a model is based on another model. Modelling is then concerned with two models at the same time. For instance, completed database structure modelling starts with a situation model that is represented by a perception model. This perception model is the basis for the business model which is again the basis for the conceptual model. The
conceptual model is mapped to a logical model according to the platform for realisation of the database system. The logical model is then mapped to the physical model. This pairwise modelling is based on a dichotomy of the models.

This dichotomy is used for closing the gap between the user world (where the information system is a social system) and the IT world (where the information system is a technical system). Figure 1 displays the mediating functions of typical information system models. Classical development methodologies are often based on consideration of two models. For instance, structure modelling might start with a business data model that is intentionally based on the perception model within the user world. This business data model is refined to a conceptual structure model in a conceptual modelling language such as ER [42]. The conceptual structure model is enhanced, transformed or compiled to a logical data model. If we follow a systematic approach then the logical data model is refined, enhanced and transformed to a physical data model.

1.5 Model-Based Engineering as Specific Model-Based Reasoning

Model-based reasoning is reasoning with the aid of models, reasoning about models in their own right, and reasoning that is model-determined [27, 30]. Models have then three different functions depending on these reasoning scenarios [6, 46]; models are instruments for reasoning which implies their prior construction and the reasoning necessary for their construction; models as targets of reasoning;

Fig. 1. Model suites: The five main models that comprise a complete model of a database system
models as a unique subject of reasoning and its preliminary. Abduction has been considered the main vehicle of model-based reasoning. In CS&CE, reasoning is also based on explicit consideration of adequacy and dependability of models within the description/prescription scenario. From one side, models are used as a representation of some thought or better some mental models (e.g. perception models) which are representing the (augmented) reality (i.e. the perceived situation model and the objectives for system construction). From the other side models are used as blueprints for realisation of intentions by software systems. In the last case, models are also documentation models for the software system, at least at the first completion of the system.

Model-based engineering has been considered for a long time as ‘greenfield’ development starting from scratch with a new development. Engineering is however nowadays often starting with legacy systems that must be modernised, extended, tuned, improved etc. This kind of ‘brownfield’ development may be based on models for the legacy systems and migration strategies [22]. Again, we observe a co-model approach with a legacy model for revision, redevelopment, modernisation and migration and a target model for development of the new modernised and extended system. So, the legacy model (or legacy models) is associated with a sub-model of the target system.

1.6 The Objectives of the Paper

Model-based engineering attracts a lot of research, e.g. [3, 21, 23, 40]. Model-driven software development (MDSD) distinguishes enterprise, platform independent, platform specific, and code models. MDSD on the basis of model suites and with a direct consideration of model properties has not yet been investigated. So, we start with a case study in Section 2. This case study is used for derivation of principles in Section 4. Finally, Section 5 discusses the role of conceptual models in model-based engineering of database system development.

Due to space limitations, the paper cannot discuss in detail techniques that are necessary for systematic model-based engineering. Many techniques are already developed for specific modelling languages, for specific application domains, and for specific development approaches. A systematic generalisation and harmonisation of these techniques is still a research task. We illustrate the approach based on entity-relationship modelling (ERM) languages, on data-intensive applications and ERM-based development. The paper aims thus in a methodological background for model-based engineering. We restrict the paper to co-models and their specific style for model-based engineering.

2 A Case Study for Structure-Representing Co-Models

Let us consider two cases of co-models. It is often claimed that the ER modelling language can be used at the business and the conceptual layer in a similar form. If we look a bit more into the details then we discover essential differences that
must be taken into consideration. For instance, we might have models that cannot be mapped to models at the lower layer or models that cannot be represented at the higher layer. At the same time, we might have many choices for lower layer models (Figure 2). Moreover, data models at one layer might not be entirely represented by data models at the other layer. For instance, cardinality constraints might not be representable by classical relational constraints. We must either enhance the relational language or represent constraints by procedural features of the relational database platforms.

2.1 Co-Models: Business Data Models and Conceptual Models

Business data models reflect the way how business users consider their data. Each business user considers only specific data within a specific viewpoint. A business application provides some kind of collaboration or exchange mechanism for these data.

The origins that are reflected in business data models are the situation model of a given application area and a collection of perception models that reflect specific viewpoints of business users. The understanding of data by business users is based on the way of work at business. So, data models represent their rather specific understanding of the application domain. These data models follow a local-as-design representation style. Conceptual models follow however a global-as-design approach [47], i.e. the model consists (i) of a global schema that harmonises and integrates the variety of viewpoints and (ii) of generalised (external) views that are derivable from the form the global schema and represent the local viewpoints. The two kinds of models - the business data models and the conceptual model - are tightly associated by an explicit infomorphism (i.e. generalised di-homomorphisms, see below). Adequateness and dependability of the conceptual model is derived from this association. Additionally, well-formedness of the conceptual model is based on the language, e.g. an extended entity-relationship (eER) modelling language, e.g. HERM [42].

Business (layer) data models and conceptual (layer) data models are a typical example of a vertical model suite since the first one is typically more abstract and the second one can be considered to be a refinement of the first one. The binding among these models is often implicit. We may however enhance the two models by a mapping that maps the first model to the second one. This mapping
combines and harmonises the different views that are used at the business user layer.

2.2 Co-Models: Conceptual Models and Logical Models

Logical models are based on the same underlying semantics for the modelling language, e.g. set semantics. Physical data models typically use multi-set semantics (also called bag semantics) for (object-)relational database management systems. Logical models may follow object-relational approaches or purely relational approaches. eER conceptual models have an implicit semantics beside the explicit semantics. For instance, relationship types obey an inclusion and an existence constraint that restricts existence of relationship objects by existence of their referred component objects – in most cases entity objects.

Conceptual views are represented by a collection of object-relational views. We have a number of potential associations between conceptual and logical models. Which one is appropriate depends on choices for structuring, for re-organisation or optimisation or normalisation, for handling of constraints, for handling of missing values, for controlled redundancy, for treatment of hierarchies, for naming, etc. Additionally, specific platform-oriented features are integrated into the logical model. The transformation follows rules and uses specific decisions.

So, the conceptual and the logical models are co-models that follow a refinement approach [49] (1) by injecting specific styles, tactics, embeddings, and language pattern to the logical model [1] and (2) by rules for transformation, extension, enhancement, and specialisation applicable to the logical model [12]. So, a conceptual model is typically associated to many logical models depending on the style of chosen refinement. We may consider an abstract description of the refinement approach as pragmas which are already given together with the conceptual model. The refinement may also result in an information loss. For instance, the view schemata defined for the conceptual model are mapped to a collection of relational views. The interrelation among the relational views is however not maintained in an explicit form.

Conceptual (layer) data models and logical (layer) data models also an example of a vertical model suite with a straightforward mapping from the conceptual layer to the logical layer.

2.3 Co-Models: Conceptual Co-Design of Structuring and Functionality

Database design and development typically is based on two models for structuring and functionality. The structure model is the ‘lead’ model for functionality since it defines the signature of the basic terms. The structure model imposes however also restrictions to the functions due to the integrity constraint enforcement and maintenance. Functionality is specified as a set of create-retrieve-update-delete functions. The data modification functions can be extended for preservation of integrity. The retrieval functions are defined based on a number
of retrieval pattern and as algebraic expressions, e.g., HERM+ [42]. So, the lead model is some kind of ‘order’ model and the functionality model is partially ‘enslaved’ [15].

Structure models and functionality models form a horizontal model suite. Their association is based on an infomorphism (see the similar vertical case in Section 4.1). All elements of the models are associated in a bipartite graph. The edges in the graph may be enhanced by existence dependencies, e.g., an operation or query uses the structural notions which are defined in the structural model. The control of such dependencies may be defined in a form similar to referential integrity.

2.4 Lessons Learned for Model-Based Engineering

A modelling language has its own obstinacy. It injects its background, its limitations and its treatment of semiotics into the model. Therefore, model-based engineering must explicitly represent these language specifics. Whenever models are used within a model suite, the association of models is language-biased and language-limited. Next, models are also driven by the directives, i.e., the artifacts to be represented, the profile of the model that is intended, the community of practice that might accept the model, and the context into which the model is set. Furthermore, the capacity and potential of the model itself restricts applicability. From the other side, we may restrict engineering to some kind of ‘best’ effective and efficient model. Finally, the classical approach to arbitrarily enhance a lower layer model limits the usefulness of the higher-layer model.

We may now consider either co-models at the same layer of abstraction (“horizontal co-models”) or at different layers of abstraction (“vertical co-models”). Database structure development is typically based both on vertical co-models that are on adjoining layers and on horizontal co-models in the co-design case.

3 The First Principle of Modelling

3.1 Logoi of Modelling

Modelling results in a model as a surface structure and is in reality combined with a deep structure that is based on the background and the directives of the model. The deep structure of a model is represented by the modelling logos\(^2\) [5, 24] that is the rationale or first principle behind modelling.

The model has its background \(\mathcal{B}\) consisting of an undisputable grounding from one side (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities) and of a disputable and adjustable basis from other side (assumptions, concepts, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense) which represent

---

\(^2\) In the Faust poem by J.W. Goethe, Faust reasons in the study room scene on the meaning of the word ‘logos’ λόγος. This word has at least 6 meanings where Faust used only four of them: word, concept, judgement, mind, power, deed, and reason.
Fig. 3. Utilisation scenarios for models and stages of their deployment
the nature of things themselves. The background provides the deep structure of the model by explanations, analysis and manifestation. It is governed by its inner directives (origins/artefact to represented $\mathcal{D}$, profile of the models (goal, purpose, function)) $\Psi$ and the outer directives (community of practice, and the context) $\mathcal{D}$. It is based on a language $\mathcal{L}$ with its general notion, capacity and potential. Model development is based on actions $\mathcal{A}$ and modelling and utilisation methods with their rational choice, i.e. the rationality expressed in the model as code, in interpretation and in action.

The modelling logos$^3$ consists of the background, the outer directives, the language, and actions. The modelling logos is expected to understand before model development and utilisation. The logos thus determines the modelling notions of trueness, verifiability, rationality, and correctness. Parameters for models themselves are the inner directives. We claim that models cannot be understood without understanding the modelling logos.

3.2 Scenarios and Resulting Functions of Models

These different meanings of the Greek word logos are used in different utilisation scenarios. Concept and conceptions are the basis for the perception and utilisation scenario. A conceptual model is a model that incorporates concepts and conceptions. Models might be accepted in a community of practice based on judgements of members of a community of practice [19]. Models may be acceptable for this community and be thus intellectually absorbed. Models then gain an expressive power and make sense within an application. Models can also be used and applied in a development process. This application may also use methods of matured development. The last one is based on model-based reasoning which can be guided by maturity approaches, e.g. CMM and SPICE. So, we observe a number of scenarios which are depicted in Figure 3.

Models function as instruments in these scenarios at various stages of maturity. For instance, the application scenario may use models as an inspiration for further development. This stage is often observed for UML-backed programming. Instead, models may be deliberately applied or managed. They may be used as co-models and thus co-evolve together with the realisation, i.e. they become reorganised during utilisation. This reorganisation may also based on systematic approaches and thus be based on a refinement strategy.

4 Engineering for Vertical Co-Models

4.1 Database Development with Vertical Co-Models

Vertical co-models are widely used in CS&CE. The methodologies developed so far do however not consider the nature of multi-models. The case studies in Section 2 showed the influence of the background-ladeness of models. It is not

---

$^3$ The logos combines specification of the language, the knowledge behind, the reality under consideration, and the actions. [24]
easy to switch from a local-as-design paradigm to a global-as-design paradigms. Models are also directives-laden, especially with the outer directives community-of-practice and context. It is simpler if data are of the same granularity, scale and scope. For this reason conceptual models use an approach to represent data at their lowest scale and smallest granularity. Scientific databases (and also industrial databases, e.g. [20]) often start with raw data and consider them as the basis of all derived, purged, combined, and analysed data. They fail whenever size of databases matters.

The association between co-models can be based on the notion of the infomorphism. We extend the notion in [22] for models as follows. Two models $M_1, M_2$ are $E_1, E_2$-infomorph though two transformations $E_1, E_2$ with $E_1(M_1) = M_2$ and $E_2(M_2) = M_1$ if any model object $o$ defined on $M_i$ can be mapped via $E_i$ to objects defined on $M_j$ for $i, j \in \{1, 2\}, i \neq j$.

We notice that this notion allows to associate models with different granularity, models that incorporate views defined on top of a global schema, model suites within the local-as-design style that have a latent association model underneath, and co-evolution of models within a model suite. It can also be extended to model refinement similarly to [49]. We may use the infomorphism also for justification of one model by another model similar to the associations discussed in Subsections 2.1 and 2.2.

4.2 Model-Based Engineering with Co-Models

Model-based engineering is turning an idea into a reality on the basis of models. Models are used as the tacit knowledge for engineering through conception, feasibility, design, manufacture and construction. They reduce complexity while at the same time providing means for sustainable development and for coping with the interdependencies between systems - technical ones as well as social ones, at different layers at the same time.

Engineering of information systems still needs a lot of research, theories, skills and practices. System development becomes nowadays based on iterative development. The time of one-way models is over. Models are becoming reused, reconfigured, continuously evolving and integrated. So, the five plus two models in Figure 1 must co-evolve. Modern CS& CE is not anymore concentrated on a singleton development but has to look outwards, to handle the ‘big picture’, to think and to reflect during practising, to manage complexity and risks at the same time in an economic form.

The details of sub-systems are beyond common sense. We must rely on instruments as an abstract source of understanding and managing. One central instrument are models for the system world, for a system, for sub-systems, for embedded systems, and for collaboration of systems. Models allow us to understand what we want, what we think to know and to manage, how we make achieve what we want, what actually to do, and finally what we think might be the consequences. Since engineering is also a business activity, engineering activities must be affordable and financially predictable. Models provide a practical commonsense view that helps us to manage professionally and at acceptable
risks. So, model-based engineering is one of the main issues of modern CS&CE. It goes far beyond model-driven development and model-driven architectures.

Therefore, we need first-class models and a technology to handle models in a holistic manner. One approach to master development is layering, i.e. coherently deploying various models of social systems and various models for technical systems. We develop this approach on the basis of business/conceptual and conceptual/logical co-models. In a similar form co-design of structuring and functionality may be managed and mastered.

So far we considered the modelling logo as a description logo. We may also consider the other model suite logos such as control, application, organisation, economics, and evolution logos for controllers, application, and tracers within model suites. Let us now sketch the controller and application ingredients for model-based engineering with co-models.

The model suite association style is based on general schemata for supporting programs (sub-model pattern for release, sharing, and access including scheduling of access), style of association (peer-to-peer, component, push-event, etc.), and on coordination activities describing the interplay among models. The control might be based on lazy or eager control styles.

The association pattern among models can be based on wrapping, componentisation, interception, extension or model models. The application processing can be active, proactive, synchronising or obligation-oriented. Synchronisation may use a variety of pattern. Whether association is based on parallel execution depends on the style of the association.

The model suite architecture describes inner association among models or sub-models and is given by a general network with pairwise or n-ary bindings among these models.

The model suite exchange is based on constraints, their enforcement and the handling mechanisms for associations among models and sub-models. They might include also obligations for maintenance of changes within a model suite.

The main issue behind this approach is to deeply understand how these models can coexist, co-evolve, influence and restrict each other, and support or hinder the other. So, we first develop an insight into the deployment and especially the modelling logo of such model suites for a co-model example.

5 Conceptual Models as Mediators Within a Model Suite

The conceptual model is often used as a medium and mediator [29]. “Models function not just as a means but also as a means of representation” [14] with a deep background such as starting points and questions, knowledge, theories, actual hypotheses, tacit knowledge in tools, goals and objectives, tools, data generation, data on hand, data processing, and data interpretation [4]. Mediating models are retrospective and prospective at the same time and ravish. Beside mediation, other and different models can also be developed for documentation, communication, negotiation, orientation, inspiration, etc.
5.1 The Dichotomy of Description and Prescription for eER Models

The main function of eER models is its utilisation during database structure construction. The model consists of a schema, a number of views, and the realisation style [39, 47]. It is descriptive and prescriptive. The descriptive part reflects the business user models and thus uses an explicit association by views. The prescriptive part can be based on realisation templates. Adequacy is given due to the association to the business models, due to the objectives of description and prescription, due to the explicit restriction to the model focus, and due to the realisation context. Dependability is based on the association to the business user model, on the objectives of co-design, and on the capacity and potential of logical modelling languages that we intend to use. So, the model reflects two rather different origins, the business model and the logical model.

5.2 Some Modelling Logos of ER Modelling

Modelling logos of (extended) entity-relationship modelling languages are hidden within the language and not explicitly discussed in the ER literature. They are partially reflected in literature that introduce other languages. They should however be known whenever ER modelling is performed.

The background is reflected by (for details see [42]): In the Global-As-Design approach, the schema reflects all viewpoints. Local viewpoints are derivable and somehow reflectable. Explicit existence postulates that any object must exist before there can be a reference to it, i.e. rigid separation of creation and use. The model assumes a closed-world view and unique names. It is based on a well understood name space or glossary or ontology. Salami-slice representation uses homogenous, decomposed types (potentially with complex attributes) with incremental type construction. Functionality representation is deferred without consideration of the performance impact to the schema. Separation into syntax and semantics allow to define semantics on top of the syntax. Explicit semantics is based on constraints. Paradigms, postulates, assumptions of database technology and database support are assumed due to the three main quality criteria (performance, performance, performance). Basic data types are hidden with some mapping facilities to DBMS typing systems. Visualisation is represented by one holistic diagram that displays the entire syntax and semantics.

Outer directives are (for details see [42]): The context is entirely determined by DBMS technology of the last decades and heavily restricted by the platform and the systems that should be used. Data must become identifiable. The population is finite what causes problems with cyclic constraints, e.g. locally defined cardinality constraints are then global constraints. The community of practice consists mainly of DBMS professionals, modellers and may be business deciders. The first two groups are used to and biased by the paradigms, postulates, assumptions, etc. of DB technology.

The potential and capacity of the ER modelling language is restricted by the flatness of the schema definition. Schema construction may be guided by style guides and well-formedness characteristics. Construction of schemata is
entirely hierarchical (or incremental or inductive) and follows approaches known for (hierarchical) first-order predicate logics. Construction is restricted to 3 or 4 or more constructors (entity, attribute, relationship types; additionally cluster types). Schema semantics is canonically defined. Hidden set semantics is used with implicit pointer semantics for relationship and cluster types. Generalisation and specialisation of all kinds are reflected through specific subtype or grouping (clustering) constructs. The manifold of specialisations is separated. Semantics is static. All schema elements are completely defined. Explicit semantics is defined through constraints which might however require treatment beyond (canonical) first order predicate logics. Viewpoints are defined through views on top of the schema definition via algebraic expressions. Derived attributes are defined via algebraic expressions. Algebra is restricted to terms that can be constructed for the algebra operations. Expressions may be generically defined with structures as parameters, e.g. insert(type) as generic operation.

Classical development methods are based on the kind of ER schema and view construction. They include methods for stepwise incremental construction, extension, decomposition, design, validation, and evaluation (see [42]). We may use a number of methodologies, such as top-down, bottom-up, modula, inside-out, and mixed. Classical utilisation actions and resulting methods are mapping and transformation methods (see [42]). Methods for integration, calibration, verification, control, reconfiguration, migration, and evolution are still under investigation.

The profile is restricted to the system construction function for mediating models.

6 Concluding: Models and Model-Based Engineering

Model-driven engineering and development has become an area of intensive research. Roles, limitations, background and directives of the model have however not been taken into consideration. In the past, panels often discussed which modelling approach and which modelling language is most appropriate. We realise now the models and also modelling languages have their own obstinacy. So, model-based engineering is background-laden and directives-laden.

Model-based engineering is based on the modelling know-how, on modelling practices, on modelling theory, and on modelling economics. We discussed the ingredients for model-based engineering for the case of co-models and of mediating models. This approach can be generalised to full co-design of structuring, functionality, interactivity and distribution. So far, the approach uses model suites. How this approach can be extended to any kind of model collections is an open research problem.

The paper has been restricted to the general programme of model-based engineering. The explicit and detailed description is the topic of two forthcoming papers. Model-based engineering uses a number of practices similar to SPICE or CMM approaches [18].
We may now combine our investigation in Figure 4. We distinguish the six dimensions: community of practice, background/knowledge/context, application scenario and stories of model utilisation, situation/state/data, dynamics/evolution/change/operations, and models as representations and instruments. Models are used in a variety of functions. For instance, models of situations/states/data are often used for structuring, description, prescription, hypothetic investigation, and analysis. Models are used by members of the community of practice for communication, reflection, understanding, and negotiation. So, we observe that the function (or simpler the purpose or the goal) of the model is determined by the concrete way how a model is used. Model-based engineering is thus engineering supported by models that are used according to the function that a model might play in the engineering process.

References

General and Specific Model Notions

Bernhard Thalheim

Christian Albrechts University Kiel, Department of Computer Science
Olshausenstr. 40, D-24098 Kiel, Germany
thalheim@is.informatik.uni-kiel.de

Abstract. Models are a universal and widely used instrument in Computer Science and Computer Engineering. There is a large variety of notions of models. A model functions in a utilisation scenario as an instrument. It is well-formed, adequate and dependable. It represents or deputes origins. This conception of the model is a very general one. Based on the notion of a stereotype as a starting point we show that specific or particular model notions are specialisations of the general notion.

1 Models in Computer Engineering and Computer Science

Models are principle instruments in modern computer engineering (CE), in teaching any kind of computer technology, and also modern computer science (CS). They are built, applied, revised and manufactured in many CE&CS subdisciplines in a large variety of application cases with different purposes and context for different communities of practice.

1.1 The Omnipresence of Models in CE&CS

The wide deployment of models is supported by an expansive scientific literature on model usages. There are many different model notions, e.g. [30] discussed more than 50 different definitions of models used in CE&CS programs. All subdisciplines in CE&CS use models such as phenomenological models, computational models, developmental models, explanatory models, didactic models, imaginary models, mathematical models, substitute models, iconic or diagrammatic models, formal models, and analogue models. There is no branch in CE&CS that does not widely use models as instruments.

It is now well understood that models are something different from theories. They are often intuitive, visualisable, and ideally capture the essence of an understanding within some community of practice and some context. At the same time, they are limited in scope, context and the applicability.

We realised also that models become an research issue on their own. Models are expressions, descriptions, icons, statements, etc. from one side and desiderata, representations, deputies, instruments, designs, products etc. from the other side. They might suggest something that we might later be able to explain or to construct. Models also help us to explain a system, help us to deal with more realistic situations, and tell us which intuition and understand is a good one.

How we handle such variety of deployments, understandings, and approaches?
1.2 The General Notion of the Model

There is however a general notion of a model and of a conception of the model:

A model is a well-formed, adequate, and dependable instrument that represents origins. [6, 24–26]

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilisation scenarios.

The model should be well-formed according to some well-formedness criterion. As an instrument or more specifically an artifact a model comes with its background, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form.

The background is often implicit and hidden. Is there any approach to consider the background in a simpler form?

A well-formed instrument is adequate for a collection of origins if it is analogous to the origins to be represented according to some analogy criterion, it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its purpose.

So far, the adequateness notion is far too fuzzy and too wide. Can be develop a simpler notion of adequateness that still covers the approaches we are used in our subdiscipline?

Well-formedness enables an instrument to be justified by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability, and by stability and plasticity within a collection of origins.

The instrument is sufficient by its quality characterisation for internal quality, external quality and quality in use or through quality characteristics [20] such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

A well-formed instrument is called dependable if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

Again, dependability is a wide field. Do we need this broad coverage for models? Or is there any specific treatment of dependability for subdisciplines or specific deployment scenarios?

If there are many specific and particular notions of the model: Can we relate different notions of models with each other? Can be define interfaces among models? Is there any standard notion for a sub-discipline? Are specific or particular notions derivable from the general notion of the model?

And finally, this notion is a very general one. How does the general notion match with other understandings and approaches to modelling in CE&CS? Or more generally for sciences based on Occam’s razor principle: Are there specific or particular notions of the model within specific constellations that sufficiently represent all relevant aspects requested and nothing more?
1.3 Generality versus Specificity

The general notion of a model covers all aspects of adequateness, dependability, well-formedness, scenario, functions and purposes, backgrounds (grounding and basis), and outer directives (context and community of practice). It covers all known so far notions in agriculture, archeology, arts, biology, chemistry, computer science, economics, electrotechnics, environmental sciences, farming, geosciences, historical sciences, languages, mathematics, medicine, ocean sciences, pedagogical science, philosophy, physics, political sciences, sociology, and sports. The models used in these disciplines are instruments used in certain scenarios.

Sciences distinguish between general, particular and specific things. Particular things are specific for general things and general for specific things. The same abstraction may be used for modelling. We may start with a general model. So far, nobody knows what is such general model for most utilisation scenarios. Models function as instruments or tools. Typically, instruments come in a variety of forms and fulfill many different functions. Instruments are partially independent or autonomous of the thing they operate on. Models are however special instruments. They are used with a specific intention within a utilisation scenario. The quality of a model becomes apparent in the context of this scenario.

It might thus be better to start with generic models. A generic model [3, 15] is a model which broadly satisfies the purpose and broadly functions in the given utilisation scenario. It is later tailored to suit the particular purpose and function. It generally represents origins under interest, provides means to establish adequacy and dependability of the model, and establishes focus and scope of the model. Generic models should satisfy at least five properties: (1) they must be accurate; (2) the quality of the generic model allows that it is used consciously; (3) they should be descriptive, not evaluative; (4) they should be flexible so that they can be modified from time to time; (5) they can be used as a first “best guess”.

Generic models might also be an abstraction of other models that are used as an inspiration for development of the new model and that are based on the experience of the modeller. Generic models can be calibrated to specific models through a process of data or situation calibration, refinement, concretisation, context enhancement, or instantiation.

Generic models [29] are typically specialised to more specific ones in a development process. Generic models are widely used under different names or development approaches such as inverse modelling, model-driven architectures and development, universal applications, data mining and analysis, pattern-based development, reference models, inductive learning, and model forensics. All these approaches develop models by stepwise refinement of the root or initial model, by selection and integration of model variations, and by mutation and recombination of the model where the the root model is a generic model with parameters (also structures and operations as parameters as well as the architecture).

Instead, we also may start with general models. Typically, we prefer however particular or idealised models as a starting point for a specific community of practice with a specific background, within a specific context, and for represen-
tation of a specific world of origins under consideration. Generic models can be calibrated to specific models through a process of data or situation calibration, refinement, concretisation, context enhancement, or instantiation. Lightweight models [28] typically cut off background and context. They assume per default some utilisation scenario and reduce the functions of the model to the main function. The purpose is then driven by this function. Often the community of practice is set to some standard community that uses a specific kind of justification.

Therefore, we face the problem: What is the best starting point for development of a model? This paper answers this question by introducing stereotypes of particular models in Sections 3 and 4. For this we use the separation of abstraction into stereotypes, pattern, and templates [1].

1.4 The Storyline and Objectives of This Paper

Since the model notion is too broad we might ask ourselves whether more specific notions can be used in subdisciplines of CE&CS. We might also consider whether some of the proposed notions are simpler and better to use. We might start with the main properties of models (mapping or analogy, truncation or abstraction or focus, pragmatic, amplification, distortion, idealisation, carrier, added value, purpose [12, 16, 18, 21]) and specialise them. We might also discuss the variety of notions [23, 30] and compare them with the general one. The main question is however whether these different notions are sufficient within their environment, i.e. which specific notion of the model is sufficient for which utilisation, for which community, within which context, under which general conditions and within which understanding.

Models are used as perception models, mental models, situation models, experimentation models, formal model, mathematical models, conceptual models, computational models, inspiration models, physical models, visualisation models, representation models, diagrammatic models, exploration models, heuristic models, etc. Although this categorisation provides an entry point for a discussion of model properties, the phenomenon of being a model can be properly investigated. Each category is too broad and combines too many different aspects at the same time.

We thus first discuss notions which are commonly accepted and discover that these notions are laden by background, community, context, and utilisation scenarios. This ladenness can be represented by definitional frames for the model notion. These frames may now be used for defining stereotypes of model notions.

2 Specialising and Refining the Model Notion

2.1 Stereotypes for Models and Particular Notions of Models

Modelling stereotypes describe the general modelling situation. Generic models are typically a general modelling solution in a certain utilisation scenario, context, background, and community of practice. For instance, a structure stereotype.
describes data structuring environment within a certain modelling situation. The corresponding generic models can be refined and used during model development. They can be considered to be classes or collections of potential models.

2.2 Two Model Notions and their Specific Approaches

Let us consider two of the 49 model notions we collected [27] for CE&CS. We will show that these notions are applicable but are heavily biased and thus paradigmatically use a lot of latent semantics behind.

Models for Model-Based Development. The Scandinavian and Dutch schools of (conceptual) modelling have developed a sophisticated approach to modelling since the late 60ies. One result is the famous FRISCO report [7]. More recently, J. Krogstie [11] states:

“Model: A model is an abstraction externalised in a professional language. A model is assumed to be simpler than, resemble, and have the same structure and way of functioning as the phenomena it represents.

Phenomenon. A phenomenon is something as it appears in the mind of a person. The world is perceived by persons to consist of phenomena. …

Property. A property is an aspect of a phenomena that can be described and given a value. A phenomenon will have a set of potentially relevant properties. …

Constitutive rule. A deontic rule that applies to phenomena that exist only because a rule exist. …

Professional language. A professional language is a language used by set of persons working in certain kind of area or in a scientific discipline. Usually such a language is not learned before the person has been active in the area for a while.

Language model. The model of a language. Within conceptual modelling, this is often termed ‘meta-model’, which is only a proper term when looking upon it from the point of view of repository-management for a modelling tool where the instantiation of the model is another model in the same or a different modelling language.

Conceptual model. A model of a domain made in a formal language or semi-formal language with a limited vocabulary.

System. A system is a set of correlated phenomena, which is itself a phenomenon. …

System model. A model of a system.”

Analysing these notions and more specifically the notion of the model, we realise that there must exist an origin that we can call matured perception model. At the same time, the modelling approach is entirely biased by its discipline, its school of thought, its context, and - as a partially explicit component - its community of thought. At the same time, we consider only phenomena in a set-based fashion and not within a conception/conception network. So, the modelling approach is using a rather restricted world view.

This restricted world view is however entirely sufficient since the model is used in one very specific utilisation scenario: system construction. We observe the import of latent paradigmatic (computing-oriented, function-backed, economic, ...) models with predefined meaning, specific context and background
concepts (space, time, settlement, environment, ...) within this scenario. The main function of the model is that of a mediator that describes the (augmented and perceived) model and that prescribes a system to be investigated or perceived. The adequateness property uses homomorphisms.

This approach is typical for model-based (software) development [4, 10, 11, 17] within the specific consideration of specific platform-independent models such as conceptual models and of platform-dependent models as refinements of the generic ones. This approach uses latent hidden generic models as community knowledge. Beside the community dependence, the development biases are also latent in this model notion.

Model Notions with Justification. Extending and revisiting the model notion with its mapping, truncation and pragmatic properties by H. Stachowiak [16], R. Kaschek [9] introduces a model as a material or virtual artifact (1) that is called a model within a community of practice (2) based on a judgement (3) of appropriateness for representation of other artifacts (things in reality, systems, ...) and serving a purpose (4) within this community. Already [9] discussed the forgetful development of software products. Classically we observe that (i) developers base their design decisions on a “partial reality”, i.e. on a number of observed properties within a part of the application, (ii) developers are developing the information system within a certain context, (iii) developers reuse their experience gained in former projects and solutions known for their reference models, and (iv) developers use a number of theories with a certain exactness and rigidity.

The design decisions made during the design process are deeply influenced by these four hidden factors. In some approaches revisions made during the information systems development are recorded. However, since the background knowledge is not recorded the documentation of the information systems development is fragmentary.

The justification of models [9] is here explicit. It should however be combined with a statement of quality that has been achieved so far. The quality criteria are implicit. The model notion [9] is based on the community of practice behind the model. Forgetful development is one of the specific properties. The community of practice drives the context of the model and of modelling. At the same time, appropriateness is more general than (homomorphic) mapping and truncation.

2.3 The Background as the Hidden Component of Models

The two cases show that the model notion is often laden by its specific background. The background consists of undisputable elements (grounding: paradigms, postulates, restrictions, theories, culture, foundations, commonsense) and disputable one (basis: concepts, foundations, language as carrier, assumptions, thought community, thought style, conventions, practices). Background laden models are already using the grounding and the basis without making it explicit.
2.4 The Particular Notion of a Conceptual Model

Conceptual models are nothing else as models that incorporate concepts and conceptions which are denoted by names in a given name space. A concept space\(^1\) consists of concepts [13] as basic elements, constructors for inductive construction of complex elements called conceptions, a number of relations among elements that satisfy a number of axioms, and functions defined on elements.

The general Sapir-Whorf hypothesis [33, 36] states the principles of language determinism (the language governs thinking) and language relativity (coded distinctions made in one language might not be expressible in another language). The weak form refers to the dependence of perception, remembering and simplicity on language. We may transfer this hypothesis to concept-ladenness of languages. Some languages might have richer concepts and conceptions than others\(^2\). Therefore, concepts and conceptions that are expressed in certain language heavily influence semiotics of models since the basis of models is also concerned with concepts and conceptions to be used and thus related to the the (discipline’s context).

They use a specific background: a concept space that clarifies the meaning of the elements of the model. The concept space is often application dependent and based on the understanding of notions in the application area. The linguistic meaning of designators and annotations is an inherent but hidden element of the

So, we notice: the conceptual model is concept space laden.

2.5 The Ladenness of Model Notions

In a similar way we observe also other kinds of ladenness:

**Context-ladenness:** The application domain and disciplinary context is often already given due to the introduction of the model. It is often enhanced by focus and scope depending on the concrete deployment of the model. The time and space issues are typically implicit.

**Community-ladenness:** A community of practice tries to be efficient. Such kind of efficiency includes an agreement of the way how thing are considered, i.e. a “school of thought” and commonly accepted practices, conventions, and assumptions.

**Development- and utilisation-ladenness:** Models must function effectively within the utilisation scenarios. For this reason, a number of biases are inherited by the the model notion due to the orientation and function of the model. Utilisation also determines most of the quality characteristics, the assessment of the model, and the tolerance that might be applied.

---

\(^1\) We follow R.T. White [24, 35] and distinguish between concepts, conceptual, conceptions.

\(^2\) Think for instance about the finer notions for whole in Aborigine language: yarla, pirti, piruki, kartalpa, yulpilpa, mutara, nyarrkalpa, pulpa, makarnpa, and katarta.
2.6 Lessons Learning: Towards a General Approach to Modelling

We observe that modelling mainly consists of three macro-steps, two intentional and implicit and one extensional and explicit:
(I) Setting the definitional frame with priming, language, and actor setting: Priming defines the undisputable decisions (called grounding), the concept space, and the context. Actors within a community of practice act in certain roles while fulfilling a task. They are biased by their disputable but somehow accepted back-ground or basis.
(II) Choice of a model stereotype consisting of accepting the definitional frame, of agenda setting, and of initialisation pattern: Agenda setting restrict potential utilisation scenarios of models. It thus results in a clarification of the model functions and thus also purpose and goal. Initialisation may be based on generic models or modelling experience, e.g. on the basis of reference models.
(III) Model development and deployment is the classical macro-step and well investigated for many modelling problems.

The two first intentional macro-steps are hardly often explicitly mentioned. We often use already existing models (generic, reference, perception, situation, documentation, etc.) as a starting point without making a reference to it.

3 Definitional Frames for Model Notions

Definitional frames are often somehow agreed practice and commonsense within a context and within a community of practice. They are somehow implicit. Without knowing and managing them we might however come-up with models that drive us to spurious results or pitfalls. This paradox is well known for natural sciences or economics. Disputes in the past on whether semantical modelling, object-role modelling, relational modelling etc. are based on a misunderstanding of the definitional frames that have been used.

3.1 Priming and Orientation

The model is mostly developed within some context of a discipline, an application area, and an environment such as an infrastructure. Context may also incorporate certain foci and scopes for the model. Context may also be concerned with time. The context is taken as granted and not questioned.

Models are instruments and therefore design for utilisation. That means they are also set into the existing world. This world is based on some fundament or grounding. The grounding consists of the commonly accepted and not disputed postulates, paradigms, restrictions, theories, culture, foundations, and commonsense. Models thus inherit this grounding and do not explicitly refer to this grounding.

Models represent origins. These origins bring in their own world view, their own concepts and conceptions. The concept(ion) space is therefore for models some referred background. It is used for assigning a meaning to the constructs of the model, for consideration of properties of the model, and for validation of the
model. Therefore, models often use concepts either in an explicit form (becoming thus conceptual models) or in a reference form as abstract formal notations which provide potentially an explanation of the model and its elements (most often for formal or mathematical models). In the first case, the concept space is given and not disputed whereas in the second case the concept space is hidden but available upon demand.

The fourth component of priming is the context agreement. It integrates the application domain, the specific thoughts in this application and thus the disciplinary context, scope, focus, infrastructure and time. We answer the when, whereat, whereabout, wherein, where, for what, wherefrom, and whence questions, may be partially also the what question.

3.2 Actors

The community of practice is far more influential than typically assumed. Community members play their specific roles, have their task portfolio, responsibilities and obligations during a development process. They have however also their interests which are injected into the modelling decisions. They have their preferred method spectrum and neglect others [2]. So, they choose also the modelling language [32] with all the limitations and potential of the language.

A community of practice is typically not interested in revision of the grounding. The community agrees typically also on the basis, i.e. on assumptions, on the thought style and understanding, on practices, and on conventions within the setted definitional frame. That means the community of practice determines the background meaning of a model and adequateness and dependability. The community also has a hidden raw understanding what means that a model is well-defined, analogous, focused and purposeful. A similar raw agreement is already made on dependability, i.e. on justification and sufficiency. The corroboration, rational coherence, validation, stability and plasticity is somehow already generically set and taken as commonly agreed.

So, we need to question the influence of the social and professional community: whom (to whom, by whom), whichever. These questions answer to the presetting of the model. The message of the model is the same within the community.

3.3 Languages and Basics

Languages enable and restrict at the same time [33, 36]. They have their own obstinacy and thus restrict representability. From the other side, the provide rules for well-formedness, especially for syntactic ones. Professional languages additionally provide rules for semantic well-formedness. The community of practice also introduces its rules of pragmatic well-formedness. So, the language supports ‘beauty’ of models due to the inherent phonetics. We known from audience theory that representation determines later thinking, usage, and understanding of a model.

For instance, ER modelling supports well a global-as-design procedure on the basis of a global conceptual schema. If we follow the approach that syntactics
also determines the operations and the algebra [19] then the different viewpoints which are required by the business user can be expressed via view collections defined on top of the conceptual schema.

Due to the carrier property, the language enables also to adjust practices, methodologies, pattern, typical routines, and commonsense. These elements of the basis complete the background. The language has a symbolic level which forms the culture of its users and provides a meaning. Professional languages use denotations and connotations. They provide a code that professionals learned to read.

So, the language answers the wherewith question. The imposed basics answer the question with what means. The language and the background form together some kind of ‘gatekeeper’ since we implicitly decide what to represent.

4 Stereotypes of Models in Utilisation Scenarios

Stereotypes of modelling have already been considered in discussions on methodologies, e.g. [5, 8, 14]. Typically, a methodology is bound to one stereotype and one kind of model within one utilisation scenario. We can however be more flexible. Stereotypes are governing, conditioning, steering and guiding the model development. They determine the model kind, the background and way of modelling activities. They persuade the activities of modelling. They provide a means for considering the economics of modelling.

4.1 Starting with Completing the Definitional Frame

The potential definitional frames are either selected on convenience or after consideration of appropriateness. Often one frame is taken for granted in most IT modelling approaches. The definitional frame sets up the acceptable background of a model. It is typically implicit.

4.2 Model Utilisation Scenario

Models are used as instruments in some utilisation scenarios. They have a number of functions in these scenarios. Based on an understanding of these functions we know what is the goal and purpose of such models. Therefore, we can now define the profile (goals, purposes, functions) that a model must fulfill. Due to the instrument property we also know which tasks are going to be solved with instruments. That means we know the task portfolio.

The profile and the portfolio create the ‘spin’ of the model since they convey a value judgement that might be immediately apparent and they create inherent bias by setting of the modelling task. The spin attempts to steer the way a model becomes useful to others.

4.3 Agenda Setting

Finally, we can define what is the agenda of the modelling tasks and of the model deployment. The agenda setting answers the why, for which reason, wherefore,
worthiness, and whither away questions. This agenda can be formalised as a protocols setting and an orientation behind the model.

Based on the agenda, we sketch also adequateness and dependability. We can determine what means that a model is well-formed, which analogy or similarity is going the be used, which kind of focus allows to restrict the modelling task, and what means to be purposeful for a model.

At the same time, we have set up the main justification approaches. We already know explanatory statements and viability for the elements of the model based on the profile and background. We can sketch the arguments that support the model. We reflect norms and standards accepted by the community of practice, e.g. common practices for achieving inner coherence. The validation procedure is already set up for the model. We also may use which kind of robustness the model must have in order not to be over-fitted.

The model must not represent anything what might be representable. We know in this pre-setting which quality characteristics for quality in use, external and internal quality must be observed and which ones can be neglected. The quality characteristics are enhanced by evaluation procedures. So, we already define which discrimination is tolerated, which modality (necessity, contingency or possibility, relativity) can applied within the context, and which confidence of the evaluation is necessary.

Justification and sufficiency form our criteria for dependability. We can define for the model that is intended to build what means to be admissible, rigid, right, and fit.

4.4 Initial Model Setting

Models represent their origins. We might start from scratch, explore origins, discover essential and relevant elements, decompose them and explore then the modelling task. A first (nominal) model is the result of a composition or amalgamation step. Model formulation results then in development of a model. We might also base modelling on already existing models either for a given system or on the basis of referential models. We might also start with a generic model. In all these cases, we are already conditioned by the definitional frame. Additionally, we selected a modelling workflow or development strategy [19, 22].

The initial setting also inherits latent models that come with the grounding, the context, and the basis.

After setting the stereotype, we start with model development according to the chosen strategy within the agenda and the definitional frame. The typical questions answered in this step are: whereof, how, what, with which restrictions. Additional questions are concerned with adequateness and dependability of the model especially with quality characteristics.

4.5 A Test Case for the Approach

We might consider all notions in [27, 31]. Let us only consider the construction scenario for IT systems. The stereotype we shall use incorporates (1) the typical and also specific IT grounding with all its paradigms, postulates, theories,
foundations, culture, commonsense, and restrictions, (2) the mediator function of models in the construction scenario, (3) the IT community of practice with its obligations, interests, tasks portfolio from one side, and the biases accepted in the community such as school of thought, practices, commonsense, and assumptions, and (4) the selection of the languages and concept space that might be used. It also provides a collection of reference models as their basis for opportunities. These reference models are latent models.

So, in this case, the modelling case is based on the needs and the functions a model might play in system construction. The context is given by current IT systems, current infrastructures and by system development foci and scopes. Therefore, IT grounded is not reconsidered. The choice of the concept space is determined by the notion of the system. The community of practice determines the language and the biases the community likes. The agenda is a mediating one. The model is used either for description of a development idea and for prescription of a forthcoming system or for documentation of an existing system. Initialisation might be based on generic models, on reference models or on already existing models.

Then we arrive with some model definition as [34]:
“A model is a simplified reproduction of a planned or real existing system with its processes on the basis of a notational and concrete concept space. According to the represented purpose-governed relevant properties, it deviates from its origin only due to the tolerance frame for the purpose.”

5 Concluding: Stereotyping as the Spinning Principle

Models are one of the instrument in sciences, engineering and every life. They are not yet properly understood for their way of functioning, their impact, their potential, their capacity, and their anti-profile (not-supported utilisations). We do not want to overload the notion. Models should be used and understood. Therefore, we need a notion that is as simple as possible in the given scenario and given situation. At the same time, we should not loose the specific agreements we have made for models. Models must be effective, efficient, user-friendly, economic, and well-organised. Otherwise, nobody can properly use the conclusions and results that have been generated by the help of models. Sometimes, models may mis-orientate, condition, biase or persuade [23] users in their understanding and must be corrected after paradigmatic revision and synthesis.

So, we need a general specification of the model kind that allows from one side to reason on the potential, capacity, adequacy, and dependability of the given model and from the other side to be aware of the anti-profile and the cases in which the model is not promising, not adequate, may direct to wrong conclusions, and has its pitfalls.

This paper uses definitional frames and stereotypes for a holistic treatment of models. From one side, the model notion covers all what is necessary. From the other side, the specific agreements have to be explicitly given and must not be guessed. So providing the stereotype allows to understand the model, its quality characteristics, its capacity and its potential. It also allows to understand in
which cases the model is not useful or more explicitly to know in which cases
the model should not been used.

This paper does not claim that existing models or model notions are bad. We
cannot handle here the large variety of modelling techniques. Model management
is out of scope of this paper. Instead, we contribute to general model theory and
harmonise notions of models by development of an approach that allows to derive
specific notions of a model from the general one and thus to inherit investigations
made for one model notion by other approaches to modelling.

References

1. B. AlBdaiwi, R. Noack, and B. Thalheim. Pattern-based conceptual data mod-
elling. In Information Modelling and Knowledge Bases, volume XXVI of Frontiers
2. R. Berghammer and B. Thalheim. Wissenschaft und Kunst der Modellierung: Mod-
elle, Modellieren, Modellierung, chapter Methodenbasierte mathematische Modell-
3. A. Bienemann, K.-D. Schewe, and B. Thalheim. Towards a theory of genericity
Advanced Institute of Science and Technology Press, Ishikawa, 2009.
7. E. D. Falkenberg, W. Hesse, P. Lindgren, O. J. L. Han, C. Rolland,
IFIP, ifip@ifip.or.at, 1998.
8. C. C. Fleming and B. von Halle. Handbook of relational database design. Addison-
Wesley, Reading, MA, 1989.
Habilitationschrift.
10. A. Kleppe, J. Warmer, and W. Bast. MDA Explained: The Model Driven Archi-
12. B. Mahr. Information science and the logic of models. Software and System Mod-
15. G. Simision and G.C. Witt. Data modeling essentials. Morgan Kaufmann, San
Francisco, 2005.
16. H. Stachowiak. Modell. In Helmut Seiffert and Gerard Radnitzky, editors, Hand-
extikon zur Wissenschaftstheorie, pages 219–222. Deutscher Taschenbuch Verlag
17. T. Stahl and M. Völter. Model-driven software architectures. dPunkt, Heidelberg,
2005. (in German).


Normal Models and Their Modelling Matrix

Bernhard Thalheim

Department of Computer Science, Christian Albrechts University Kiel, D-24098 Kiel
thalheim@is.informatik.uni-kiel.de

Abstract. Models are one of the central instruments of modern Computer Science and Computer Engineering. The notion of the model is however not commonly agreed. There are - from one side - very general and universal notions and - from the other side - rather specific ones which are easy to use within some focus and scope and fail to be applicable in other sub-disciplines. Model development is typically based on an explicit and rather quick description of the ‘surface’ or normal model and on the mostly unconditional acceptance of a deep model. We discover that model development is based on stereotypes. The basis of a stereotype is the deep model which is tacit and latent knowledge in normal models. The deep model is the ‘logos’ of a normal model. The scenarios of model deployment and the functions the model plays in these scenarios are tacit and latent engineering in normal models.

Keywords: model, model notion, conceptual modelling, (modelling) matrix, deep model, normal model

1 Models, Models, Models: Everywhere but Different

1.1 101 Notions of the Model Concept

Computer science and computer engineering expressively use the conception of model for daily work. Modelling is one of their four central paradigms beside structures (in the small and large), evolution or transformation (in the small and large), and collaboration (based on communication, cooperation, and coordination). E.g. [69] selected 35 of notions which are commonly used in business informatics. As a very short list we may consider the following statements:

[3]: A model is a mathematical description of a business problem.

[4]: A model is the result of a construction process for which the selected part of the origin satisfies the purpose.

[7]: A model is the representation of an object system for the purpose of some subject. It is the result of a construction process by the modeller who addresses a representation of these objects for model user at a certain time and based on some language. A model consists of this construction, the origin, the time and a language.
A model can be simply considered to be a material or virtual artifact which is called model within a community of practice based on a judgement of appropriateness for representation of other artifacts (things in reality, systems, ...) and serving a purpose within this community.

A model is an abstraction externalised in a professional language. A model is assumed to be simpler than, resemble, and have the same structure and way of functioning as the phenomena it represents.

The model prescribes concepts as a particular kind of relation relating a subject and an entity.

A model is an object that has been developed and is used for solution of tasks which cannot be directly solved for the origin by a subject because of its structural and behavioural analogy to an origin.

Models are governed by the purpose, are mappings of an origin and reflect some of the properties observed or envisioned for the origin. They use languages as carrier.

A model is a simplified reproduction of a planned or real existing system with its processes on the basis of a notational and concrete concept space. According to the represented purpose-governed relevant properties, it deviates from its origin only due to the tolerance frame for the purpose.

The following general notion in [66] has been combined and generalised the understanding of the concept of a model in Archeology, Arts, Biology, Chemistry, Computer Science, Economics, Electrotechnics, Environmental Sciences, Farming and Agriculture, Geosciences, Historical Sciences, Humanities, Languages and Semiotics, Mathematics, Medicine, Ocean Sciences, Pedagogical Science, Philosophy, Physics, Political Sciences, Sociology, and Sport Science.

**Definition 1** [14, 47, 62, 64, 67] A model is an instrument that is adequate and dependable. It has a profile (goal or purpose or function), represents artifacts and is used for some deployment scenario. As an instrument, a model has its own background (e.g. foundation (paradigms, postulates, theories, disciplinary culture, etc.) and basis (concepts, language, assumptions, practice, etc.)). It should be well-defined or well-formed.

Adequacy is based on satisfaction of the purpose, analogy to the artifacts it represents and the focus under which the model is used. Dependability is based on a justification for its usage as a model and on a quality certificate. Models can be evaluated by one of the evaluation frameworks. A model is functional if methods for its development and for its deployment are given. A model is effective if it can be deployed according to its portfolio, i.e. according to the tasks assigned to the model. Deployment is often using some deployment model, e.g. for explanation, exploration, construction, description and prescription.

### 1.2 Models as the Third Dimension of Science

Models have been considered to be somewhere in the middle between state of affairs (world, situations, data etc.) and theories (concepts and conceptions,
statements, beliefs, etc.) since they may describe certain aspects of a situation and may represent parts of a theory. Figure 1 displays this understanding.

```
<table>
<thead>
<tr>
<th>First dimension</th>
<th>Second dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>State of affairs, ...</td>
<td>Theories</td>
</tr>
<tr>
<td></td>
<td>Models as mediator between state of affairs and theories</td>
</tr>
<tr>
<td>Models representing states of affairs, phenomena, situations, ...</td>
<td>Models as representations of theories</td>
</tr>
</tbody>
</table>
```

**Fig. 1.** Models between state of affairs and theories

“Models are partially independent of both theories and worlds.” [39] The understanding of a model to be a mediator between a world and a theory is however far too restricted.

Models should be considered to be the third dimension of science [8, 66, 68] as depicted in Figure. Disciplines have developed a different understanding of the notion of a model, of the function of models in scientific research and of the purpose of the model. Models are often considered to be artifacts. Models might also be mental models and thought concepts. Models are used in utilisation scenarios such as construction of systems, verification, optimization, explanation, and documentation. *In these scenarios they function as instruments.*

![Diagram](image.png)

**Fig. 2.** Models are independent and are the third dimension of science

---


2 An instrument is among others (1) a means whereby something is achieved, performed, or furthered; (2) one used by another as a means or aid or tool [45].
Given the utilisation scenarios, we may use models as perception models, mental models, situation models, experimentation models, formal model, mathematical models, conceptual models, computational models, inspiration models, physical models, visualisation models, representation models, diagrammatic models, exploration models, heuristic models, informative models, instructive models, etc. They are a means for some purpose (or better: function within a certain utilisation scenario), are often volatile after having been used, are useful inside and often useless outside the utilisation scenario. Models are different in the four generations of science (empirical science, theory-oriented science, computational science, data science) [17].

1.3 Kuhn’s Conception of Normal Science

T. Kuhn realised that the history of science consists of normal science punctuated by periods of revolution. He combined the underpinning of normal science by a notion of paradigms. His notion of paradigms substantially and circularly changes within his work. He did not get a satisfying definition for it. He integrated normative and empirical disciplines and got a general picture on how science works.

Until his work, philosophers of science made some assumptions about how science works, because they were confident in the methods of science and its success. However failures and irrationalities were not explainable. Kuhn suggested that science can only be understood “warts and all”. The main conception in his book [27] is the distinction between normal science and (revolutionary) evolving science. Normal science is strict and governed by what he called paradigm as an object of consensus within a community and context. Education in such sciences is governed by success stories, e.g. examples, and is not governed by rules or methods. So, it is far more dogmatic [9, 16, 27]. Dogmatism, brainwashing, and indoctrination have the advantage of simplicity, fruitfulness, parsimony, and understandability within certain community and are thus enablers of success. Normal sciences maintain confidence within this community. They condition the members of a scientific community.

The investigations Kuhn has made can also be observed for modern natural sciences. Modern physics is partially normal science. It uses standard models. For instance, astrophysics [19] is based on the Lyndon-Bell hypothesis that assumes the existence of a black hole in galaxies with accretion layers around the black hole with broad and narrow line regions it as the central energy source. The “inner” or “deep” model behind defines the space of possible models. The modelling approach develops the final model based on a composition approach starting with parametric model fragments. Each fragment is conditioned on

3 Later [29], he revises this notion to a ‘disciplinary matrix’. His understanding of the matrix combines a wider notion of commonsense within a community of practice for relatively unproblematic disciplinary communication and for relative common agreement. We shall turn to this notion and clarify what it means in this paper. Our clarification follows [30].
a background, i.e. a grounding and a basis, for instance, for approximations, perspective, granularity, and operating assumptions underlying the fragment. Mutually contradictory backgrounds are organised into background groups, and a set of coherence constraints is used to govern the use of the backgrounds [40]. Techniques such as constraint satisfaction, (causal, fitting, ...) approximations, conflict resolution graphs, and inverse modelling are then used for construction of an adequate and dependable model. Typical other deep models are the standard big bang model and the standard model of particle physics.

1.4 The Usefulness of a General Notion of the Model

The work [66] offered a rational reconstruction of the notion of a model. Definition 1 introduces an explicit and clear the full notion of the model, the reasons behind modelling, modelling decisions, and modelling practices. This work follows the positivist approach by [44, 43] and sees modelling as an a priori engineering discipline. Additionally, models have also an explanatory value. Models and modelling methods have their own obstinacy. They widen the understanding, theory, and engineering within an area and restrict at the same time.

We realised that models may be artifacts or mental things. Models are however going to be used in a certain way. This way can be stereotyped as a scenario. Models function within such scenarios. This functioning explains then what is the purpose and the goal of the model itself. Taking this turn, models are then instruments as an element of technology.

1.5 Outline of the Paper

Modelling does not start from scratch. Rather it uses previous experience, approaches developed so far, commonalities agreed within a community of practice, disciplinary and other context, and also a consensus within an application. Therefore, it is not ‘greenfield’ work. It might be ‘brownfield’ work for migration or modernisation project. In all cases, modelling is laden by its grounding, its basis, its specific application scenarios, its community of practice, and its context. This laddeness can be understood as the deep model or the ‘logos’ underneath the normal model.

The main issue of this paper is the development of an understanding of the matrix of conceptual models. Matrices consist of deep models and modelling scenario setting. Normal models are governed by such modelling matrices.

4 The word ‘normal’ has different meanings [1]. It seems that Computer Science and Logics prefer ‘normal’ as conforming or constituting a norm or standard or level or type or social norm. We prefer the meaning as being appropriately average or within certain limits or occurring naturally or being characterised by average development. Being ‘normal’ also means to be in accordance to accepted consensus or rules or laws.
2 Normal Modelling

Normal modelling is the day-to-day business of modelling in most areas and also in Computer Science and Computer Engineering. The matrix introduced in the sequel can be seen as a pervasive, disciplinary and well-accepted framework in which modellers perceive and develop their model. Normal modelling is concerned with model development and model deployment for the given application task and nothing else beyond that.

2.1 Modelling is Often Stereotyped

The modelling process is often stereotyped [65]. It reuses experience gained in the community of practice and especially by the modellers. It is additionally governed by puzzles and expectations and especially the origins with the selected concept space. This approach assures modellers that each model is adequate and dependable, and provides standards for evaluating its adequateness and dependability. It uses a definitional frame, is based on certain modelling situations - or better on certain modelling scenarios - that determine the agenda of the modelling process, and can be started from scratch or with an initial model, e.g. a generic model.

The definitional frame defines the setting of the modelling process, i.e. (1) its priming and orientation that is governed by the context (application domain or discipline, school of thought, time, space, granularity, scope) and the grounding (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities), (2) the actors (which form its community of practice) with their roles, responsibilities, and obligations, and (3) the language (as a carrier) and basics (assumptions, concepts, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense). Another part of the definitional frame is defined as specific adequacy and dependability criteria which are applied to a model.

2.2 Modelling is Mainly Normal Modelling

We will realise in the sequel that such kind of stereotyped modelling uses two models behind the model. It defines a macro-model for the modelling process itself, i.e. the way how the model and especially the surface model is going to be developed. It is also the basis for the deep model that directs the modelling process and the surface or normal model. The deep model can be understood as the common basis for a number of models. Education on conceptual modelling starts, for instance, directly with the deep model. In this case, the deep model has to be accepted and is thus hidden and latent.

Normal modelling is similar to normal science. It is based on some kind of consensus about how modelling should be done. It is governed by some implicit knowledge, by commonsense, and - more generally - by consensus behind that we call in the sequel ‘matrix’. It is thus puzzle-solving. Normal modelling is
what modellers do most of the time. A typical puzzle-solving task is the development of a conceptual schema for a given application within a given business context for a given community of practice, and within a given system orientation. The next puzzle is then solved by the next conceptual schema. Textbooks and education mainly develops puzzle-solving skills.

In normal modelling, the modelling theories, modelling tools, modelling attitudes, and modelling assumptions comprise the modelling matrix. They are kept fixed, permitting the cumulative generation of puzzle-solutions. The modelling matrix undergoes revision whenever the underlying technology, the context, or the application are changing. If the consensus on modelling is lacking then competing schools of thought possess differing procedures, theories, even practices. Normal modelling proceeds on the basis of perceived similarity to exemplars.

Education and also edification is governed more by examples than by rules or methods. E.g. the field of conceptual modelling is mainly taught on the basis of examples; even more: nowadays on the basis of toy examples. In daily practice, models should be used and understood. Therefore, we need a notion that is as simple as possible in the given scenario and given situation. At the same time, we should not lose the specific agreements we have made for models. Models must be effective, efficient, user-friendly, economic, and well-organised. Otherwise, nobody can properly use the conclusions and results that have been generated by the help of models. Sometimes, models may mis-orientate, condition, bias or persuade users in their understanding and must be corrected after paradigmatic revision and synthesis.

The orientation on normal models has also its pitfalls. For instance, cardinality constraints [56] have mainly been developed for relational technology of the early 90ies or 80ies. At his time, the mapping of these constraints was a deep research issue. Nowadays, object-oriented database system technology allows a far more sophisticated handling of constraints. Maintenance can be deferred (eager or lazy integrity enforcement). Time management allows to handle more optimal timepoints for consistence maintenance. Consistency can be supported at the row level. Integrity constraints can be maintained at the application level. Integrity can be made through views. Finally, flexible strategies may be used, besides the no-action and rollback approach, e.g. on the basis of triggers or stored procedures. Therefore, we may generalise cardinality constraints to conditional cardinality constraints [54].

2.3 Normal Models are Governed by Their Modelling Matrix

A matrix is “something within or from which something else originates, develops, or takes form” [1]. The matrix is assumed to be correct for normal models. Normal modelling involves showing how systems and their models can be fitted into the elements the matrix provides. Most of this work is detail-oriented. So, the matrix governs the modelling process. A failure to solve a modelling task reflects on the modellers’ skills, and not on the legitimacy of the setting.
Normal modelling accepts one notion of a model as normal. It just happens in a broad set of presupposed, unquestioned assumptions that govern among other things the sort of models to be developed, how these models are investigated and deployed, and how these models are interpreted. If the matrix would be question then modelling becomes difficult if not too time consuming. The matrix guides and instructs. Normal modelling is perfectly good modelling as long as the tasks are solved by the models that have been developed. So, modellers can be ‘blinded’ by the success although they are close-minded. The consensus provides a good means for collaboration and minor modernisation. The modelling tasks are focussed on the task spectrum that is preferred at present. So, the matrix got its kind of faith and trust. This kind of ‘brainwashing’ or indoctrination is the basis for teaching.

2.4 Normal Modelling Develops its Boundaries

All matrices have their limit, restrictions and even pitfalls. It might happen that the model does not solve the task in a proper form or that the model is not sufficient or that models develop their obstinacy or that models result in anomalies. Problematic tasks are not counterexamples to normal modelling. In this case, the matrix must be revised since it is not adequate anymore.

The first resolution step is the introduction of new elements to the current matrix. E.g. the entity-relationship modelling language has been heavily extended by about 50 constructs in the 80ies and early 90ies [56] until it has been detected that these extensions will not become coherent. At the same time, work-arounds have been built for overcoming limitations. Currently object-relational technology is available. It seems that ER modelling might somehow suffice with model creation. However, classical theories are not sufficient anymore. The confidence into the ER approach to modelling weakens nowadays. With the advent of object-centred modelling and the supporting XML technology one might ask whether conceptual modelling can be based on the ER matrix. Since data collections might also evolve in their structuring, the class-oriented technique must be nowadays revised what has already been discovered within the research on conceptual modelling for ‘big’ data. Before changing the matrix we must, however, develop a proper understanding of it. We will turn to this problem in the next section.

The transformation of models become a bottleneck whenever the matrixes of the given model and of its transformation do not match properly. Such impedance mismatches have already widely been discussed for object-oriented programming in imperative environments. A similar observation is valid too for conceptual modelling based on sets and physical modelling based on multi-sets and references (see, for instance, SQL with specific referential integrity and multi-set handling).
3 A Case Study: Information Systems Modelling

3.1 Historical Matrices

The matrix is an essential component of the identity within a community of practice or within a scientific community. It identifies puzzles to be solved, governs expectation, assures modellers that each puzzle fulfills its purpose, and provides standards for evaluating.

According to [1], a matrix is “something within or from which something else originates, develops, or takes from”. Normal models will be understood as models crammed with the modelling matrix. The existence of such modelling matrices makes modelling simpler and supports parsimony. It is a kind of complex laddeness of a model [65]. What modellers develop depends, in pertinent part, on what they already believe or expect. Developing is less passive, less receptive than many had thought. Modelling is dependent on the chosen modelling matrix.

Let us consider one example where we observe surprisingly many postulates, paradigms, theories, assumptions, accepted practices, bindings to a school of thought, context, and commonsense. The matrix is well-accepted but not explicitly explained in textbooks and research papers. The entity-relationship modelling language became popular in the late 70ies as a means for documenting logical relational schemata and for visualising the association among types.

The entity-relationship modelling language became now some kind of standard despite the unknown and not explicitly given matrix underlying this language. It uses a Global-As-Design approach where the schema reflects all viewpoints. Local viewpoints are derivable and somehow reflectable. The default semantics (and sometimes the only one to be considered) is set semantics for collections of objects for a type. The reference semantics for relationship types is hidden and not properly understandable during schema development but used in transformation. Explicit existence postulates that any object must exist before there can be a reference to it, i.e. rigid separation of creation and use. The model assumes a closed-world view and unique names. It is based on a well understood name space or glossary or ontology. Salami-slice representation uses homogenous, decomposed types (potentially with complex attributes) with incremental type construction. It is type-centric. According to tradition of logic-based computer science, semantics follows syntax, i.e. the definition of semantics can be given if the syntax is already defined that it uses. The user perspective is cut out, i.e. we base the model on neglected pragmatics. Functionality representation is deferred without consideration of the performance.

---

5 In his oral presentation of his keynote speech at ER96, C. Bachman [5] claimed that the new modelling languages has been introduced as a reaction of the inflexibility and due to the insufficiency of his network modelling language for representation of relational schemata. Due to the popularity of his modelling language it has not been possible for him to publish the new language. His claim has been the basis for the development of a new approach under his supervisorship [12] (reprinted in [14]).
impact to the schema. *Separation into syntax and semantics* allow to define semantics on top of the syntax. Explicit semantics is based on constraints. *Paradigms, postulates, assumptions* of database technology and database support are assumed due to the three main quality criteria (performance, performance, performance). *Basic data types* are hidden until mapping to facilities provided by DBMS typing systems or to logical models. *Visualisation* is represented by one holistic diagram that displays the entire syntax and semantics.

One satyr or misbelief is representation of associations among types on the basis of binary types despite the valence of normal verbs in natural languages. In general, *binarisation* is possible by introduction of abstract artificial types and by relating the new type to each of the association component. The advent of data cubes has shown that an explicit co-handling of views empowers database technology. *Star and snowflake schemata* introduced for data warehouses are nothing else than view in the basis of high-order relationship types.

### 3.2 Puzzle-Solving in Information Systems Modelling

In the sequel, we observe that the matrix forms the implicit and tacit knowledge behind modelling, i.e. the second component of the rigor cycle in design science research. The matrix generates a consensus about how modelling should be done. This consensus distinguishes modelling from other scientific or engineering endeavours.

Modern application with dynamic structuring of objects such as big data collections cannot be properly represented by the static structuring which is one of silver bullet assumptions of DBMS since it provides optimisation facilities that brought the victory of relational technology over network or hierarchical technology.

Puzzle-solving left open a good number of problems for future research. One of the lacunas is the NULL marker problem. It becomes a bottleneck whenever aggregation functions are going to be applied. The representation of NULL-polluted types by a collection of NULL-free subtypes is computationally infeasible. Schema-wide constraint maintenance is another big problem at present.

Education in this area has been built on success stories and proceeds on the basis of perceived similarity to success cases. At present, information system modelling still modelling in the small. Modelling in the large or modelling in the world must be based on different matrices; which ones is not clear yet. Puzzle-solving allows to transfer experience gained for one problem to another class of problems and to evaluate and appreciate solutions of other (e.g. reference or generic models). Design science research is oriented on cumulative addition

---

6 RDF representation has chosen this way on the price of the maintenance and retrieval nightmare. A better binarisation has been used by MIMER resp. RAPID on the basis of sixth normal form storage (called nowadays one-column representation).
of new knowledge in terms of the application of the modelling and designing method.

3.3 Limitations and Pitfalls of Conceptual Modelling

Alike any language, the entity-relationship approach is not covering all issues. It is not cognitive complete since it represents only 2 of 6 cognitive categories (container, link). Pitfalls of this approach are similar to the 88 pitfalls of object-oriented programming [70]. Salami-slice tactics is often not appropriate. Things in the application domain are however multi-faceted. For instance, a human is represented via a Person type that is separated from the Student type etc. The ER language is still based on the approach one-schema-one-diagram approach. Schemata are typically flat. Applications are however structured. For instance, [25] presented a three-dimensional structure of schemata with the application dimension, the volatile workflow data change recording dimension, and the metadata dimension. Additionally one might think of the user involvement dimension within a schema.

The ER modelling language has nowadays also been aging. Object-relational DBMS support features that should be representable at the conceptual level due to their utility. Modern technology provides user defined types, identification trees for components of relationship types and subtypes with overwriting by new surrogate types, flexible view-oriented handling of integrity constraints, indexing mechanism, maintenance of data blocks etc. These features are not representable in the classical ER modelling language but useful for conceptualisation.

The modelling process is far more dogmatic then understood in cookbooks or textbooks. It is somehow ritual or routine-based in education as well in practice. Models are mainly shallow models. They represent a part of the application. The specifics of constraints are not well applied. For instance, cardinality constraints specify the extremal minimal/maximal cases. Users however concentrate on the normal situation.

3.4 Views: The Overlooked Element of Conceptual Modelling

One of the limitations of the ER matrix is neglecting user views and viewpoints we consider now. The misunderstanding of view and viewpoints causes a small crisis in understanding database technology and modelling. The crisis has manifested in the development of data warehouses, data marts, star and snowflake schemata. The latter schemata or else than conceptual or business user views on the database. Currently the approach falls into a lot of difficulties and resulted in the development of a ‘novel’ technology.

As shown in [59], the extended ER language HERM [56] covers 5 of 6 where the last sixth category center-periphery can be represented on the basis of HERM views.

A similar trillion $ mistake is now the evolution of big data. The hype will shred a lot of resources until a new consensus occurs. In order to understand we remember...
The three-layer architecture of information systems is a commonly accepted and widely taught conception. It is however neither true nor useful. It is only a starting point for understanding how a system might work. In reality, user viewpoints come with the application or business user level. They are represented by user schemata. Then these viewpoint schemata are integrated into the conceptual schema in the ‘global-as-design’ and ‘local-as-view’ approach. In order to represent the viewpoints we should use view schemata at the conceptual level what is however not consensus [67]. The logical level turns all the user view schemata to view definitions with the loss of the association between the relational views due to the limitations of relational technology. So far, this is the current state-of-the-art. It is nothing else than an anomaly since this schematology repeatedly resisted solution.

It could be improved with the approach developed in [20]. [67] defines a conceptual model to consist of a conceptual schema and of a collection of conceptual views that are associated (in most cases tightly by a mapping facility) to the conceptual schema. A conceptual schema is then mapped to a collection of logical views.

A database schema could not be anymore seen as an integrated, holistic schema with the same level of detail. Instead, we are able to represent a number of viewpoints at different abstraction level, with different foci and scopus, with different aging and currency, with supporting mechanisms depending on currency requirements, etc. So, the database structure model forms some kind of ‘web’ [23] instead of one schema with derived views. Viewpoints represent structures in whatever order is best for human comprehension and thus expressing it in a stream of consciousness order.

4 The Modelling Matrix

4.1 Deep Models and Scenarios Form the Modelling Matrix

T. Kuhn [29] widely used the notion of paradigm in a variety of forms and explanations. Essentially his notions can be understood as a disciplinary matrix [32], i.e. a symbolic generalisation, a meta-model, and collection of sample cases. Based on the observations on stereotyped modelling we may distinguish four initialisation phases:

(i) orientation on modelling scenarios and used macro-models for development with derivation of the function (and thus purpose and goal) a model has;

the discussions about the relational data model in panels at ER92, ER93, ER95, and ER95. The new consensus was relatively undeveloped. It was not able to represent all modelling situations the network or hierarchical data models could. Later it was realised that the different modelling styles could not been judged on a common scale. All three approaches have some shared habits and ways of seeing things. Proponents of these different approaches tended to talk past each other. Dogmatism and idiosyncrasy function in a complex social arrangement [28] such as conferences and journals.
(ii) acceptance of the grounding and of language and the general concept space;
(iii) setting of a deep model as the hidden, latent model or acceptance of such for some context and a community of practice;
(iv) acquisition of origins for modelling.

Definition 2 The deep model consists of the grounding for modelling (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities), the outer directives (context and community of practice), and basis (assumptions, general concept space, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense) of modelling.

The deep model thus uses a collection of undisputable elements of the background as grounding and additionally a disputable and adjustable basis which is commonly accepted in the given context by the community of practice. It is typically used for many normal models but not explicitly stated whenever a normal model has been stated. The deep model is far more dogmatic than often understood. It is some kind of model ‘logos’ behind the normal model.

At the same time, the deep model is a rich source of knowledge [32, 34] that is already provided by the deep model, i.e. the deep model carries the knowledge and beliefs as well as the culture of the community of practice. It supports communication within the community of practice that accepts the deep model as common ground and has already agreed on the judgements made for the deep model. This common background also includes a common ontology. The deep model provides an identity within this community for the shared ‘correct’ opinion. The normal model becomes an epistemic instrument that is based on the common ground.

Definition 3 The modelling matrix consists of the deep model and the modelling scenarios. The agenda is derived from the modelling scenario and the scenarios.

So, the modelling scenario and the deep model serve as a part of the definitional frame within a model development process. They define also the capacity and potential of a model whenever it is utilised. The normal model can be deployed in a specific form as long as the scenarios and the deep model are not changed. For instance, database structure modelling on the basis of the entity-relationship approach has an ordinary interpretation for all developed schemata.

Different matrixes solve different problems. It might happen that a normal model with one matrix does not make sense if the matrix is changed. A typical case is co-modelling is modelling on the basis of the entity-relationship modelling language for structures and on the basis of BPMN diagrams for processes [68].

Models typically represent a number of origins. It is often the case that these origins use a common application-specific concept space, e.g. an application ontology with its lexicology and lexicography [59]. The application-specific concept space is annotated by a namespace.
A modelling matrix may be enhanced by generic or reference models. Generic models are abstractions of a set of models that represent similar solutions. They are later tailored to suit the particular purpose and function. A generic generally represents origins under interest, provides means to establish adequacy and dependability of the model, and establishes focus and scope of the model. A reference model is used as a blueprint for a fully fledged model and provides a general solution in an application area.

4.2 Adequacy and Dependability Governed by the Modelling Matrix

The modelling matrix allows to derive specific pattern for specification of adequacy and dependability of a model. The general notion of the model in Definition 1 defines adequacy based on an analogy property, a focus property, and purposefulness. Dependability is based on a justification and a quality certificate. Justification is given by an empirical corroboration according to modelling objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability, and by stability and plasticity within a collection of origins. Quality can be defined by characteristics that state the internal and the external quality as well as the quality in use. The certificate is the result of an evaluation of an accepted bundle of quality characteristics through some evaluation procedure.

It seems that the statement of adequacy and dependability is a heavy and sumptuous procedure. In reality it is far more simpler due to the existence of a modelling matrix. The entity-relationship modelling matrix uses, for instance, a homomorphism or infomorphism mapping property for analogy, a focus that is already determined by the situation or perception models or other origins, the purpose of full representation within the language setting, an empirical corroboration due to the mapping from the situation or perception model or other origins, the conformity that is already inherent in the modelling matrix, falsifiability via validation of the model against the origins, and stability against origins as a general class of situation and perception models or other origins. A similar definitional frame can be observed for many model kinds in Computer Science and Engineering.

As already discussed in Section 3.1, the matrix of entity-relationship modelling is quite comprehensive. We are explicitly explicitly using this matrix in dependence on the scenario. For instance, in system construction scenarios: closed-world schemata, Salami slice schemata, methods for simple transformation; adequate for direct incorporation; hierarchical schemata; separation of syntax and semantics; tools with well-defined semantics; viewpoint derivation; componentisation and modularisation; integrity constraint formulation support; methods for integration, variation. In communication scenarios we orient the matrix to: viewpoint and flavour representation; flexible usage (full logical independence); variable name space representation; methods for reason-
This orientation governs the well-formedness criteria such as:

- unambiguous esp. for transformation,
- easy to read,
- aspect-separated, e.g. by colouring different parts,
- naming styles, e.g. either singular or plural,
- higher normal forms,
- optional structure routed to the subtype,
- freeness of semantical cycles,
- distinguishability of attributes, e.g. unique name assumption,
- meaningful names, avoidance of auxiliary verbs, e.g. ‘has’, ‘is’, ..., 
- non-empty classes, and
- flag avoidance.

Central characteristics for well-formed schema are: closed world and unique name assumptions; concept enhancement and well-defined name space; no sharpening or contrasting; well-founded logics; layering of functionality, views and interaction.

The adequacy of eER schemata is based on the following properties for origins $A$ and for the scenario $S$:

1. $A$-analogous: structural analogy (homomorphic, but not qualitative, functional) resulting in structural alignment; metaphysical, epistemological and heuristic adequacy
2. $A$-reduced (or $A$-focused): compactness, no repetition, high-level descriptive abstraction; conceptual minimal
3. $S$-purposeful: either for construction of another representation (thus with construction hints and tactics; with simple transformation; normalised, simple integrity enforcement) or for communication with the (business) user (thus with different viewpoints and flavours; simple viewpoints; cognitive complete).

The focus of eER Schemata is based on the following characteristics:

- Separation into kernel object types, dependent types, and properties: Kernel objects have their own relative existence independence.
- Kernel object types and typical/central types become entity types; properties may be complex and are typically mapped to (complex) attribute types; hierarchies are separated and then represented by generalisation/specialisation hierarchies; relationship types are either application association types, user-relating types, meta-associations, or workflow hocks. This is similar to good practices for E/R/A/C mappings.
- All derivable constructs are represented otherwise. Irrelevant, specific elements are avoided.
- The schema concentrates on important, relevant and typical elements.

---

9 For details we refer to classical database design books [36, 6, 51, 55, 56]
The schema must be as simple as possible, avoid unnecessary abstractions, provide a precise meaning for each type, reduce any complexity on Salami slice techniques, and should combine similar elements.

Rigid incremental schema.

The purposefulness of eER Schemata is given be the following orientation in dependence on the purpose:

Orientation model: corporate overview as a context, data as a source and sink, “environment” model;

Communication model: external schemata depending on the context;

Conceptual model: things of significance, concepts, assertions; semantic model for the business language (“divergent”); architectural model (general categories, “convergent”, platform independent);

Realisation model: based on technology and platform; internal (logical) schemata (platform-specific: relational, XML, ...) (with technological twists), physical schema (storage, (vendor-specific)); and

Documentation model of ground structures used in a given application system.

Scenarios typically combine a number of functions for the ER models. So we might use several schemata and especially view(s) schemata as a model suite [57].

Dependability of eER schemata and models is defined by:

1. Justification is based on embedding the model into the understanding of the application domain, i.e. through an external corroboration. The internal corroboration is based on the language. The origins determine items of the model. So, no additional acquisition and elicitation is needed. The model conforms to standards accepted in CoP, e.g. Salami slice tactics; correctness; restriction to essential business items; approved, closed world schemata; partially evolution prone, partial flexibility; simple diagramming with overlay diagrams.

The model should be cognitive complete based on an appropriate representation of things of interest in real world with some ordering (e.g. hierarchies (up-down, front-back)) and additionally based on other cognitive dimensions (container, part-whole, link, centre-periphery, source-path-goal) [59]. The model is a deputy of relationships of interest in the real world with some ordering additionally based on other cognitive dimensions. We might use additional characteristics of interest for both sides [24, 41, 13].

2. Sufficiency is defined by an evaluation form and by characteristics for internal and external quality and quality of use. Typical criteria are [56]: completeness, naturalness, minimalism, system independence, flexibility, self-explanation, ease of reading and using of firm quality and evaluated. We mainly use quality in use characteristics without any error tolerance. Additionally, we assume (a) avoidance of redundancy (or at least restriction to necessary (controlled)), (b) avoidance imposed implementation restrictions, (c) internal and external characteristics for the usage of the model.
as blueprint without requirement for completeness of constraint sets, (d) natural keys, (d) avoidance of mega-attributes, and (e) complete confidence in all model components.

We notice, that most of the adequacy and dependability characteristics are assumed to be given with any eER schema or model. They are not mentioned but assumed. So, they are a part of the matrix.

A similar definitional frame can be defined for BPMN and other workflow diagrams.

Specific definitional frames are used for adequacy and dependability statements for models

- which provide specific extensions as an amplification which are not observed in the origins,
- which are distortions and are used for improving the origins (e.g. the physical world) or for inclusion of visions of better reality, e.g. for construction via transformation or in Galilean models, and
- which are idealisations through abstraction from origins by scoping the model to the ideal state of affairs.

Therefore, the modelling matrix allows to reduce and to simplify the statement whether a model is adequate and dependable. The reduction also stems from the definitional frame that is already used for the deep model.

4.3 Development of the Normal Model and the Matrix

Education and practice in modelling typically starts with acceptance of a matrix. Whether this matrix is adequate or not is not questioned. So, we can use the modelling matrix and define the specific model in dependence on a function or purpose. Let us now consider, revise and extend the model notions in [67].

**Definition 4** The normal conceptual eER database structure model for communication and negotiation comprises the database schema, reflects viewpoints and perspectives of different involved parties and their perception models. The matrix for communication scenario implicitly links to (namespaces or) concept fields of parties which are partially used. It defines adequacy and dependability based on the association of the perception models to viewpoints and of the viewpoints with the schema. A partial communication model does not use a schema and does not associate viewpoints to schema elements.

As already observed in [67], normal models used for communication and negotiation follow additional principles: Viewpoints and specific semantics of users are explicitly given. The normal model is completely logically independent from the platform for realisation. The name space is rather flexible. The normal model is functioning and effective if methods for reasoning, understanding, presentation, exploration, explanation, validation, appraisal and experimenting are attached.
Definition 5  The normal conceptual eER database structure model for conceptualisation consists of a collection of views for support of business users. The deep model is based on a mapping for schema elements that associates potential elements of the normal model to the common concept field and the perception models of business users. They may be extended by a skeleton that combines business user viewpoints or by a global schema which combines these viewpoints. The matrix uses a strict adequacy and dependability. It is based on a context-driven conceptualisation of the application domain.

Conceptualisation is based on one or more concept or conception spaces of business users. Semantics is typically rather flexible. The normal model and the viewpoint rather reflect the normal cases and do not extend these cases to the extremal cases, e.g. for (cardinality) constraints.

The deep model and the normal model for description can be defined in a similar way. They are representations, refinements and amplifications [58, 63] of situation or reality models and therefore refinements and extensions of the communication model.

Definition 6  The normal conceptual database structure model for description comprises the database schema, and a collection of views for support of business users. The model reflects a collection of a commonly accepted reality models that reflects perception or situation models with explicit association to views, and a shallow declaration of model adequacy and dependability. The deep model and the matrix are driven by the description scenario and completely bound to the understanding in the application area and to technology, methodology and theory which is commonly agreed within the community of practice.

The descriptive normal model reflects the origins and abstracts from reality by scoping the model to the normally considered state of affairs. The deep model also provides an idealisation.

Prescriptive models that are used for system construction are filled with anticipation of the envisioned system. They deliberately diverge from reality in order to simplify salient properties of interest, transforming them into artifacts that are easier to work with.

Definition 7  The normal conceptual database structure model for prescription comprises the database schema and a collection of views for both support of business users and system operating. It is based on a deep model that provides a number of a realisation templates according to the platform capabilities. The matrix declaration of model uses strict adequacy and dependability.

The matrix also defines directives (or pragmas) [2] and transformation parameters [56]. The deep model also consists of general descriptions or templates for realisation style and tactics, for configuration parameters (coding, services, policies, handlers), for generic operations, for hints for realisation of the database, for performance expectations, for constraint enforcement policies, and for support features for the system realisation.

These notions of normal models, deep models, and matrices specialise general notions like those given in the introduction or the notion by W. Steinmüller
A model is information: on something (content, meaning), created by someone (sender), for somebody (receiver), for some purpose (usage context). )[53] or B. Mahr ( “A model is always at the same time a model ‘of something’ and a model ‘for something’. Its function is to ‘carry’ some ‘cargo’ from its ‘matrix’ to its ‘applicate’. ”)[35] or F. Matthes and J. Schmidt (“A relational database model on the basis of the approach by E.F. Codd describes semantics of declarations and statements within a database specification language and thus corresponds to an abstract model of a programming language with its static and dynamic semantics which can be specified through formal type and evaluation rules.”)[38].

Modelling is often ‘brownfield’ work. The model exists already and has been developed based on another matrix. Consider, for instance the schemata in [15, 18, 37, 50, 49, 52]. The schemata follow a certain matrix, e.g. in this case IDEF. Therefore, typical applications combine a number of matrices.

5 Conclusion

5.1 Model ≡ Normal Model ⊗ Matrix

A model can thus be understood as a normal model combined with a matrix and especially its deep model similar to the visible (or surface) and invisible parts of an iceberg. The matrix forms a relatively stable component for a larger collection of models and can be thus neglected for these models. The matrix is considered to be valid and does not thus need a justification. This observation led us to the conclusion that modelling is mainly normal modelling.

5.2 The Model Matrix as the Stabile Ground of Normal Models

We observed that models consist of a normal model and its matrix (or a number of its matrices). The matrix is neither questioned nor a matter of redefinition in a modelling process. It is taken for granted. A special case are ‘brownfield’ models which have a legacy matrix and a current matrix and which may consist of a model suite of mutual models for each of the matrices.

A matrix may evolve as well due to its limitations, revisions of dependability and adequacy required for an application, misconceptions, or missing elements. In our area, we observe changes of the deep model only for cases when technology entirely changes, e.g. the transfer from network or hierarchical modelling languages to the relational ones. The relational environment has changed however as well. So far, it is at its best an evolution step for matrices if at all. Database structure modelling has not changed for more than two decades although technology has changed a lot. Matrix evolution is also caused by changes in the scenarios.

Matrices are relatively stable. Normal models are under continuous change also due to rational and empirical evaluation or due to quality problems, e.g.
validity & completeness, reliability & coherence, and conformity & correspondence. Therefore, normal model evolution is mainly based on a stabile ground, i.e. a stabile matrix.

One advantage of such stabile grounds is the potential for accumulation and maturation of normal model development and utilisation. It enables knowledge elicitation and acquisition used in design science [72]. It is then part of the rigor cycle.

A simple form of matrix evolution is the combination of scenarios into a coherent set of scenarios. This combination or adduction allows to combine the matrices into holistic ones. The deep models are then typically model suites. A specific form of matrix evolution is consolidation of the matrix, for instance, by development of supporting theories and by maturing methodologies. In this case, the normal models can still be used in the same form.

5.3 Model Notions for Normal Models

We may now elaborate the notions in the introduction. It seems to be obvious how these notions match to our understanding of normal models and their matrices. So, let us consider two additional examples.

An Example for Database IT Practice. [42] considers the mediator-/communication scenario. The model is used for “the representation of some aspects” of the situation model, “enables clearer communication” about the situation model, and “serves as a blueprint to shape and to construct the proposed structures” in the situation model. So, a normal “data model is a device that

- helps the users or stakeholders understand clearly the database system that is being implemented based on the information requirements of an organization, and

- enables the database practitioners to implement the database system exactly conforming to the information requirements.”

This notion of the model is determined by the given two scenario, by the deep model of database models, by the community of business users and database developers, by data engineering and DBMS as its context.

Models for Domain resp. Software Engineering. An application domain is a universe of discourse, an area of human activity or an area of science. Domain engineering is understood as modelling: “a careful description of the domain as it is, void of any reference to possibly desired new software, including requirements to new software”. [10] “By a domain theory we understand a formal model of a domain such that properties of the model the domain can be stated and formally verified - claiming that these properties are properties of the domain being modelled.” “A domain model is thus a description of a sufficient number of domain entities, domain functions, domain events and
domain behaviours - so formulated and detailed that one is able to answer most relevant questions about the domain.”

The deep model is partially explicitly given as a domain theory. The implicit part is, for instance, the notion of an application domain, the focus to description and the functions of the model, the underlying mathematical theory, the modelling language (entities, ...), and the way of associating. All this forms the matrix of the “domain model”.

A similar observation can be made for classical software engineering, e.g. [26, 65].

5.4 Modelling from Art to Science

Modelling is still considered to be an art. It will become a science in future in the understanding of [8]. Moving paths are thus: from practices to principles, from skilled performance to fundamental recurrences, from action to explanation, from invention to discovery, from synthesis to analysis, and from construction to dissection. Modelling as science is organised to understand, exploit and cope with an application. It encompasses natural and artificial aspects of the application. It codifies the body of knowledge mainly on the basis of deep models and matrices. It will have a commitment to normal models for discovery and validation. Models will thus become reproducible. Modelling is enhanced by falsifiability, testing, validation and verification. Modelling as a science has the ability to make reliable predictions, some of them might be surprising. Modelling might also be based on other techniques than presented in this paper. All models in [66] have their matrix. So, modelling based on normal models with their matrix will still be one of the main forms of modelling culture.

References


Data Mining Design and Systematic Modelling
Yannic Kropp
Christian Albrechts University Kiel, Department of Computer Science, D-24098 Kiel, Germany
yk@is.informatik.uni-kiel.de

Bernhard Thalheim
thalheim@is.informatik.uni-kiel.de

Abstract. Data mining is currently a well-established technique and supported by many algorithms. It is dependent on the data on hand, on properties of the algorithms, on the technology developed so far, and on the expectations and limits to be applied. It must be thus matured, predictable, optimisable, evolving, adaptable and well-founded similar to mathematics and SPICE/CMM-based software engineering. Data mining must therefore be systematic if the results have to be fit to its purpose. One basis of this systematic approach is model management and model reasoning. We claim that systematic data mining is nothing else than systematic modelling. The main notion is the notion of the model in a variety of forms, abstraction and associations among models.

Keywords: data mining, modelling, models, framework, deep model, normal model, modelling matrix

1 Introduction

Data mining and analysis is nowadays well-understood from the algorithms side. There are thousands of algorithms that have been proposed. The number of success stories is overwhelming and has caused the big data hype. At the same time, brute-force application of algorithms is still the standard. Nowadays data analysis and data mining algorithms are still taken for granted. They transform data sets and hypotheses into conclusions. For instance, cluster algorithms check on given data sets and for a clustering requirements portfolio whether this portfolio can be supported and provide as a set of clusters in the positive case as an output. The Hopkins index is one of the criteria that allow to judge whether clusters exist within a data set. A systematic approach to data mining has already been proposed in [3, 17]. It is based on mathematics and mathematical statistics and thus able to handle errors, biases and configuration of data mining as well. Our experience in large data mining projects in archaeology, ecology, climate research, medical research etc. has however shown that ad-hoc and brute-force mining is still the main approach. The results are taken for granted and believed despite the modelling, understanding, flow of work and data handling pitfalls. So, the results often become dubious.

Data are the main source for information in data mining and analysis. Their quality properties have been neglected for a long time. At the same time, modern data management allows to handle these problems. In [16] we compare the critical findings or pitfalls of [21] with resolution techniques that can be applied to overcome the crucial pitfalls of data mining in environmental sciences reported there. The algorithms themselves are another source of pitfalls that are typically used for the solution of data mining and analysis tasks. It is neglected that an algorithm also has an application area, application restrictions, data requirements, results at certain granularity and precision. These problems must be systematically tackled if we want to rely on the results of mining and analysis. Otherwise analysis may become misleading, biased, or not possible. Therefore, we explicitly treat properties of mining and analysis. A similar observation can be made for data handling.

Data mining is often considered to be a separate sub-discipline of computer engineering and science. The statistics basis of data mining is well accepted. We typically start with a general (or better generic) model and use for refinement or improvement of the model the data that are on hand and that seem to be appropriate. This technique is known in sciences under several names such as inverse modelling, generic modelling, pattern-based reasoning, (inductive) learning, universal application, and systematic modelling.

Data mining is typically not only based on one model but rather on a model ensemble or model suite. The association among models in a model suite is explicitly specified. These associations provide an explicit form via model suites. Reasoning techniques combine methods from logics (deductive, inductive, abductive, counter-inductive, etc.), from artificial intelligence (hypothetic, qualitative, concept-based, adductive, etc.), computational methods (algorithmics [6], topology, geometry, reduction, etc.), and cognition (problem representation and solving, causal reasoning, etc.).

These choices and handling approaches need a systematic underpinning. Techniques from artificial intelligence, statistics, and engineering are bundled within the CRISP framework (e.g. [3]). They can be enhanced by techniques that have originally been developed for modelling, for design science, business informatics, learning theory, action theory etc.

We combine and generalize the CRISP, heuristics, modelling theory, design science, business informatics, statistics, and learning approaches in this paper. First, we introduce our notion of the model. Next we show how data mining can be designed. We apply this investigation to systematic modelling and later to systematic data mining. It is our goal to develop a holistic and systematic framework for data mining and
analysis. Many issues are left out of the scope of this paper such as a literature review, a formal introduction of the approach, and a detailed discussion of data mining application cases.

2 Models and Modelling

Models are principle instruments in mathematics, data analysis, modern computer engineering (CE), teaching any kind of computer technology, and also modern computer science (CS). They are built, applied, revised and manufactured in many CE&CS sub-disciplines in a large variety of application cases with different purposes and context for different communities of practice. It is now well understood that models are something different from theories. They are often intuitive, visualizable, and ideally capture the essence of an understanding within some community of practice and some context. At the same time, they are limited in scope, context and the applicability.

2.1 The Notion of the Model

There is however a general notion of a model and of a conception of the model: A model is a well-formed, adequate, and dependable instrument that represents origins [9, 29, 30].

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilization scenarios. A well-formed instrument is adequate for a collection of origins if it is analogous to the origins to be represented according to some analogy criterion, it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its purpose.

Well-formedness enables an instrument to be justified by an empirical corroboratation according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins.

The instrument is sufficient by its quality characterization for internal quality, external quality and quality in use or through quality characteristics [28] such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

2.2 Generic and Specific Models

The general notion of a model covers all aspects of adequateness, dependability, well-formedness, scenario, functions and purposes, backgrounds (grounding and basis), and outer directives (context and community of practice). It covers all known so far notions in agriculture, archaeology, arts, biology, chemistry, computer science, economics, electro-technics, environmental sciences, farming, geosciences, historical sciences, languages, mathematics, medicine, ocean sciences, pedagogical science, philosophy, physics, political sciences, sociology, and sports. The models used in these disciplines are instruments used in certain scenarios.

Sciences distinguish between general, particular and specific things. Particular things are specific for general things and general for specific things. The same abstraction may be used for modelling. We may start with a general model. So far, nobody knows how to define general models for most utilization scenarios. Models function as instruments or tools. Typically, instruments come in a variety of forms and fulfill many different functions. Instruments are partially independent or autonomous of the thing they operate on. Models are however special instruments. They are used with a specific intention within a utilization scenario. The quality of a model becomes apparent in the context of this scenario.

It might thus be better to start with generic models. A generic model [4, 26, 31, 32] is a model which broadly satisfies the purpose and broadly functions in the given utilization scenario. It is later tailored to suit the particular purpose and function. It generally represents origins of interest, provides means to establish adequacy and dependability of the model, and establishes focus and scope of the model. Generic models should satisfy at least five properties: (i) they must be accurate; (ii) the quality of generic models allows that they are used consciously; (iii) they should be descriptive, not evaluative; (iv) they should be flexible so that they can be modified from time to time; (v) they can be used as a first “best guess”.

2.3 Model Suites

Most disciplines integrate a variety of models or a society of models, e.g. [7, 14] Models used in CE&CS are mainly at the same level of abstraction. It is already well-known for threescore years that they form a model ensemble (e.g. [10, 23]) or horizontal model suite (e.g. [8, 27]). Developed models vary in their scopes, aspects, and facets they represent and their abstraction.

A model suite consists of a set of models [M1,..., Mn], of an association or collaboration schema among the models, of controllers that maintain consistency or coherence of the model suite, of application schemata for explicit maintenance and evolution of the model suite, and of tracers for the establishment of the coherence.

Multi-modelling [11, 19, 24] became a culture in CE&CS. Maintenance of coherence, co-evolution, and consistency among models has become a bottleneck in development. Moreover, different languages with different capabilities have become an obstacle similar to multi-language retrieval [20] and impedance mismatches. Models are often loosely coupled. Their dependence and relationship is often not explicitly expressed. This problem becomes more complex if models are used for different purposes such as
construction of systems, verification, optimization, explanation, and documentation.

2.4 Stepwise Refinement of Models

Refinement of a model to a particular or special model provides mechanisms for model transformation along the adequacy, the justification and the sufficiency of a model. Refinement is based on specialization for better suitability of a model, on removal of unessential elements, on combination of models to provide a more holistic view, on integration that is based on binding of model components to other components and on enhancement that typically improves a model to become more adequate or dependable.

Control of correctness of refinement [33] for information systems takes into account (A) a focus on the refined structure and refined vocabulary, (B) a focus to information systems structures of interest, (C) abstract information systems computation segments, (D) a description of database segments of interest, and (E) an equivalence relation among those data of interest.

2.5 Deep Models and the Modelling Matrix

Model development is typically based on an explicit and rather quick description of the ‘surface’ or normal model and on the mostly unconditional acceptance of a deep model. The latter one directs the modelling process and the surface or normal model. Modelling itself is often understood as development and design of the normal model. The deep model is taken for granted and accepted for a number of normal models.

The deep model can be understood as the common basis for a number of models. It consists of the grounding for modelling (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities), the outer directives (context and community of practice), and basis (assumptions, general concept space, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense) of modelling. It uses a collection of undisputable elements of the background as grounding and additionally a disputable and adjustable basis which is commonly accepted in the given context by the community of practice. Education on modelling starts, for instance, directly with the deep model. In this case, the deep model has to be accepted and is thus hidden and latent.

A (modelling) matrix is something within or from which something else originates, develops, or takes from. The matrix is assumed to be correct for normal models. It consists of the deep model and the modelling scenarios. The modelling agenda is derived from the modelling scenario and the utilization scenarios. The modelling scenario and the deep model serve as a part of the definitional frame within a model development process. They define also the capacity and potential of a model whenever it is utilized.

Deep models and the modelling matrix also define some frame for adequacy and dependability. This frame is enhanced for specific normal models. It is then used for a statement in which cases a normal model represents the origins under consideration.

2.6 Deep Models and Matrices in Archaeology

Let us consider an application case. The CRC 12661 “Scales of Transformation – Human Environmental Interaction in Prehistoric and Archaic Societies” investigates processes of transformation from 15,000 BCE to 1 BCE, including crisis and collapse, on different scales and dimensions, and as involving different types of groups, societies, and social formations. It is based on the matrix and a deep model as sketched in Figure 1. This matrix determines which normal models can still be considered and which not. The initial model for any normal model accepts this matrix.

Figure 1 Modeling in archaeology with a matrix

We base our consideration on the matrix and the deep model on [19] and the discussions in the CRC. Whether the deep model or the model matrix is appropriate has already been discussed. The final version presented in this paper illustrates our understanding.

2.7 Stereotyping of a Data Mining Process

Typical modeling (and data mining) processes follow some kind of ritual or typical guideline, i.e. they are stereotyped. The stereotype of a modelling process is based on a general modelling situation. Most modelling methodologies are bound to one stereotype and one kind of model within one model utilization scenario.

1 https://www.sfb1266.uni-kiel.de/en
Stereotypes are governing, conditioning, steering and guiding the model development. They determine the model kind, the background and way of modelling activities. They persuade the activities of modelling. They provide a means for considering the economics of modelling. Often, stereotypes use a definitional frame that primes and orients the processes and that considers the community of practice or actors within the model development and utilization processes, the deep model or the matrix with its specific language and model basis, and the agenda for model development. It might be enhanced by initial models which are derived from generic models in accordance to the matrix.

The model utilization scenario determines the function that a model might have and therefore also the goals and purposes of a model.

2.8 The Agenda

The agenda is something like a guideline for modeling activities and for model associations within a model suite. It improves the quality of model outcomes by spending some effort to decide what and how much reasoning to do as opposed to what activities to do. It balances resources between the data-level actions and the reasoning actions. E.g. [17] uses an agent approach with preparation agents, exploration agents, descriptive agents, and predictive agents. The agenda for a model suite uses thus decisions points that require agenda control according to performance and resource considerations. This understanding supports introspective monitoring about performance for the data mining process, coordinated control of the entire mining process, and coordinated refinement of the models. Such kind of control is already necessary due to the problem space, the limitations of resources, and the amount of uncertainty in knowledge, concepts, data, and the environment.

3 Data Mining Design

3.1 Conceptualization of Data Mining and Analysis

The data mining and analysis task must be enhanced by an explicit treatment of the languages used for concepts and hypotheses, and by an explicit description of knowledge that can be used. The algorithmic solution of the task is based on knowledge on algorithms that are used and on data that are available and that are required for the application of the algorithms. Typically, analysis algorithms are iterative and can run forever. We are interested only in convergent ones and thus need termination criteria. Therefore, conceptualization of the data mining and analysis task consists of a detailed description of six main parameters (e.g. for inductive learning [34]):

(a) The data analysis algorithm: Algorithm development is the main activity in data mining research. Each of these algorithms transfers data and some specific parameters of the algorithm to a result.
(b) The concept space: the concept space defines the concepts under consideration for analysis based on certain language and common understanding.
(c) The data space: The data space typically consists of a multi-layered data set of different granularity. Data sets may be enhanced by metadata that characterize the data sets and associate the data sets to other data sets.
(d) The hypotheses space: An algorithm is supposed to map evidence on the concepts to be supported or rejected into hypotheses about it.
(e) The prior knowledge space: Specifying the hypothesis space already provides some prior knowledge. In particular, the analysis task starts with the assumption that the target concept is representable in a certain way.
(f) The acceptability and success criteria: Criteria for successful analysis allow to derive termination criteria for the data analysis.

Each instantiation and refinement of the six parameters leads to specific data mining tasks. The result of data mining and data analysis is described within the knowledge space. The data mining and analysis task may thus be considered to be a transformation of data sets, concept sets and hypothesis sets into chunks of knowledge through the application of algorithms.

Problem solving and modelling considers, however, typically six aspects [16]:

(1) Application, problems, and users: The domain consists of a model of the application, a specification of problems under consideration, of tasks that are issued, and of profiles of users.
(2) Context: The context of a problem is anything what could support the problem solution, e.g. the sciences’ background, theories, knowledge, foundations, and concepts to be used for problem specification, problem background, and solutions.
(3) Technology: Technology is the enabler and defines the methodology. It provides [23] means for the flow of problem solving steps, the flow of activities, the distribution, the collaboration, and the exchange.
(4) Techniques and methods: Techniques and methods can be given as algorithms. Specific algorithms are data improvers and cleaners, data aggregators, data integrators, controllers, checkers, acceptance determiners, and termination algorithms.
(5) Data: Data have their own structuring, their quality and their life span. They are typically enhanced by metadata. Data management is a central element of most problem solving processes.
(6) Solutions: The solutions to problem solving can be formally given, illustrated by visual means, and presented by models. Models are typically only normal models. The deep model and the matrix is already provided by the context and accepted by the community of practice in dependence of the needs of this community for the given application scenario. Therefore, models may be the final result of a data mining and analysis process beside other means.

Comparing these six spaces with the six parameters we discover that only four spaces are considered so far in data mining. We miss the user and
application space as well as the representation space. Figure 2 shows the difference.

![Figure 2 Parameters of Data Mining and the Problem Solving Aspects](image)

### 3.2 Meta-models of Data Mining

An abstraction layer approach separates the application domain, the model domain and the data domain [17]. This separation is illustrated in Figure 3.

![Figure 3 The V meta-model of Data Mining Design](image)

The data mining design framework uses the inverse modeling approach. It starts with the consideration of the application domain and develops models as mediators between the data and the application domain worlds. In the sequel we are going to combine the three approaches of this section. The meta-model corresponds to other meta-models such as inductive modelling or hypothetical reasoning (hypotheses development, experimenting and testing, analysis of results, interim conclusions, reappraisal against real world).

### 4 Data Mining: A Systematic Model-Based Approach

Our approach presented so far allows to revise and to reformulate the model-oriented data mining process on the basis of well-defined engineering [15, 25] or alternatively on systematic mathematical problem solving [22]. Figure 4 displays this revision. We realize that the first two phases are typically implicitly assumed and not considered. We concentrate on the non-iterative form. Iterative processes can be handled in a similar form.

#### 4.1 Setting the Deep Model and the Matrix

The problem to be tackled must be clearly stated in dependence on the utilization scenario, the tasks to be solved, the community of practice involved, and the given context. The result of this step is the deep model and its matrix. The first one is based on the background, the specific context parameter such as infrastructure and environment, and candidates for deep models.

![Figure 4 The Phases in Data Mining Design (Non-iterative form)](image)

The data mining tasks can be now formulated based on the matrix and the deep model. We set up the context, the environment, the general goal of the problem and also criteria for adequateness and dependability of the solution, e.g. invariance properties for problem description and for the task setting and its mathematical formulation and solution faithfulness properties for later application of the solution in the given environment. What is exactly the problem, the expected benefit? What should a solution look like? What is known about the application?

Deep models already use a background consisting of an undisputable grounding and a selectable basis. The explicit statement of the background provides an understanding of the postulates, paradigms, assumptions, conceptions, practices, etc. Without the background, the results of the analysis cannot be properly understood. Models have their profile, i.e. goals, purposes and functions. These must be explicitly given. The parameters of a generic model can be either order or slave parameters [12], either primary or secondary or tertiary (also called genotypes or phenotypes or observables) [1, 5], and either ruling (or order) or driven parameters [12]. Data mining can be enhanced by knowledge management techniques.

Additionally, the concept space into which the data mining task is embedded must be specified. This concept space is enhanced during data analysis.

#### 4.2 Stereotyping the Process

The general flow of data mining activities is typically implicitly assumed on the basis of stereotypes which form a set of tasks, e.g. tasks of prove in whatever system, transformation tasks, description tasks, and investigation tasks. Proofs can follow the classical...
Data mining and analysis is rather stereotyped. For instance, mathematical culture has already developed a good number of stereotypes for problem formulation. It is based on a mathematical language for the formulation of analysis tasks, on selection and instantiation of the best fitting variable space and the space of opportunities provided by mathematicians.

Data mining uses **generic models** which are the basis of normal models. Models are based on a separation of concern according the problem setting: dependence-indicating, dependence-describing, separation or partition spaces, pattern kinds, reasoning kinds, etc. This separation of concern governs the classical data mining algorithmic classes: association analysis, cluster analysis, data grouping with or without classification, classifiers and rules, dependences among parameters and data subsets, predictor analysis, synergetics, blind or informed or heuristic investigation of the search space, and pattern learning.

### 4.3 Initialization of the Normal Data Models

Data mining algorithms have their capacity and potential [2]. Potential and capacity can be based on SWOT (strengths, weaknesses, opportunities, and threats), SCOPE (situation, core competencies, obstacles, prospects, expectation), and SMART (how simple, meaningful, adequate, realistic, and trackable) analysis of methods and algorithms. Each of the algorithm classes has its strengths and weaknesses, its satisfaction of the tasks and the purpose, and its limits of applicability. Algorithm selection also includes an explicit specification of the order of application of these algorithms and of mapping parameters that are derived by means of one algorithm to those that are an input for the others, i.e. an explicit association within the model suite. Additionally, evaluation algorithms for the success criteria are selected. Algorithms have their own obstinacy, their hypotheses and assumptions that must be taken into consideration. Whether an algorithm can be considered depends on acceptance criteria derived in the previous two steps.

So, we ask: **What kind of model suite architecture suits the problem best? What are applicable development approaches for modelling? What is the best modelling technique to get the right model suite? What kind of reasoning is supported? What not? What are the limitations? Which pitfalls should be avoided?**

The result of the entire data mining process heavily depends on the appropriateness of the data sets, their properties and quality, and more generally the data schemata with essentially three components: application data schema with detailed description of data types, metadata schema [18], and generated and auxiliary data schemata. The first component is well-investigated in data mining and data management monographs. The second and third components inherit research results from database management, from data mart or warehouses, and layering of data. An essential element is the explicit specification of the quality of data. It allows to derive algorithms for data improvement and to derive limitations for applicability of algorithms. Auxiliary data support performance of the algorithms.

Therefore typical data-oriented questions are: **What data do we have available? Is the data relevant to the problem? Is it valid? Does it reflect our expectations? Is the data quality, quantity, recency sufficient? Which data we should concentrate on? How is the data transformed for modelling? How may we increase the quality of data?**

### 4.4 The Data Mining Process Itself

The data mining process can be understood as a coherent and stepwise refinement of the given model suite. The model refinement may use an explicit transformation or an extract-transform-load process among models within the model suite. Evaluation and termination algorithms are an essential element of any data mining algorithm. They can be based on quality criteria for the finalized models in the model suite, e.g. generality, error-proneness, stability, selection-proneness, validation, understandability, repeatability, usability, usefulness, and novelty.

Typical questions to answer within this process are: **How good is the model suite in terms of the task setting? What have we really learned about the application domain? What is the real adequacy and dependability of the models in the model suite? How these models can be deployed best? How do we know that the models in the model suite are still valid? Which data are supporting which model in the model suite? Which kind of errors of data is inherited by which part of which model?**

The final result of the data mining process is then a combination of the deep model and the normal model whereas the first one is a latent or hidden component in most cases. If we want, however, to reason on the results then the deep model must be understood as well. Otherwise, the results may become surprising and may not be convincing.

### 4.5 Controllers and Selectors

Algorithmics [6] treats algorithms as general solution pattern that have parameters for their instantiation, handling mechanisms for their specialization to a given environment, and enhancers for context injection. So, an algorithm can be derived based on explicit selectors and control rules [4] if we neglect context injection. We can use this approach for data mining design (DMD). For instance, an algorithm pattern such as regression can use this approach for data mining design (DMD).
on enablers that specify applicability of the approach, on error rules, on data evaluation rules that detect dependencies among control parameters and derive data quality measures, and on quality rules for confidence statements.

4.7 Data Mining and Design Science

Let us finally associate our approach with design science research [13]. Design science considers systematic modelling as an embodiment of three closely related cycles of activities. The relevance cycle initiates design science research with an application context that not only provides the requirements for the research as inputs but also defines acceptance criteria for the ultimate evaluation of the research results. The central design cycle iterates between the core activities of building and evaluating the design artifacts and processes of the research. The orthogonal rigor cycle provides past knowledge to the research project to ensure its innovation. It is contingent on the researchers’ thoroughly research and references the knowledge base in order to guarantee that the designs produced are research contributions and not routine designs based upon the application of well-known processes.

The relevance cycle is concerned with the problem specification and setting and the matrix and agenda derivation. The design cycle is related to all other phases of our framework. The rigor cycle is enhanced by our framework and provides thus a systematic modelling approach.

5 Conclusion

The literature on data mining is fairly rich. Mining tools have already gained the maturity for supporting any kind of data analysis if the data mining problem is well understood, the intentions for models are properly understood, and if the problem is professionally set up. Data mining aims at development of model suites that allows to derive and to draw dependable and thus justifiable conclusions on the given data set. Data mining is a process that can be based on a framework for systematic modelling that is driven by a deep model and a matrix. Textbooks on data mining typically explore in detail algorithms as blind search. Data mining is a specific form of modeling. Therefore, we can combine modeling with data mining in a more sophisticated form. Models have however an inner structure with parts which are given by the application, by the context, by the commonsense and by a community of practice. These fixed parts are then enhanced by normal models. A typical normal model is the result of a data mining process.

The current state of the art in data mining is mainly technology and algorithm driven. The problem selection is made on intuition and experience. So, the matrix and the deep model are latent and hidden. The problem specification is not explicit. Therefore, this paper aims at the entire data mining process and highlights a way to leave the ad-hoc, blind and somehow chaotic data analysis. The approach we are developing integrates the theory of models, the theory of problem solving, design science, and knowledge and content management. We realized that data mining can be systematized. The framework for data mining design exemplarily presented is an example in Figure 4.

References


Acknowledgement. We thank for the support of this paper by the CRC 1266. We are very thankful for the fruitful discussions with the members of the CRC.
The Rigor Cycle of Conceptual Modelling

Ajantha Dahanayake¹ and Bernhard Thalheim²

¹ School of Business and Management, Lappeenranta University of Technology, Finland
Ajantha.Dahanayake@lut.fi
² Department of Computer Science, Christian Albrechts University Kiel, D-24098 Kiel
thalheim@is.informatik.uni-kiel.de

Abstract. Modelling is still conducted as the work of an artisan and workmanship. While a general notion of the model and of the conceptual model has already been developed, the modelling process is not investigated so well. It is currently based on modelling methodologies. Modelling has to be based on principles and a general theory of modelling activities. The modelling activities need however a conceptualisation. We combine approaches developed in design science, ontology engineering, decision processes, and conceptual modelling for development of general stages, phases and steps of modelling. The main elements of our approach discussed in this paper are the way how a modelling decision is made and which phases and steps are commonly observed during modelling.

Keywords: Conceptual modelling, modelling actions, modelling decisions, phases and steps of modelling

1 Introduction

Design science research and conceptual modelling research have attracted a lot of research. Design science “is the scientific study and creation of artefacts as they are developed and used by people with the goal for solving practical problems of general interest.” [20] “Conceptual modeling is about describing” (syntax, “semantics” (, and pragmatics) “of software applications at a high level of abstraction. Specifically, conceptual modelers (1) describe structure models in terms of entities, relationships, and constraints; (2) describe behavior or functional models in terms of states, transitions among states, and actions performed in states and transitions; and (3) describe interactions and user interfaces in terms of messages sent and received, information exchanged, and look-and-feel navigation and appearance.” [9].

Comparing these general statements, we observe a good overlap whenever information systems are the target of development. The design of an IT artefact includes explication of the problem, definition of requirements, development of the artefact, demonstration of the artefact, and evaluation of the artefact. A similar flow of activities can be distinguished for modelling. Both approaches to
development of artificial artefacts [41] may be based on methodological frameworks, on general paradigms and principles, and on general ethical, economical, ecological etc. principles.

Therefore, it is beneficial to integrate the two approaches. In general, design and modelling are partially different and - at the same time - largely similar activities\(^1\). It seems that the two development approaches share many issues and can benefit from each other.

The controversy [1, 21, 25, 32] discuss the differences between design science and information systems research. It seems that the two research directions are completely different and do not have too much in common. Design science has its background in industrial and interior design and in psychology. Conceptual modelling started with database modelling and is more directly influenced by computer engineering.

Conceptual modelling is a specific form of modelling. Models become conceptualised due to incorporation of concepts - or more generally, conceptions - into the model. These concepts are commonly shared within a community of practice that is involved in the modelling process. Models are a universal vehicle or better instrument in almost all sciences and engineering. They can be understood as the ‘third’ dimension of science [5, 52].

Therefore, we can compare design science research for information systems development with conceptual modelling of information systems. Design science distinguishes the relevance cycle, the modelling cycle, and finally the rigor cycle. In this paper we look more specifically into the rigor cycle and use it for development principles that can be incorporated into for conceptual modelling. Since neither the rigor cycle nor the principles of conceptual modelling have led to an accepted theory, we start our research with one specific aspect of systematic modelling: support for design decisions.

1.1 Models in Design Science

Models, modelling languages, modelling frameworks and their background have dominated conceptual modelling research and information systems engineering for the last four decades. Design science research considers artefacts. It is understood as an object or thing made by humans with the intention that it will be used to address a practical problem. Artefacts are, for instance, physical objects, drawings or blueprints. Models are also artefacts whenever they are not virtual. Artefacts are used in development scenarios. Their functions are what they can do for members in their community of practice, what role they can play for them, and how they can support them in their activities.

Conceptual models are mediators between the application world and the implementation or system world [6]. Design science distinguishes the relevance

\(^1\) To design means, for instance, (1) to create, fashion, execute, or construct according to a plan, (2) to conceive and plan out in the mind driven by a purpose and devised for a specific function or end, and (3) to develop an artefact.

To model means, for instance, (1) to plan or form after a pattern, (2) to shape or fashion, and (3) to construct a model guided by an origin. [38].
cycle as the iterative process that re-inspects the application and the model, the
design cycle as the iterative model development process, and the rigor cycle that
aims in grounding and adding concepts developed to the knowledge base [14].
Research in design science and on conceptual modelling has resulted in a large
body of knowledge, practices, and techniques. Modelling is based on modelling
activities. Each modelling step considers specific work products, orients towards
specific aspects of the system or application, involves different partners, and uses
a variety of resources [49] used for system development in computer engineering.
The separation into the application world, the modelling and model world and
the knowledge or design science world [13, 48] supported an assessment of the
results of modelling and an evaluation of the results of research on modelling.

Conceptual modelling has been oriented in the past mainly to clarification
on languages, on methods for deployment of such languages, on (mathematical)
theories as foundations of syntactic, semantics and pragmatics of model, and
on evaluation and quality guaranteeing methods [16, 31, 34, 48]. The application
world is used as a starting point for the development of systems that solve some
problems of the application domain under consideration. By analyzing these
two directions we come to a conclusion similar to [60]. In reality design science
research and research on conceptual modelling are two research issues that may
benefit from each other. The two communities are already engaged in a discussion
of the added value of each side [3, 4, 24, 26, 33, 35, 36, 42, 58, 57, 59].

1.2 The Three Perspectives of Conceptual Modelling

Based on the notions in the Encyclopedia Britannica [38], we distinguish between
the conception of a model, the conception of a model activity, and the conception
of systematic, reflected and well-organised modelling.

The model as an artifact: A model is a well-formed, adequate, and depend-
able instrument that represents origins. [2, 8, 50, 51]

To model as an activity: ‘To model’ is a scientific or engineering activity be-
side theoretical or experimental investigation. The activity is an additive
process. Corrections are possible during this activity. Modelled work may
be used for construction of systems, for exploration of a system, for defini-
tion and negotiation, for communication, for understanding and for problem
solving.

Modelling as a systematically performed, reflected, technological pro-
cess: Modelling is a technique for systematically using knowledge from com-
puter science and engineering to introduce technological innovations into the
planning and development stages of a system.

1.3 Modelling as an Activity

Modelling includes two different kinds of activities:

Model deployment is based on activities such as
adaption, concept enrichment, optimisation, specialisation, instantiation, refinement, grinding,
applicability studies (evaluation, assurance, composition for application),
iintegration, selection, renovation, modernisation, (r)evolution, migration,
problem solution, classification, practice, understanding, theory or paradigm (r)evolution, and
explanation.

Model development is typically based on another set of activities such as
- abstraction of origin, scoping, validation, verification, testing, optimisation,
- construction, composition, definition, integration, classification, invention,
- enrichment, adaption, mutation, recombination, refinement, reuse, preparation for deployment, and
- understanding, theory or paradigm injection.

1.4 Objectives and the Storyline of the Paper

In this paper, we discuss modelling foundational principles and theoretical underpinnings for purpose-oriented models and modelling. Our approach is based on the three cycles of design science research activities of research artifact creation. We thus combine conceptual modelling, design science approaches, decision processes, ontology engineering, and the theory of information system models. We do not intent to review all relevant literature in the rich body of knowledge developed in design science research or conceptual modelling research. There are conference series such as DESRIST, ER, and Models etc. and journals such as DKE, EJIS, and MISQ etc. Instead, we follow the approach [58] and use design science research for conceptual modelling of information systems.

Section 2 provides an account of design science, its position on modelling, and the stages of design. Section 3 describes the modelling decisions and its parallels to systematic decision support and the modelling act leading to models and solution imperfection. Section 4 gives an account of systematic conceptual modelling, exploration, and model amalgamation leading to formal model foundation. Section 5 summarises the conclusions of this research.

2 Design Science and Modelling

Design science originated in the area of IT development. It concentrates on novel artifacts in the form of models, methods, and systems that support people while developing, using, maintaining, reconsidering, and migrating IT solutions. It considers four perspectives [20]: (1) people, practices and problems; (2) artifacts as solutions to problems in IT practices; (3) the context and anatomy of artifacts; and (4) the study of artifacts.
2.1 The Relevance Cycle in Design Science

Design science research requires the creation of an innovative, purposeful artifact for a special problem domain. The artifact must be evaluated in order to ensure its utility for the specified problem. The relevance cycle initiates design science research with an application context that not only provides the requirements for the research as inputs but also defines acceptance criteria for the ultimate evaluation of the research results. The rigor cycle provides past knowledge to the research project to ensure its innovation. It is contingent on the researchers to thoroughly research and reference the knowledge base in order to guarantee that the designs produced are research contributions and not routine designs based upon the application of well-known processes. The central design cycle iterates between the core activities of building and evaluating the design artifacts and processes of the research.

2.2 The Modelling or Design Cycle

Modelling is a crucial activity in the creation of the design, the artifact. The models and modelling itself implies an ethical change from describing and explaining of the existing world to shaping it. One can question the values of this type of models and modelling oriented design research, i.e. whose values and what values dominate it, emphasizing that research may openly or latently serve the interests of particular dominant groups. The interests served may be those of the host organization as perceived by its top management, those of IS users, those of IS professionals or potentially those of other stakeholder groups in society. Therefore, in order to define the acceptance criteria for ultimate evaluation of the research, modelling and models need to be mapped to a theoretical foundation.

2.3 The Rigor Cycle

The rigor cycle is considered as the conceptualisation and generalisation or knowledge development cycle [56]. The rigor cycle also aims at the development of knowledge about the application domain and the model. This part of the rigor cycle is conceptualisation. The second target of the rigor cycle is the derivation of abstract knowledge and experience, of scientific theories that can be applied in similar application cases, of (pragmatical) experience for modelling, and of meta-artifact or reference models based on model-driven development (MDD) approaches. Design science aims at another kind of model refinement by adding more rigor after evaluation of a model. This refinement is essentially model evolution and model evaluation. Another refinement is the enhancement of models by concepts. This refinement is essentially a ‘semantication’ or conceptualisation of the model.

We observe that the rigor cycle is orthogonal to the modelling and relevance cycles. The modelling cycle may be broken into a description stage that relates the application domain to the model and a prescription stage that uses the
model for system construction. The rigor cycle which is somehow orthogonal has at least two facets: one facet that is important for the model and one facet that is important for generalisation of the model, e.g., for derivation of patterns or reference models and for extraction of model and modelling knowledge beyond the actual modelling activity. In this paper, we concentrate on the rigor cycle of conceptualization and the knowledge development for modelling foundational principles and theoretical underpinning to validate the purposeful values of models and modelling within the design science research activities.

2.4 Stages Of “To Model”

Based on foundations of conceptual modelling [46], ontology engineering [45], and design science for information systems development (e.g. [29]) and summarising, we distinguish three stages of modelling activities:

Stage I: Model development is based on four phases: description, formulation, ramification, and validation. In the description phase, individual perception and situation models involved into the modelling situation, are isolated and the corresponding primary properties are identified and represented. We realise in the next sections that this phase includes exploration and model amalgamation. In the formulation phase, properties are interrelated, integrated and combined into a preliminary, initial model. This model is analysed in a ramification phase in order to check whether the model is a proper solution and to interpret and to consider its implications. Finally, the model and its capability and capacity are assessed in a validation phase.

Stage II: Model deployment considers the developed model within the given application situation, assesses this model in other application contexts in order to evaluate its stability and plasticity, and derives its added value.

Stage III: The rigor cycle also investigates the experience we have gained during developing the given model. Conceptual modelling uses this experience as a hidden intuitive basis for further development. We may however use this experience within a paradigmatic synthesis for recapitulation and consolidation of conceptualisation concept gathering, ontologisation, grounding and tagging, i.e. for knowledge acquisition.

3 Modelling Decisions

The main question is now how, when, why, on what, in which way and why design decisions are made besides the organisation of the design process itself, its flow of activities, and the involvement of actors into the design process.

3.1 Systematic Decision Support

According to [22], modelling and modelling decisions enhancement (DE) activities are encouraged within the studio concept. A DE studio has five main
components:
- Studio style: Learning, Enquiry and Participative.
- Decision process coordinators: these include facilitators, domain experts, and suite.
- Scripting: the balance between improvisation and formalized methods.
- Suites and development and support expertise.
- Location and rooms: the options here range from fixed point to distributed Web conferencing and from simple technology infrastructure to multimedia heaven.

[22] consider the mix of skills required to fulfill the demands of a studio for modelling decisions listed as landscaping, facilitation, recipes, suites and process as a means of a complete package of developing an architecture - a solution. Let us combine this approach with the technology proposal for change management in [19].

- **Landscaping** is the domain of expertise of the business strategist and domain expert. In terms of both understanding the decision issues and decision-makers, information resources, processes and the basics of what to model, why and how. In addition, the landscaper has to have some credibility, whether as an insider or outside adviser, with senior managers and stakeholders. Otherwise, the studio is just an exercise or a “pilot”, “prototype” or “lab”, all of which are euphemisms for “don’t take this too seriously.”
- **Facilitation**: Behavioral knowledge and process skills are a key for the process of arriving at a solution.
- **Recipes** apply wherever possible proven recipes that include effective scripts. Recipes are proven, repeatable and transferable, specify ingredients and sequencing, permit variations and innovations, and result in something people eat and are likely to come back for another meal. Building recipes requires research and writing and the willingness to place “secrets” and “methodology” in the public domain. It demands teaching as well: developing a body of knowledge and building a critical mass of skilled practitioners. Since technology moves so fast, each new generation of software draws on a new generation of developer and there is little passing on of experience and knowledge.
- **Suites** ensure that tools are designed and implemented within an overall distributed architecture. The goal of suite development is to make the “system” as transparent, easy to access, reliable as the electrical system, where any breakdown is a news item and crisis.
- **Processes** make commitment to a decision of the explicit target and agenda.

The blockage here is organizational culture, management style, stakeholder relationships and legacy of existing decision processes.

### 3.2 The Modelling Action and Design Decisions

The modelling action is similar to the speech act and consists of

1. a selection and construction of an appropriate model depending on the task and purpose and depending on the properties we are targeting and the con-
text of the intended system and thus of the language appropriate for the system,
2. a workmanship on the model for detection of additional information about the original and of improved model,
3. an analogy conclusion or other derivations on the model and its relationship to the real world, and
4. a preparation of the model for its use in systems, to future evolution and to change.

Therefore, the DE studio approach provides a specific tactics to modelling.

### 3.3 Modelling Knowledge and Decisions Imperfection

In the case of conceptual modelling, the rigor cycle can be based on knowledge obtained within the five consecutive phases [11]:

1. exploration,
2. model amalgamation and adduction,
3. model formulation,
4. model deployment, and
5. paradigmatic synthesis.

Modelling decisions have to be based on transparent and realistic objectives [11, 52, 54, 55]. The correspondences of elements of the conceptual model to particular pattern in the real world or the perception models must be based on conformity criteria. Modelling can be considered as progressive cognition within the context, for the purpose of development and within the concept space. Models cannot be developed in its full scientific rigor and are thus objects of evolution. Modelling actions balance between exploratory decisions (description, explanation, prediction) and inventive aspects (reification, refinement). Modelling kits are supporting the quality of modelling decisions. Modelling actions suffer from the breadth-depth paradox. We want to have as much detail as necessary and want to be as broad as sufficient. Modelling decisions must be continuously evaluated within a modelling process by either mode or all three modes of assessment (coherence, correspondence, commensurability). They are conditionally anchored to the experience, knowledge and intuition gained so far. Modelling also includes negotiation within the community of practice and with the stakeholders of the information system.

Adequateness of models is based on analogy, focusing, and purposefulness of the model. Focusing provides a means for explicit modelling of the divergence from the real world with incompleteness, open issues and potential errors [15]. Therefore, a model is imperfect [17] due to exceptional states that are not considered, incompleteness due to limitations of the modelling language and the scope of modelling, and due to errors, which are either based on real errors or exceptional states or on biases by the community of practice.
4 Systematic Development of Conceptual Models

Let us now consider the modelling phases and steps and highlight the decisions that must be made during modelling. We concentrate in this paper on the first two model development phases: description and formulation. Ramification and validation extend the approach in [55]. The two next stages (model deployment and paradigmatic synthesis) are deferred to a forthcoming paper due to space limitations. The section is based on our entire experience on conceptual modelling and on the experiences of several decades of database realisation. The body of knowledge developed so far and used in real practice is very large. It needs however a systematisation, categorisation and generalisation. There are very few publications (e.g. [10, 12, 30, 40, 39, 43]) that provide such systematisation of the experience gained so far. The generalisation and the categorisation is however an open research field so far.

Modelling of structures is a systematically performed technological process. It is a technique for applying knowledge from other branches of engineering and disciplines of science in effective combination to solve a multifaceted engineering problem. In addition to structure development, it is important to define databases systems themselves. The systems are first of all man-made.

Due to involvement of the second author into the development and the service for the CASE workbenchs (DB) and ID we have collected a large number of real life applications. Some of them have been really large or very large, i.e., consisting of more than 1,000 attribute, entity and relationship types. The largest schema in our database schema library contains of more than 19,000 entity and relationship types and more than 60,000 attribute types that need to be considered as different. Another large database schema is the SAP R/3 schema. It has been analysed in 1999 by a SAP group headed by the second author during his sabbatical at SAP. At that time, the R/3 database used more than 16,500 relation types, more than 35,000 views and more than 150,000 functions. The number of attributes has been estimated by 40,000. Meanwhile, more than 21,000 relation types are used. The schema has a large number of redundant types which redundancy is only partially maintained. The SAP R/3 is a very typical example of a partially documented system. Many of the design decisions are now forgotten. The high type redundancy is mainly caused by the incomplete knowledge on the schema that has been developed in different departments of SAP over several decades.
resistant. Modelling and especially information system modelling\textsuperscript{3,4} is a creation and production process, an explanation and exploration process, an optimisation and variation process, and a verification process. This distinction allows to relate the specific purpose with macro-steps of modelling and with criteria for approval or refusal of modelling results. Modelling is thus at the same time problem solving and engineering.

4.1 The Model Description Phase

In this paper we concentrate our investigation of the model description phase on two (macro-)steps: model exploration and model amalgamation.

Main Phases of Model Description for Starting from Scratch. The exploration step is based on state-of-affairs and the functions a model should play during information system development. The state-of-affairs is typically represented by a reality model that already abstracts from the state-of-affairs and perception models that are used in the community of modellers and business users. We may distinguish in this step a number of activities: the situation and the perception models are disassembled. Later, monstration may be applied to some situation model. This situation model is negotiated within the community of practice and users and represented by a nominal model.

\textsuperscript{3} We develop our approach here on the approach that is established and widely practised and taught in almost all textbooks. We leave out the more sophisticated approach in [53]. At the business user level, user viewpoints can be represented by user viewpoint schemata or more generally views. At the conceptual level, these viewpoints are going to be harmonised and mapped to a conceptual schema. It is assumed that the user viewpoints are then sub-schemata of the conceptual schema. This conceptual schema is mapped to a logical and later to a physical schema. The viewpoints are cut down to logical views which typically consist of single-table definitions on the basis of a query to the logical schema. A user viewpoint is then called external view. The query might be more complex and thus not be based on a sub-schema of the conceptual schema. The database structure architecture consists of the logical schema, external views defined on top of the logical schema and an implementation or physical schema. With the introduction of the conceptual model, the architecture description has been changed by considering the logical and the physical as an implementation schema and using the conceptual schema as the mediator between views and the implementation schemata. It creates a mismatch since the views are defined on top of the implementation schema. [18] breaks with this three-layer architecture by proposing the conceptual view tower mechanism where business user viewpoints are represented by conceptual views. [53] rounds off this approach by considering the conceptual model to consist of a conceptual schema and a collection of conceptual views.

\textsuperscript{4} We concentrate the investigation of the modelling process to ‘greenfield’ modelling called modelling from scratch and to model gardening called evolutionary modelling. We do not investigate ‘brownfield’ modelling and modernisation for already operating database systems based on modelling and redevelopment for legacy systems based on macro-modelling methods and especially migration strategies [23].
The next step is model amalgamation. The result is a real model. Amalgamation is oriented on the justification of the model and on the quality criteria for the model. It integrates also criteria for well-formedness of models.

Modelling typically also results in modelling experience that can be elicited during or after model development. This modelling experience elicitation and acquisition is part of paradigmatic synthesis and therefore of the rigor cycle.

**Evolutionary Model-Based and Background-Aware Modelling.** Modelling is often not performed from scratch. Rather we start with an explication of experience that uses stereotypical or generic models. We may also start with existing models for an already existing information system. We thus elaborate artifacts of interest, e.g. reference or existing models. We explicitly extract the background of these models. Next we explicate their purpose, their background, their context and compare the result with the reality models and the objectives of development. The result is again a situation model.

This situation model is now assessed and evaluated. It is typically reformulated by specialisation and refinement. We thus explicitly describe why the model is adequate and dependable. This model can also be enhanced by formal methods.

In a similar form, the experience gained is incorporated into the body of modelling knowledge.

### 4.2 The Exploration (Macro-)Step

Exploration start with a well-defined modelling task, a well-defined scope, and consider choices. It is often assumed to be based on deductive approaches. It seems to better however to consider inductive and abductive approaches first.

It is based on the following three steps:

**Disassemble:** The perception and situation models are converted into constituent parts in dependence on the specific assumptions, specific reality and state-of-the-art properties, and specific foci and scopes. Methods are dissolution, segmentation, analysis of coherent units, refinement and categorisation, and examination.

**Monstration/manifestation** is the act of demonstrating, exhibiting, and demonstration. It often considers familiar situations and examples. We consider typical application situations, typical phenomena, typical system states, concepts and conceptions. During monstration we are more interested in specific kinds of models, Galilean models oriented on improvement of the state-of-the-art.

Monstration may follow the W*H specification framework [7]. Typical questions answered are: What is the demonstrated situation about? What systems and phenomena are involved into the situation? What is the state of every system? What concepts are necessary to describe and/or explain? What is the reference system? How can these concepts be represented?

Manifestation and reflection consider model properties, model variants and model capabilities. Based on these considerations we may derive obligations for model revision.
Nominal models in the exploration step use parameters and variables that can be instantiated during further model development and progressive model refinement. The glossary, namespace, agreements on conceptions, assumptions on the model background, decisions on model structures, and composition pattern are often imported from the application domain.

The result of this step is now a nominal (or perception) model that is a generalisation of a subsidiary models. This model explicitly represents the background via its grounding (paradigms, culture, background, foundations, theories, postulates, restrictions, authorities, conventions, commonsense) and its basis (concepts, language, routine, training, infrastructure, assumptions, though community, thought style, pattern, methodology, guidelines, practices).

4.3 The Model Amalgamation (Macro-)Step

Amalgamation aims at combination or unification of model elements into one form [47]. It includes merger, consolidation, and mixing or blending of different elements. We typically focus on one real model during conceptual data modelling of information systems. We might also use several models but leave it out of scope within this paper.

Amalgamation is mainly based on inductive reasoning. It might incorporate abductive and adduction reasoning based on the association to the situation and perception models. It can be enhanced by methods of plausible reasoning. Classical development strategies (top-down, bottom-up, modular, inside-out, mixed) provide means for combination and unification of elements. In this case we can use local top-down and bottom-up development operations for ER models [46].

Model composition can follow a number of strategies and tactics, especially for unification. Since we are interested in adequacy and dependability of models we explicitly propagate these properties during amalgamation. Typical ER-based composition principles are: global-as-design, unification of viewpoints, explicit consideration of realisability, empiric evaluation by sample data, homomorphic mappings from the situation and perception models, and consideration of specific elements of the ER modelling language. This step is often governed by practical guidelines and rational constraints (general guiding principles, acceptable tolerance (approximation limits, precision intervals, data preciseness), convenient modes for logging and handling of data, appropriate mathematical or formal representations and operations, norms and corroborations for the real model and criteria for evolution (refinement, modernisation, modification, replacement): level of detail, type system and mapping style to type system, handling of exceptions and deviations (NULL, default), treatment of hierarchies, controlled redundancy, ground type system, quantity matrix (Mengengerïst), constraint enforcement, treatment of cardinality, inherent constraints, naming conventions, abbreviation rules, kind of semantics (set or pointer), weak types, translation and tolerance for complex attributes, handling of identification). We may also require that structures used correspond to natural situations (good design is functional, useful,
aesthetic, innovative, good business, honest, long-lasting, minimalistic, understandable, user-oriented, unobtrusive, simple as possible, thorough down to the last detail, and focused).

Finally, maturity of models based on SPICE or CMM has to be provided [55]. The real model should be fully defined, must be well-understood, provide all semantics in an explicit form, and use explicit concepts. We might use different definitional frames but in a coherent form. General modelling principles are modularisation, abstraction, and explicit coupling. It is a good approach to use best practices within the modelling framework. They should allow to preserve also design principles that are given for the realisation environment.

4.4 The Model Formulation Phase

The model formulation phase aims at formulation of the well-formed, adequate and dependable model. This model may use several representation forms, e.g. conceptual data models combine diagrammatic and formal representations. It may also contain several sub-models that represent viewpoints of business users [53]. Finally, the assessment of the model is explicitly given.

The formulation is based on decisions such as depiction of the elements of previous models with an explicit consideration of the model function and purpose. We develop criteria for adequacy and dependability of the model and start to explicitly represent the model grounding and background. Since the model should also represent various viewpoints of business users we have to enhance it by explicit view schemata and aggregations as well as abstractions. Reasoning on justification might be based on an argument calculus [52] or argument logics [27].

The model must be completed. Typical drivers for completion are application domain requirements, the background behind the situation and perception models, the specifics of the modelling language, and the generic model behind the model. We need to adjust the scope of modelling elements. The goal is also to develop a well-formed model that fits well to the situation and perception models.

The model is also assessed by an elementary deployment and tested against the real world. This assessment is often backed by some test data based on an experimentation strategy. It might also be tested by elementary utilisation. The first main result of assessment is a justification of the model by an explanatory statement, by confirmation of rational coherence, by a validation of the model against the state-of-affairs, and by explicit consideration of stability of the model against non-essential deviations of the state-of-affairs. The second main result of assessment is an explicit statement on model quality based on quality characteristics for quality in use, external quality and internal quality. The assessment allows to reason on the model capacity and potential.
5 Conclusion

5.1 Combining Conceptual Modelling and Design Science

This paper shows how design science and conceptual modelling can benefit from each other. More specifically we discuss how conceptual modelling can benefit from design science research. Let us use and enhance Figure 1 from [49]. Modelling of IT artifacts typically starts with an understanding of the state of affairs, with objectives and consideration of requirements. This perception may be described and directly used for the development of new IT artifacts without any model alike agile approaches. The modelling cycle results however in a model...
that can be used for IT development either directly or in a more reflected way. The rigor cycle in design science and modelling as a systematically performed, reflected and well organised process is based on an understanding of all actions undertaken. This process can also be used for development of new knowledge and its integration into the existing body of knowledge.

Therefore, we observe that the rigor cycle and systematics of modelling may each other enhance and complement.

5.2 Contributions of Design Science to Conceptual Modelling

Since design science research and conceptual modelling are tackling the same problem - proper development of (information) systems - we discussed in this paper how design science research can be used for an underpinning of modelling activities. The decision steps we presented are the basis for a general stepwise procedure of systematic design.

The formalisation of this approach is delayed to a forthcoming paper. Formalisation also includes approaches to a general theory of modelling such as [11, 28, 37, 44]. The main issue was so far the development of the combined approach.

References


Conceptual Modeling: Enhancement through Semiotics

Veda C. Storey\textsuperscript{1} and Bernhard Thalheim\textsuperscript{2}

\textsuperscript{1}Computer Information Systems, J. Mack Robinson College of Business
Georgia State University, Atlanta, GA, USA
E-mail: vstorey@gsu.edu

\textsuperscript{2}Department of Computer Information Systems, Kiel University, Germany
E-mail: thalheim@is.informatik.uni-kiel.de

Abstract. Conceptual modeling uses languages to represent the real world. Semiotics, as a general theory of signs and symbols, deals with the study of languages and is comprised of syntax, semantics, and pragmatics. Pragmatics includes the explicit representation of the intentions of users. A common assumption is that all levels of database design (user, conceptual, logical, and physical) can be modeled using the same language. However, languages at the conceptual level are often enhanced by concepts that attempt to capture inherent pragmatics. This research proposes that concepts from semiotics can provide the background needed to understand an application. Specifically, pragmatics and semantics are considered at both the user and conceptual level, based on proposed constraints.

Keywords: Conceptual modeling, languages, semiotics, semantics, constraints

1 Introduction

Conceptual models act as mediators between the application and an implementation [11]. Conceptual modelers often attempt to model situations that occur in the real world using one language as a construction mechanism, and a model for a schema. Representing how the world operates must be described at the right level of specification. This tends to be done, for example, using an entity-relationship diagram as a modeling tool. However, it is difficult to expect one language to be able to handle all phases of modeling. Semantic issues need to be captured and modeled during both the design phases. The objective of this research, therefore, is to understand how to create better conceptual models by considering these different levels of abstraction and how they might be addressed. Although language is usually the main vehicle for modeling, additional understanding is needed for collaboration among stakeholders. Semiotics, as a general theory of signs and symbols, deals with the study of languages, and could serve as the needed background. The contributions are to: propose that models should be defined from the perspective of semiotics, and propose an additional set of constraints.
2   Modeling Challenges in Conceptual Modeling

Levels of abstraction. Many modeling languages are applied at different levels of abstraction. Business issues might be applied at the application level. Prescription issues for implementation are at a detailed level of specification. Although different, they are often all represented by an entity-relationship diagram.

Semantics. Semantics (meaning of terms) is challenging [5]. Constraints are often used as a surrogate for business rules [6]. Attempting to capture and represent semantics in terms of first-order predicate logic seems restrictive. Implicit or lexical semantics contribute to complete semantics.

Inclusion constraints. These could be class-based; for example, a student is a person. The person identification is reused for student as a co-existence constraint, expressible via identification (becoming a foreign key constraint in the relational model). Then an enforcement mechanism can be: 1) canonically declared based on reference existence and reference enforcement; or 2) expressed by the on-event-if-condition-then-action (ECA) paradigm. The enforcement can be refined for control, application, optimization, and exception handling. If the inclusion constraint is not class-based, but value-based, then support and enforcement become more challenging. For example, the Student type may use an attribute Name, which corresponds to a person’s Name in a type Person.

Cardinality constraints. These have two main approaches to define their semantics: look-up and participation. Look-up works well for binary associations without relationship attributes. Participation constraints mix two different kinds of semantics with rigidity for extreme cases, despite the need to represent normal cases. ‘Min/Max’ captures the absolute extreme for all potential cases. The ‘min’ captures a (generalized) inclusion constraint; ‘max’ is intended to capture a (generalized) multiplicity constraint. For a relationship where the minimum participation could be ‘0’ (someone is a student but not taking courses yet), a null value would be allowed in an implementation. However, a “normal” interpretation of the relationship is that a student must be registered for at least one course (null not allowed). Cardinality constraints impact other constraints in the schema [3].

Implicit constraints. Constraints can be implicit or hidden due to syntax construction. The eER modeling language uses relationship types with inherent (construction) inclusion and existence constraints as based-on constraints. Relationship objects reference their component objects; for example, entity objects. Therefore, the relationship objects can only exist if the corresponding entity object exists, making the semantics implicit, based upon the way in which relationships are constructed and used. They become explicit in the corresponding SQL specification.

Type semantics. eER modeling uses a Salami-slice strategy, oriented on the homogeneity of types and thus on decomposition into small, meaningful semantic units. Things in the application domain are multifaceted. A human is represented via a Person type that is separated from the Student type, which is associated via an IsA relationship (or subclass), to the Person type. At the same time, Student can be associated with other...
types, such as: student_engagement, student_facilities, dormitory, etc. Depending upon the view, a student might best be considered using the notion of a student or the notion of the more general object, person. Research has analyzed classification challenges [4].

**Implicit representation of viewpoints.** At the application level, it might be beneficial to consider user viewpoints that are represented as views [11]. For instance, a student might best be considered, including more general objects, e.g. person.

**Separation of syntax and semantics.** The separation of syntax and semantics is generally problematic. Most modelers learn a language using simple problems. However, real world problems are complex, so one language, or modeling technique, is not appropriate for all. It is impossible to represent a business problem at an application level of abstraction and implementation issues based on a singleton diagram. The problem is understanding and representing semantics.

**Restricted and mixed semantics.** Instead of general constraint frames, specific cases are often considered; e.g., mapping ratios (1:N, N:M, 1:1) to capture some binary relationship semantics. Sometimes, N:M ratios declare the maximum to be higher than 1. Look-up and participation cardinalities may be used with the same syntactic notion.

### 3 Models, Expressions, and Stakeholder Levels

**Models and Conceptual Models.** The notion of a model is complex and not necessarily well understood; similarly, for the process of modeling. Consider four perspectives: 1) the origins to be considered by the model; 2) the profile of the model (e.g. its function, purpose, or goal); 3) the stakeholders or the community of practice that the model must satisfy; and 4) the context within which the model and the origins are considered. The first two perspectives are internal; the second two, external.

A model is guided on its background [10]: the grounding of the model (paradigms, postulates, theories, culture, and conventions); and the basis for the model (e.g. languages used, concepts and conceptions, community, and commonly accepted practices). The basis of a model may change on demand. The perceptions of users might need to be represented in a model. Multiple coherent perceptions, a description of a system, or an augmented system might also be useful. A model can have many different purposes: to describe or explain a situation; specify and represent a concept someone has in mind; to aid in communication among stakeholders; or to decompose complex situations. A model is a well-formed, adequate and dependable artifact, commonly accepted by its community of practice within a given context [10], [11].

**Semiotics of Signs: Icons, Symbols and Indexes.** Semiotics, the study of the theory of signs, emphasizes the properties of things in their capacity. It is reasonable to apply semiotics to aid in this understanding since, before using a modeling language, it is first necessary to understand the language and its inherent bias.

**Syntax** refers to the arrangement of words in sentences and phrases. Syntax should be simple, parsimonious, and harmonic.
Semantics is concerned with the meaning of sentences and defines the interpretation of a sentence in the real world, depending on its context. It refers to the meaning of signs and what they represent in the real world.

Pragmatics considers the relationship between parts of sentences or signs and their users within a situation and context. It is user-dependent.

Although language is the main vehicle for modeling, semiotics is the background needed for understanding so that collaboration among stakeholders can result. Syntax, semantics, and pragmatics may follow different paradigms, leading to some effective use. The strictness of first-order predicate logic might be inappropriate during modeling. It is, however, needed in the final result. For example, natural utterances use the connective “and/or” with the meaning of logical OR. Similar observations can be made for all connectives, especially, for quantifiers.

Syntax has been well investigated for formal languages. Semantics can be defined in a variety of ways; e.g. for evaluation of variables, incorporation of context, scope of states, exceptions, and matching between syntactic language and semantic structure [8]. Problems arise when pragmatics is taken into consideration because the pragmatic interpretation depends on the community of practice, its culture, scope and attention.

Syntax, semantics and pragmatics of models are all important issues, and depend upon the needs of a model and its context.

Abstraction Levels of Stakeholders. At the application level, the perceptions of the users must be considered and combined with the context. At the conceptual modeling level, the resulting conceptual model must be based on what was developed at the application level. The logical level is typically based on an understanding of the platform, with the best practice being to use models that are mappings or compilations of the conceptual model.

4 Illustrative Example

A conceptual database model consists of a conceptual schema and a number of view schemata [11]. The view schemata are the result of transformations [1] [9] that map the viewpoints of the application level to sub-schemata of the conceptual schemata.

Consider a student-dormitory-course schema in Figure 1. Suppose a student is enrolled in several programs at a university. The dormitory association is dependent upon the program that a student takes. Specifically, a student lives in a dormitory that corresponds to the program (business, music, etc.) in which the student is enrolled. A student might obtain some financial support from a program, depending upon the level of completion of the program. A student makes courses that are required for a given program. The credit hours assigned to a course, may vary across courses, depending upon whether the course is intended for one program, or whether it is a mandatory or elective course. Any course can only be counted one time towards one program. A student is required to take a minimum number of classes per term. If a student fails a course, then the student may retake the course, up to a maximum of three times. A
course has an associated tuition fee that must be within the limits of a given term, which may vary from term to term.

There are, however, some aspects of this situation that are difficult to model.
- A student can only take a course a maximum of three times. This might be overcome by adding a separate entity, called class or section, and a relationship: Course has Classes, with min/max cardinalities of (0,3) from student to class.
- A course can have different credit hours depending upon the program.
- A student can have multiple majors, which requires a decision about the dormitory to which a student should be assigned.
- The normal case for enrolled in does not capture freshmen who are not enrolled.
- The student must take courses that are required by the program.

These problems are at the application level. Someone must represent the university situation correctly and implement the corresponding results into a database. Also, involved is the end-user, a student. The database designer must attempt to models these in one conceptual model.

Fig. 1. Entity-Relationship Model of Student-Dormitory Application

5 Semiotics Reconsidered

Semantics and Pragmatics at the Application Level. Models at the application level have their own origins that they represent, profile, context, and community. The origins are consolidated perception models, enhanced by situation models that are commonly accepted in the application domain. Each community has a community-specific model; that is, a “local-as-design” approach. Objects under consideration are not homogeneous, for example, a department is considered together with its department head. Or, a student view incorporates all of the classes a student takes and refers to a university program class view from the university administration. A student is typically enrolled in one and only one program. There might be other students. Generalization and specialization follow natural semantics.
Models at this level of abstraction can be used at the conceptual level for communication and negotiation within and between communities of practice. Semantics and pragmatics differ based on the perception and understanding within the communities. Models may not be complete. Semantics may not be rigid. Objects are often considered to be holistic; for example, students together with their courses based on their programs. Therefore, we are not bound to normal data type construction. Constraints typically consider normal cases instead of extreme ones. Class planning might not require that students take classes, but student planning is based on the minimum and maximum credit hours a student must acquire in a given term.

Models at the application level have their own coherence. The underlying model allows us to integrate the different models. Models at the user level are typically not denotative but connotative, and follow cultural or community interpretations. For this reason, ontologies are appropriate for specifying domain-specific content [2].

**Model Semantics at the Conceptual Level.** A conceptual data model reflects, integrates and harmonizes the user views. Types specify homogeneous classes and are decomposed accordingly. The functionality definition is based on an entity-relationship algebra and given only after the structure model is complete. Constraints refine the structure; that is, semantics are defined only after the syntax is complete. The entity-relationship schema uses a diagram that is assumed to be complete, and represents its component at the same level of granularity and precision. Pragmatics tend to be hidden in a conceptual model, even though it is, in essence, an underlying model. It is assumed to be defined though external views.

**Constraints at the Application Level and Conceptual Level.** Constraints are generally considered valid for all of an application. However, a user’s community might consider the ‘normal’ case or abstract (generalize) from exceptions, or omit them. Users use different scope, context, origins, and purposes. E.g., cardinality constraints represent some aspect, within specific semantics and pragmatics.

**The Nature of Constraints.** At the conceptual level, pragmatics must be handled by syntax and semantics. Cardinality constraints can do so, but are rigid and based on participation or lookup definition [7]. In the participation approach, extreme cases are included, in an attempt to represent exceptional cases. For example, an (1,N) constraint states that a corresponding relationship must exist for all entity classes. One solution is to use a harmonization of all user models and integrate them into the conceptual model. In this “global-as-design” approach, user views represent the external views of users, resulting in the challenge of properly representing finer semantics and pragmatics of these views. Due to the “local-as-view” design, constraints are introduced from the user’s point of view. A conceptual model should harmonize all of these views to provide a holistic view of all constraints. A similar harmonization can occur at the logical level.

In Figure 1, a freshman could be enrolled in a program or not. If the freshman is enrolled, then a dormitory can be assigned based on the program enrolled. Later the freshman might also take courses. Then, a student is either a normal student, a student who does not take courses, or a student who does not have yet a dormitory. At the logical level, we can use tables for each of these specific cases and define a view that combines them. At the logical level, horizontal decomposition can be applied [10].
relation type can be decomposed by selection expressions E1, …, En into separate types, provided this decomposition forms a partition on the class for this type. Therefore, we might also use a conceptual type, made up of conceptual base types. The base type has semantics without any context, but all subclasses are identified.

**Objectives for Developing Better Constraints.** Semantics can vary, depending on the user. This results in problems when mapping to a conceptual model, so the conceptual model should be more flexible. In most practices, normalization deals with the exceptional case where semantics causes a change of structure and the schema. That is, semantics drives syntax, in contrast to “semantics follows syntax.” DBMS provide a much finer means for integrity maintenance. Maintenance can be deferred (eager or lazy integrity enforcement). Consistency can be supported at the row level. Integrity constraints can be maintained at the application level. Integrity can be made through views. Finally, flexible strategies may be used, besides the no-action and rollback approach; for example, on the basis of triggers or stored procedures.

These observations show that conceptual integrity constraints can be more elaborated if we can map the constraints to DBMS features. Here, we simply aim to show how semantics and syntax can be developed in a holistic approach. We further assume that pragmatics is defined at the application level, based on views, leading to the following observations and requirements.

1. DBMS technology must provide a better way of treating syntax and semantics at the conceptual level, which captures pragmatics at the user level.
2. A holistic view is needed for integrated usage of syntax together with semantics.
3. Flexibility is required for changes needed to accommodate new technology.
4. A mapping procedure for advanced integrity constraints should be supported.

**Proposed Extensions of Integrity Constraints by Context as Part of Semantics.**

1. *Actions* on a database are insert, delete and update for: a single object, one class, or objects tightly bundled via class inclusion constraints. Actions might be defined as an *action pattern*. This extends single-object actions to a complex object action while disabling the basic actions whenever a complex pattern exists.
2. The *scope pattern* is a view-defining query. This query defines either a single type view or, in general, the view schema on the conceptual schema.
3. *Enforcement style pattern* is for constraints that are timed as eager (default) or lazy (with(out) delay) enforcement, after an action (default), or as control before an action, with a level statement (e.g. DBMS, transaction, and interface levels).
4. *Reaction pattern* is for immediate enforcement or exception handling with a timed exit sub-pattern or timed enforcement, based on an enforcement obligation.

The above illustrates the need to deal with structure versus semantics. They can be formally defined and implemented. Then, in contrast to traditional approaches in which “semantics follows syntax,” syntax and semantics may be treated as a whole.

**Holistic View.** A *conditional integrity constraint* is a pair of a context and a constraint. Constraints can be combined to partition a problem based on a scope pattern. For example, cardinality constraints Card(R,R') = (1,1) are for R = enrolled_in, and R' = Student with a selection predicate for: freshmen, a student who does not yet have an assigned dormitory, and students who did not yet take courses. The cardinality
constraint is only valid for “normal” students. Adding an attribute `term` to the type `takes` could ensure that a student has not taken a course more than three times.

For example, for freshman with a dormitory, we may use a relaxed enforcement style. For freshman without a dormitory, we might use an interface style. That is, an insertion of such a student is only possible by an encapsulated insertion of the student, the programs, and the dormitory with a temporary insertion into the corresponding basic types; and a transfer of the object to another basic class whenever additional data are inserted. However, problems that exist or can be deduced for these constraints are not usually considered. All user needs cannot be represented by semiotics. View integration is difficult with global constraints, and usually completed based on user views. From a semiotics perspective, the user view should be considered as much as possible.

6 Conclusion

Many problems arise from the need to carry out modeling at multiple levels, depending upon the stakeholders. Since semiotics deals with language, it is proposed as an underlying basis from which to understand and capture semantics at different levels of abstraction. Additional conditional constraints are needed to model context, namely, action, scope, enforcement style and reaction.

Acknowledgements. This research was supported by the J. Mack Robinson College of Business, Georgia State University. Thanks to Melinda McDaniel for her assistance.

References

A model is a well-formed, adequate, and dependable instrument that represents origins. Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilisation scenarios. As an instrument or more specifically an artifact a model comes with its background, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form. The background is often implicit and hidden.

A well-formed instrument is adequate for a collection of origins if it is analogous to the origins to be represented according to some analogy criterion, it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its purpose. Well-formedness enables an instrument to be justified by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins. The instrument is sufficient by its quality characterisation for internal quality, external quality and quality in use or through quality characteristics such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions). A well-formed instrument is called dependable if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

Content of This Third Collection

   See also [DT18,KTD+].

27. B. Thalheim. Conceptual Model Notions -
   A Matter of Controversy; Conceptual Modelling and its Lacunas. [Tha18a] .......................... 14
   See also [KT17a,JHWD+16].

28. B. Thalheim. Qualitative and quantitative models [Tha18d] ................................................. 27
   See also [KT18a,Tha18c,Tha17a,AGK+19].

   See also [JTH+18,JT19a,ST19,JHMT19] and our papers on culture.

   See also [KT18a,Tha18c,Tha17a,Tha19b,Tha19d,KTD+].

   Digital playground for policy decision making. [HFHT18] ................................................. 74
   See also [KT18b,NKT,KT19].

   See also [Tha19d,TJ20,KT20].
33. H. Jaakkola and B. Thalheim. Models as programs: The envisioned and principal key to true fifth generation programming. [JT19b] 100
See also [TSF19,MT19,KT19].

34. B. Thalheim. Conceptual models and their foundations. [Tha19b] 120
See also [Tha19c,KTD+].

35. A. Molnar and B. Thalheim. Usage models mapped to programs. [MT19] 134
See also [TSF19,KT19,KT20].

See also [JT19b,MT19,KT19].

See also [NT15,ST10,ST19,MT19a,MT19c,MT19e,MT17a,MT19b].

38. B. Thalheim. Models for communication, understanding, search, and analysis. [Tha19d] 171
See also [MT19a,MT19,KT19].

See also [KT18a,KT19,KT20,KT19b,KT19].

See also [AGK+,Tha19b].

Remark: See our previous work on the entity-relationship approach to modelling, e.g. [ET11,MT00,MT09b,ST15], on semantics [ST13,MT11b], on programming, on the theory of databases and information systems and on the technology of information systems at http://dblp.uni-trier.de/pers/hd/t/Thalheim:Bernhard or https://www.researchgate.net/profile/Bernhard_Thalheim/publications or https://scholar.google.com/citations?user=lK3yb9gAAAAJ or http://independent.academia.edu/BernhardThalheim or ...
**Content of The Second Collection**

available through research gate and academia

**Table of Content of the Second Collection**

*Remark: page numbers from the second collection in the sequel.*

<table>
<thead>
<tr>
<th>No.</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Compare with [DT11, TN15, Tha11a, Tha12b].</td>
<td></td>
</tr>
<tr>
<td></td>
<td>See also previous work on model suites, model composition [MST09a, ST10, Tha10a].</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Compare with the BPMN foundation work in the BPMN collection, especially with [BST09, KST09, TT13], [TTFZ14, TFTL + 14].</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Compare with the foundational book [Tha00, Tha09c, Tha09a, Tha09b] and my previous work on entity-relationship approach.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Compare with our research on semantics and logical foundation.</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>B. Thalheim and A. Dahanayake. A conceptual model for services [TD15]</td>
<td>75</td>
</tr>
<tr>
<td></td>
<td>See also [DT13a, DT10b, MSTW09a, DT12, ADT12a, ADT12b, DT15, TD16], [MSTQ09, MSTW09b, MSTW11], [MSTW12].</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>B. Thalheim and M. Tropmann-Frick. Wherefore models are used and accepted? The model functions as a quality instrument in utilisation scenarios [TTF16c, TJ20]</td>
<td>86</td>
</tr>
<tr>
<td></td>
<td>This paper continues research quality [JT10], architecture [JT11, NT14, TELT14], and privacy [AFT09].</td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>B. Thalheim. Model-based engineering for database system development [Tha17b]</td>
<td>99</td>
</tr>
<tr>
<td></td>
<td>See our approaches to development of methodologies for modelling, especially database modelling and the codesign framework in the 90ies and 00s.</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>B. Thalheim. General and specific model notions [Tha17a]</td>
<td>117</td>
</tr>
<tr>
<td>22</td>
<td>B. Thalheim. Normal models and their modelling matrix [Tha18c]</td>
<td>131</td>
</tr>
<tr>
<td>23</td>
<td>Y. Kropp and B. Thalheim. Data mining design and systematic modelling [KT17b]</td>
<td>155</td>
</tr>
<tr>
<td></td>
<td>See the previous work on data mining design [BT12, JP10, ST12].</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>A. Dahanayake and B. Thalheim. The rigor cycle of conceptual modelling [DT18]</td>
<td>163</td>
</tr>
<tr>
<td></td>
<td>See also our contributions to design science foundations of conceptual modelling [DT10b, DT10a, DT11, DT13b].</td>
<td></td>
</tr>
</tbody>
</table>
# The First Collection

available through research gate and academia

## Table of Content of the First Collection

*Remark: page numbers from the first collection in the sequel.*

   Preliminary version: Lecture Notes in Computer Science 5833, [Tha09d] [.89ex]


4. The Science and Art of Conceptual Modelling. TLDKS VI, LNCS 7600, [Tha12c] ............................................. 94-122
   Preliminary version: Lecture Notes in Computer Science 6860, [Tha11c]


6. The Definition of the (Conceptual) Model. EJC’13, [Tha13a, Tha14] .......................................................... 135-148

7. Das Modell des Modelles. EWE’15 [Tha15] ........................................................................................................ 149-151

   The main book on the Kiel approach to models: [TN15].


10. A Conceptual Model for Services. CMS’15@ER’15 [TD15] ................................................................. 164-174


References


Model Adequacy

Bernhard Thalheim
Christian-Albrechts University at Kiel, Department of Computer Science, D-24098 Kiel

December 28, 2017

Abstract

Models, modeling languages, modeling frameworks and their background have dominated research on information systems engineering for last four decades. Models are mainly used as mediators between the application world and the implementation or system world. Modelling is still conducted as the work of an artisan and workmanship. While a general notion of the model and of the conceptual model has already been developed, the modelling process is not investigated so well.

Modelling has to be based on principles and a general theory of modelling activities. One of the lacunas is still a proper understanding of adequacy of models, adequacy of modelling and deployment methods, and a theory of adequacy. We will concentrate on the first issue.

Keywords: model notion; model adequacy; analogy; focus/truncation/abstraction; purposeful; well-formed model; model dependability

1 Models, Modelling Activities, Systematic Modelling

Models are principle and central instruments in mathematics, data analysis, modern computer engineering (CE), in teaching any kind of computer technology, and also modern computer science (CS). They are built, applied, revised and manufactured in many CE&CS sub-disciplines in a large variety of application cases with different purposes and context for different communities of practice. CE&CS expressively use the conception of model for daily work. Modelling is one of their four central paradigms beside structures (in the small and large), evolution or transformation (in the small and large), and collaboration (based on communication, cooperation, and coordination). It is now well understood that models are something different from theories. They are often intuitive, visualisable, and ideally capture the essence of an understanding within some community of practice and some context. At the same time, they are limited in scope, context and the applicability. Models have been considered to be somewhere in the middle between the perception and understanding of the state of affairs (world, situations, data etc.) and theories (concepts and conceptions, statements, beliefs, etc.) since they may describe certain aspects of a situation and may represent parts of a theory. Models should thus be considered to be the third dimension of science [2, 50, 52]. Other disciplines (see for instance [50]) have developed a different understanding of the notion of model, of the function of models in scientific research and of the purpose of the model. Models are often considered to be artifacts where also virtual models are considered beside real one. Models might also be mental models and thought concepts. Models are used as instruments in utilisation scenarios. They function in these scenarios.

2 The Notion of the Model

There is however a general notion of a model and of a conception of the model:

A model is a well-formed, adequate, and dependable instrument that represents origins. (see [8, 45, 47])

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilisation scenarios.

The model should be well-formed according to some well-formedness criterion. As an instrument or more specifically an artifact a model comes with its background, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form. The background is often implicit and hidden.

1The title of the book [4] has inspired this observation.
A well-formed instrument is **adequate** for a collection of origins if it is **analogous** to the origins to be represented according to some analogy criterion, it is more **focused** (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its **purpose**.

Well-formedness enables an instrument to be **justified** by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins.

The instrument is **sufficient** by its **quality** characterisation for internal quality, external quality and quality in use or through quality characteristics (see [40]) such as correctness, generality, usefulness, comprehensibility, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

A well-formed instrument is called **dependable** if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

### 3 Adequacy as a Generalisation of Mapping, Truncation, and Pragmatic Properties

Following H. Stachowiak (see, for instance, [33, 34]), a model is often defined in a phenomenalistic way based on three properties:

1. **Mapping** property: the model has an origin and can be based on a mapping from the origin to the instrument.
2. **Truncation** (reduction) property: the model lacks some of the ascriptions made to the origin.
3. **Pragmatic** property: the model use is only justified for particular model users, the tools of investigation, and the period of time.

We observe however that these properties do not qualify a representation as a model. The mapping and truncation properties are far too strict and need further investigation. A model must not be a mapping from some origin. Homomorphism is a nice property but far too strict in most applications. We might use representations that are not images of mappings such as a Turing machine, a system architecture, or development strategies. Furthermore, we might use representations that are not reducts of origins such as (conceptual) information system models for the variety of viewpoints users of databases might have. Truncation (or abstraction) considers a model to be an Aristotelian one by abstraction by disregarding the irrelevant. The relevance criterion is based on the purpose (or goal or function) of a model. So, truncation is far too fuzzy. Models are developed by a community of practice for utilisation by a community of practice and in a context. The utilisation depends on the intentions of users and their context. So, we observe that the utilisation of models determines (a) the kind of model, (b) the governing purposes or goals of utilisation of the model, (c) the properties of a model, (d) the amplification a model provides with extensions, (e) the idealisation by scoping the model to the ideal state of affairs, (f) the divergence by deliberately diverging from reality in order to simplify salient properties of interest, and (g) the added value of a model. The seven additional statements are combined in the **mission** a model has. The mission clarifies how the model functions well within its intended scenarios of usage according to its capacity and potential. The mission must be coherent with the context, the determination or specific basis of conduct or utilisation of the model, and must be acceptable for the users or – more concrete – the community of practice. Therefore, the mission clarifies the functions (and anti-functions or forbidden ones), purposes and goals of the utilisation, the potential and the capacity of the model.

### 4 An Agenda: Towards Adequacy of Modelling Methods

The theory of modelling is still struggling with a number of research challenges (see [40]): Adjustable selection of principles depending on modelling goals; model suites with explicit model association; development of a language culture; models 2.0; explicit treatment of model value; coexistence of theory, languages, and tools; adequate representation variants of models; compiler development for models; model families and variants. These challenges are the background behind the consternation that has been summarised at Modellierung 208 by W. Hesse (see also [11, 12]): ... but they do not know what they do ...; Babylonian language confusion and muddle; “it’s not a bug, it’s a feature” and other statements for de-facto-standards and lobbyists; why I should cope with what was the state of art yesterday; each day a new wheel, new buzzwords without any sense, and a new trend; without consideration of the value of the model; competition is a feature, inhomogeneity; Laokoon forever; dreams about a sound mathematical foundation; take but don’t think - take it only without critics; academia in the ivory tower without executable models; where is the Ariadne thread through.

This consternation and the challenges can be summarised by a research agenda, e.g. with the following problems:
Can be develop a simple notion of adequateness that still covers the approaches we are used in our subdiscipline?

Do we need this broad coverage for models? Or is there any specific treatment of dependability for subdisciplines or specific deployment scenarios?

Which modelling methods are purposeful within which setting?

Which model deployment methods are properly supporting the function of a model within a utilisation scenario?

How does the given notion of model match with other understandings and approaches to modelling in computer science and engineering?

What is the background of modelling, especially the basis that can be changed depending on the function that a model plays in some utilisation scenario?

Language matters, enables, restricts and biases (see [54]). What is the role of languages in modelling?

Which modelling context results in which modelling approach?

What is the difference between the modelling process that is performed in daily practice and systematic and well-founded modelling?

Are we really modelling reality or are we only modelling our perception and our agreement about reality?

What is the influence of the modeller’s community and schools of thought?

5 The Storyline for this Keynote

In this keynote we discuss mainly the first element of the research agenda: adequateness of models, modelling methods, and modelling as a systematic activity. So far, the adequateness notion is far too fuzzy and too wide. The keynote is based on a large body of knowledge developed on models, modelling activities, and systematic modelling \(^2\) The basis of our understanding of adequacy and dependability is the case study in the Kiel compendium of models, modelling activities and systematic modelling (see [50]). This MMM approach to modelling has been investigated for models in agriculture, archaeology, arts, biology, chemistry, computer science, economics, electrotechnics, environmental sciences, farming, geosciences, historical sciences, languages, mathematics, medicine, ocean sciences, pedagogical science, philosophy, physics, political sciences, sociology, and sports.

The introduction is based on a discussion of adequacy for two modelling methods widely used in our area. The specific utility of models follow the line given in [19, 20]. We are going to introduce a general and formal notion of adequacy. Since adequacy cannot be separated from dependability we have also to investigate it for the two modelling methods. Finally, the keynote ends with a collection of open problems on adequacy of modelling methods.

References


\(^2\) For details and classical database design books we refer to [1, 5, 17, 21, 22, 26, 31, 37, 38].

For details on language theory we refer to [3, 7, 18, 27, 28, 43, 56].

For details of design science research we refer to [13, 15, 30, 55].

Formalisation also includes approaches to a general theory of modelling such as [9, 10, 16, 23, 24, 25, 29, 32, 35, 57].

For details of our work we refer to [2, 6, 8, 14, 39, 41, 42, 43, 44, 45, 46, 48, 49, 51, 52, 53].


Abstract

The conception of a conceptual model is differently defined in Computer Science and Engineering as well as in other sciences. There is no common notion of this conception yet. The same is valid for the understanding of the notion of model. One notion is: A model is a well-formed, adequate, and dependable instrument that represents origins and functions in some utilisation scenario. The conceptual model of an information system consists of a conceptual schema and of a collection of conceptual views that are associated (in most cases tightly by a mapping facility) to the conceptual schema. In a nutshell, a conceptual model is an enhancement of a model by concepts from a concept(ion) space.

The variety of notions for conceptual model is rather broad. We analyse some of the notions, systematise these notions, and discuss essential ingredients of conceptual models. This discussion allows to derive a research program in our area.

Keywords: Model, Conceptual model, Concept and notion of a model, Art of modelling.

1 What is a Conceptual Model

Modelling is a topic that has already been in the center of research in computer engineering and computer science since its beginnings. It is an old subdiscipline of most natural sciences with a history of more than 2,500 years. It is often restricted to Mathematics and mathematical models what is however to much limiting the focus and the scope. Meanwhile it became a branch in the Philosophy of Science. The number of papers devoted to modelling doubles each year since the early 2000’s.

It is often claimed that there cannot be a common notion of model that can be used in sciences, engineering, and daily life. The following notion covers all known so far notions in agriculture, archaeology, arts, biology, chemistry, computer science, economics, electrotechnics, environmental sciences, farming, geosciences, historical sciences, languages, mathematics, medicine, ocean sciences, pedagogical science, philosophy, physics, political sciences, sociology, and sports. The models used in these disciplines are instruments that are deployed in certain scenarios (see [39]). A commonly acceptable statement for a general model notion is the following one:

A model is a well-formed, adequate, and dependable instrument that represents origins and functions in some utilisation scenario. Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilisation scenarios. The function determines the purposes and goals.

CS-conceptual modelling2 is often related back to the introduction of the entity-relationship model(ling language) for information systems development. It surprises nowadays that there is no commonly accepted notion of conceptual model yet. There have been several trials but none of them was sufficient and was able to cover the idea of the conceptual model.

The database and information systems research communities are extensively using the term “conceptual model”3. The notion of conceptual model still needs some clarification: what is a conceptual model and what not; which application scenario use which kind of conceptual model; is conceptual modelling only database modelling; do we need to have an understanding of modelling; is a conceptual database model only a reflection of a logical database model; is a conceptual model a model or not; etc. Let us illustrate the wide spread and understanding of conceptual models, the activity of conceptual modelling, and the modelling as a scientific and engineering process by some examples4,5:

Reality and world description: Conceptual modelling is the activity of formally describing some aspects of

---

1We refer to the model-to-model-modelling compendium (see [39]) for notions that are not introduced in this paper.

2In the paper we restrict ourselves to this kind of conceptual model and thus omit the CS acronym. In general, a conceptual model is a representation of a system in its widest sense on the basis of concept(i)on(s) that allow people to consciously act and being guided in certain situations of their systems.

3Faceted search for the term “conceptual model” in DBLP results in more than 5,000 hits for titles in papers (normal DBLP search also above 3,400 titles).

4The notion of conceptualisation, conceptual models, and concepts are far older than considered in computer science. The earliest contribution to models and conceptualisations we are aware of is pre-socratic philosophy.

5Wikiquote (see [44]) lists almost 40 notions. We add our list to this list.
the physical and social world around us for purposes of understanding and communication. Such descriptions, often referred to as conceptual schemata, require the adoption of a formal notation, a conceptual model in our terminology.

Community description: Conceptual modeling is about describing the semantics of software applications at a high level of abstraction.

Specifically, conceptual modelers (1) describe structure models in terms of entities, relationships, and constraints; (2) describe behavior or functional models in terms of states, transitions among states, and actions performed in states and transitions; and (3) describe interactions and user interfaces in terms of messages sent and received and information exchanged. In their typical usage, conceptual-model diagrams are high-level abstractions that enable clients and analysts to understand one another, enable analysts to communicate successfully with application programmers, and in some cases automatically generate (parts of) the software application.

Conceptual database modeling: A data model is a collection of concepts that can be used to describe a set of data and operations to manipulate the data. When a data model describes a set of concepts from a given reality, we call it a conceptual model.

Instance-integrating conceptual modeling: A conceptual model consists of a conceptual schema and an information base. A conceptual schema provides a language for reasoning about an object system, and it specifies rules for the structure and the behaviour of the system. A description of a particular state is given in an information base, which is a set of type and attribute statements expressed in the language of the conceptual schema.

System-representation models: A conceptual model is a descriptive model of a system based on qualitative assumptions about its elements, their interrelationships, and system boundaries.

Représentational models: A conceptual model is a type of diagram which shows a set of relationships between factors that are believed to impact or lead to a target condition; a diagram that defines theoretical entities, objects, or conditions of a system and the relationships between them.

Enterprise modeling and conceptual modeling: A conceptual model is a model which represents a conceptual understanding (i.e., conceptualisation) of some domain for a particular purpose. A model is an artefact acknowledged by the observer as representing some domain for a particular purpose.

Holistic view: In most cases, a model is also a conceptual model.

Conceptual models as a result of an activity: We use the name of conceptual modeling for the activity that elicits and describes general knowledge a particular information system needs to know. The main objective of conceptual modeling is to obtain that description, which is called a conceptual schema.

Purpose-oriented modeling: Conceptual modeling is about abstracting a model that is fit-for-purpose and by this we mean a model that is valid, credible, feasible and useful.

Documentation-oriented conceptual model: A conceptual data model is a summary-level data model that is most often used on strategic data projects. It typically describes an entire enterprise. Due to its highly abstract nature, it may be referred to as a conceptual model.

Documentation and understanding viewpoint: A conceptual model of an application is the model of the application that the designers want users to understand. By using the application, talking with other users, and reading the documentation, users build a model in their minds of how to use the application. Hopefully, the model that users build in their minds is close to the one the designers intended.

And continuing: These terms are introduced by analogy to data models and database schemata. The reader may want to think of data models as special conceptual models where the intended matter consists of data structures and associated operations.

Some research challenges in conceptual modeling: Provide the right set of modeling constructs at the right level of abstraction to enable successfully communication among clients, analysts, and application programmers. Formalize conceptual-modeling abstractions so that they retain their ease-of-communication property and yet are able to (partially or even fully) generate functioning application software. Make conceptual modeling serve as analysis and development tools for exotic applications such as: modeling the computational features of DNA-level life to improve human genome understanding, annotating text conceptually in order to superimpose a web of knowledge over document collections, leveraging conceptual models to integrate data (virtually or actually) providing users with a unified view of a collection of data, extending conceptual modeling to support geometric and spatial modeling, and managing the evolution and migration information systems. Develop a theory of conceptual models and conceptual modeling and establish a formal foundation of conceptual modeling.

Another version is the following one: The conceptual level has a conceptual schema, which describes the structure of the whole database for a community of users. A conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations, and constraints. A high-level data model or an implementation data model can be used at this level.

8The slides of the keynote talk state: A conceptual model is a simplification of a system built with an intended goal in mind. An abstraction of a system to reason about it (either a physical system or a real or language-based system). A description of specification of a system and its environment for some purpose. One main conclusion that we can reach is that the distinction between "model" and "conceptual model" is not always as precise as it should be.
Conceptualisations of models: Conceptual models are nothing else as models that incorporate concepts and conceptions which are denoted by names in a given name space. A concept space \(^{10}\) consists of concepts (see [24]) as basic elements, constructors for inductive construction of complex elements called conceptions, a number of relations among elements that satisfy a number of axioms, and functions defined on elements. (see [38])

At the ER’2017 conference a special brainstorming and discussion session has been organised with the task to coin the notion of a conceptual model. It seems to be surprising that there is no commonly accepted notion of a conceptual model after more than 40 years of introduction of this concept into database research. One proposal of the brainstorming discussion was:

ER 2017 discussion proposal: A conceptual model is a partial representation of a domain that can answer a question.

As for a model, the purpose dimension determines the quality characteristics and the properties of a model.

In a nutshell, a conceptual model is an enhancement of a model by concepts from a concept(ion) space. It is formulated in a language that allows well-structured formulations, is based on mental/perception/domain-situation models with their embedded concept(ion)s, and is oriented on a modelling matrix that is a common consensus within its community of practice.

We thus meet a good number of challenges, e.g. the following ones: is there any acceptable and general notion of conceptual model; do conceptual models really provide an added and sustainable value; what are the differences between conceptual models and models; what is a model; what means conceptualisation; how to support language-based conceptual modelling; etc. This paper is oriented on these questions and tries to develop an answer to them. We restrict the investigation to conceptual models in computer science and computer engineering and thus do not consider conceptual modelling for product design, service design, other system’s design, natural and social sciences. Physical conceptual models are also left out of scope.

2 Revisiting Conceptual Modelling

2.1 State-Of-Art and State-Of-Needs

Modelling offers the benefit of producing better and understandable systems. It is based on a higher level of abstraction compared to most programming languages. Whether a model must be formal is an open question. The best approach is to consider model suites (or ensembles) that consist of a coherent collection of models which are representing different points of view and attention. We observe a resurgence in domain specific approaches that are challenged by technical, organisational and especially language design problems. UML is not the solution yet because UML Models aren’t executable but MDA needs them to be. The vast majority of UML models we have seen in industrial project are mere sketches and are informal and incomplete. They are not yet a viable basis for precise and executable models. Without precise models, no formal checking can take place. Therefore, these issues must be addressed either if modelling is well-accepted and gains significant presence in applications.

From the other side, the large body of knowledge on conceptual modelling in computer science is a results of hundreds of research papers over the last three-score years although different names have been used for it. Modelling is often based on a finalised-model-of-the-real-world paradigm despite the constant change in applications. Model quality has already been considered in a dozen papers. Modelling literacy is rarely addressed in education. Models must however be reliable, refindable, and translatable artifacts in software processes.

Conceptual modelling is supported by a large variety of tools. e.g. (see [21]). However, few of them support executable models. Of that few, far fewer still are actually rewarding to use. Conceptual models are acknowledged as mediators in the software development process. However, they are used and then not evolving with the evolution of the software. Reuse, migration, adaptation, and integration of models is still a lacuna. The lack of robust, evolution-prone and convenient translators is one reason. An environment as a constituent part for modelling and translation into a consistent, easy-to-use and -revise, seamless, and industry-quality tools is still on the agenda. Information and software systems become eco-systems. Modelling eco-systems are not yet properly addressed.

Models are also used for communication based on some injection of a name space while the community of practice uses a wealth of terms and terminology with which they express their nuances of viewpoints. So, we need a number of representation models besides the singleton graphical representation. At the same time, models must be properly formal and based on rules strictly to be followed or else having a risk of making illogical statements. Modelling must thus be based on methodologies.

2.2 Myths of (Conceptual) Modelling

Modelling and especially conceptual modelling is not yet well understood and misinterpreted in a variety of ways. It has brought a good number of myths similar to those known for software development (see [1]):

1. Modelling is mainly for documentation. The introduction of the conceptual modelling for database systems has been motivated by documentation scenario. A conclusion might be that modelling is a superfluous

\(^{10}\)We follow R.T. White (see [37, 42]) and distinguish between concepts, conceptual, conceptional, and conceptions.
activity, especially in the case that documentation is not an issue.

2. Modelling is finished with the use of the model and an initial phase. Historic development of software started with requirements which were frozen afterwards and with modelling and specifications that were complete and became frozen before realisation begins.

3. Modelling is only useful for heavyweight V-style software development. Modelling and especially conceptual modelling is abandoned due to its burden and the discovery of the complexity of the software that is targeted.

4. The collection of origins must be “frozen” before starting with modelling. Models should be plastic and stable (one of the justification and thus dependability properties), i.e. the collection of origins to be modelled could change.

5. The model is carved in stone and changes only from time to time if at all. The realisation becomes ‘alive’ and thus meets continuous change requests. The model can have some faults, errors, misconceptions, misses etc. Extensions and additional services are common for systems. So, the model has to change as well.

6. Modelling is starts with selecting and accommodating a CASE tool. Although CASE tools are useful they impose their own philosophy, language, and treatment. Moreover, CASE tools allow to become too detailed. Instead, conceptual modelling should allow to create the model that is simple as possible and as detailed as necessary.

7. Conceptual modelling is a waste of time. Developers are interested in quick success and have their own perception model in mind. It seems to be superfluous to model and better to focus solely on how to write the code.

8. Conceptual data modelling is a primary concern. Data- and structure-driven development without consideration of the usage of the data in applications results in ‘optimal’ or ‘normalised’ data structure models and bad database performance. One must keep in mind the usage of the data, i.e. use a co-design method, e.g. (see [34]).

9. The community of practice has a common understanding how to conceptually model. Modelling skills evolve over years and are based on modelling practice and experience. Further, conceptual models are based on a common domain-situation model that has to be shared within the community of practice. So, the perception models of modellers should match.

10. Modelling is independent on the language. Modelling cannot be performed in any language environment.

Language matters, enables, restricts and biases (see [43]). Understanding these and other myths allows to better understand the modelling process and the models. One way to overcome them is the development of sophisticated and acknowledged frameworks. Model-centred development (see [23]) uses models as a kernel for development of systems. Conceptual modelling ist still taught as modelling in the small whereas modelling in the large is the real challenge.

2.3 Specifics of Notions

Let us return to the list of notions given in Section 1. Each of these notions has its graces, biases, orientations, applicability, acceptability, and specifics.

Scopes of conceptual models may vary from very general models to fine-grained models. General models allow to reason on system properties whereas fine-grained models serve as a blueprint for development.

Result-oriented viewpoint: Conceptual models can be seen as the final result and documentation of an activity that follows a certain development strategy such as agile, extreme, waterfall etc. methodologies.

Communication viewpoint: Conceptual models are a means for communication and negotiation among different stakeholders.

System construction orientation: Database, information and software system development is becoming more complex, more voluminous, requires higher variety, and changes with higher velocity. So a quick and parsimonious comprehension becomes essential and supports higher veracity and an added value for the system itself.

Perception and domain-situation models are specific mental models either of one member or of the community of practice within one application area. It is not the real world or the reality what is represented. It is the common consensus, world view and perception what is represented.

Conceptual models as documentation: Models provide also quality in use, i.e. they allow to survey, to understand, to negotiate, and to communicate.

Conceptual modelling with prototypes: Models can be enhanced by prototypes or sample populations. A typical approach is sample-based development (see [16]).

Visualisation issues: Conceptual models may be combined with representation models, e.g. visualisation models on the basis of diagrammatic languages.

Biased conceptual modelling approaches: Conceptual models are often models with a hidden background, especially hidden assumptions that are commonly accepted in a community of practice in a given context and utilisation scenario.
Semiotics and semiology of conceptual modelling:
Conceptual models are often language-based. The language selection is predetermined and not a matter of consideration in the modelling process.

Quality models: Conceptual models should be well-formed and satisfy quality requirements depending on their function in utilisation scenarios.

Concepts, conceptions: The elements in a conceptual models are annotated by names from some name space. These names provide a reference to the meaning, i.e. a reference to concepts and conceptions in a concept space.

Conceptual model suites: Models can be holistic or consist of several associated models where in the latter case each of them represents different viewpoints. For instance, a conceptual database model consists of a schema and a number of derived views which represent viewpoints of business users.

Normal models: Conceptual models represent only certain aspects and are considered to be intentionally enhanced by elements that stem from commonsense, consensuses, and contexts.

A normal models (called ‘lumped’ model in [45]) is a part of the model that is considered to be essential and absolutely necessary. The normal model has a context, a community of practice that puts up with it, a utilisation scenario for which is is minimally sufficient, and a latent – or better deep – model on which it is based (see [45] for ‘base’ model). The deep model combines the unchangeable part of a model and is determined by the grounding for modelling (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities), the outer directives (context and community of practice), and the basis (assumptions, general concept space, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense) of modelling. The (modelling) matrix consists of the deep model and the modelling scenarios. The last ones are typically stereotyped in dependence on the chosen modelling method.

This variety of viewpoints to conceptual models illustrates the different requirements and objectives of models. So, we might ask whether a common notion of a conceptual model exists or whether we should use different notions.

2.4 Problems and Challenges

Conceptual modelling techniques suffer from a number of weaknesses. These weaknesses are mainly caused by concentration on database modelling and by non-consideration of application domain problems that must be solved by information systems. We follow the state-of-the-art analysis of A. van Lamsweerde (see [40, 41]) who gave a critical insight into software specification and arrive with the following general weaknesses for conceptual modelling of information and database systems:

Limited scope. The vast majority of techniques are limited to the specification of data structuring, that is, properties about what the schema of the database system is expected to do. Classical functional and nonfunctional properties are in general left outside or delayed until coding.

Poor separation of concerns. Most modelling approaches provide no support for making a clear separation between (a) intended properties of the system considered, (b) assumptions about the environment of this system, and (c) properties of the application domain.

Low-level schematology. The concepts in terms of which problems have to be structured and formalized are concepts of modelling in the small - most often, data types and some operations. It is time to raise the level of abstraction and conceptual richness found in application domains.

Isolation. Database modelling approaches are isolated from other software products and processes both vertically and horizontally. They neither pay attention to what upstream products in the software might provide or require nor pay attention to what companion products should support nor provide a link to application domain description.

Cost. Many information systems modelling approaches require high expertise in database systems and in the white-box use of tools.

Poor tool feedback. Many database system development tools are effective at pointing out problems, but in general they do a poor job of (a) suggesting causes at the root of such problems, and (b) proposing better modelling solutions.

Modern modelling approaches must not start from scratch. We can reuse achievements of database modelling in a systematic form and thus maintain theories and technologies while supporting new paradigms.

Constructiveness. Models of information systems can be built incrementally from higher-level ones in a way that guarantees high quality by construction. A method, is typically made of a collection of model building strategies, paradigm and high-level solution selection rules, model refinement rules, guidelines, and heuristics. Some oft hem might be domain-independent, some others might be domain-specific.
Support for comparative analysis. Database models depend on the experience of the developer, the background or reference solutions on hand, and on preferences of developers. Therefore, the results within a team of developers might need a revision or a transformation to a holistic solution. Beyond the modelling qualities we may develop precise criteria and measures for assessing models and comparing their relative merits.

Integration. Tomorrow’s modelling should care for the vertical and horizontal integration of models within the entire analysis, design, development, deployment and maintenance life cycle - from high-level goals to be supported by appropriate architectures, from informal formulation of information system models to conceptual models, and from conceptual models to implementation models and their integration into deployment of information systems.

Higher level of abstraction. Information systems modelling should move from infological design to holistic co-design of structuring, functionality, interactivity and distribution. These techniques must additionally be error-prone due to the complexity of modern information systems. These abstraction techniques may be combined with refinement techniques similar to those that have been developed for the abstract state machines.

Richer structuring mechanisms. Most modelling paradigms of the modelling-in-the-small approach available so far for modularising large database schemata have been lifted from software engineering approaches, e.g., component development. Problem-oriented constructs be developed as well as model suites that provide a means for handling a variety of models and viewpoints.

Extended scope. Information system development approaches need to be extended in order to cope with the co-design of structuring, functionality, interactivity and distribution despite an explicit treatment of quality or non-functional properties.

Separation of concerns. Information system modelling languages should enforce a strict separation between descriptive and prescriptive properties, to be exploited by analysis tools accordingly.

Lightweight techniques. The use of novel modelling paradigms should not require deep theoretical background or a deep insight into information systems technology. The results or models should be compiled to appropriate implementations.

Multi-paradigm modelling. Complex information systems have multiple facets. Since no single modelling paradigm or universal language will ever serve all purposes of a system. The various facets then need to be linked to each other in a coherent way.

Multilevel reasoning and analysis. A multi-paradigm framework should support different levels of modelling, analysis, design and development - from abstract and general to deep-level analysis and repairing of detected deficiencies.

Multi-format modelling. To enhance the communicability and collaboration within a development and support team the same model fragment must be provided in a number of formats in a coherent and consistent way.

Reasoning in spite of errors. Many modelling approaches require that the model must be complete before the analysis can start. We claim that is should be made possible to start analysis and model reasoning much earlier and incrementally.

Constructive feedback from tools. Instead of just pointing out problems, future tools should assist in resolving them.

Support for evolution. In general, applications keep evolving due to changes in the application domain, to changes of technology, changes in information systems purposes etc. A more constructive approach should also help managing the evolution of models.

Support for reuse. Problems in the application domain considered are more likely to be similar than solutions. Models reuse should therefore be even more promising than code reuse.

Measurability of modelling progress. To be more convincing, the benefits of using information models should be measurable as well as their deficiencies.

This list of theories, solutions and methodological approaches is not exhaustive. It demonstrates, however, that modelling in the large and modern information systems modelling require specific approaches beyond integration of architectures into the analysis, design and development process.

2.5 The Research Issue

Let us reconsider the notions presented in Section 1. Table 1 compares essential properties of models. Missing model elements are denoted by \( \text{not} \).g(iven).

We observe that dependability is often either implicit or not considered in the model notion. Implicitness is mainly based on the orientation to normal models. The model matrix and especially the deep model are considered to be agreed before developing the model.

The origin is too wide in most cases. Models are not oriented towards representing some reality or the world. They are typically based on some kind of agreement made within a community of practice and according to some context, i.e.
they reflect some domain-situation model or more generally some mental model. They might represent a perception model of some members of the community practice. They say what the phenomena in the given domain are like.

Table 1 directs to a conclusion that the function is mainly oriented towards description and partially prescription for systems development. The notion of the conceptual model has, however, mainly considered in system construction scenarios.

Concepts are often hidden behind the curtain of conceptual models. A conceptual model does not reflect the reality. Instead it reflects the mental understanding within its utilisation scenario. These observations show now directly some open issues that should be solved within a theory and practice of conceptual modelling. Let us state some of them.

---

Research question 1. What are the origins for conceptual models? Are these mainly domain-situation and perception models from one side and systems on the other side?

Research question 2. How tightly conceptual models are bound to their modelling matrix and especially their deep model? To what extent conceptual models are normal models that are intentionally combined with their deep models?

Research question 3. Which functions have conceptual models in which utilisation scenarios? Which properties must be satisfied by conceptual models in these scenarios? Which purposes and goals can be derived?

Research question 4. What is the role of the community of practice in conceptual modelling? Which kind of model supports which community in which context?

Research question 5. Conceptual modelling is less automated and more human dependent than any other development, analysis, and design process for information systems. Is it a highly creative process? Is there any formalisation and foundation for this process?

Research question 6. Since models must not be conceptual models (see models in [39]), we might ask whether there exists a set of characteristics or criteria that separate...
a conceptual model from a model that is not conceptual. What are the concept space that can be used for an enhancement of a model by concepts or conceptions?

3 The Nature of Models

3.1 The Notion of a (Conceptual) Model

The model is an utterance and also an imagination. As already stated above (see also [39]), a model is a well-formed, adequate, and dependable instrument that represents origins and functions in some utilisation scenario. A model is a representation of some origins and may consist of many expressions such as sentences. Adequacy is based on satisfaction of the purpose or function or goal, analogy to the origins it represents and the focus under which the model is used. Dependability is based on a justification for its usage as a model and on a quality certificate. Models can be evaluated by one of the evaluation frameworks. A model is functional if methods for its development and for its deployment are given. A model is effective if it can be deployed according to its portfolio, i.e. according to the tasks assigned to the model. Deployment is often using some deployment macro-model, e.g. for explanation, exploration, construction, documentation, description and prescription.

Models function as instruments or tools. Typically, instruments come in a variety of forms and fulfill many different functions. Instruments are partially independent or autonomous of the thing they operate on. Models are however special instruments. They are used with a specific intention within a utilisation scenario. The quality of a model becomes apparent in the context of this scenario.

Model development is often targeted on normal models and implicitly accepts the deep model. A model is developed for some modelling scenarios and thus biased by its modelling matrix. The deep model and the matrix thus ‘infect’ the normal model.

Within the scope of this paper, we concentrate on representation models as proxies. So, a model of a collection of origins, within some context, for some utilisation scenario and corresponding functions within these scenarios, and for a community of practice is

- a relatively enduring,
- accessible
- but limited
- internal and at the same time external
- representation of the collection of origins.

The model becomes conceptual by incorporation of concepts and conceptions commonly accepted, of ideas provided by members from the community of practice, or of general well-understood language-like semiotic components. One main utilisation scenario for conceptual database model is system construction\(^1\). In this case, the conceptual model thus becomes predictively accurate for the system envisioned and technologically fruitful. The model is an utterance and also an imagination. Other scenarios for conceptual models are: system modernisation, explanation, exploration, communication, negotiation, problem solving, supplantation, documentation, and even theory development.

Conceptual models must not limited to representation of static aspects of systems. They can also be used for representation of dynamic aspects such as business stories, business processes, and system behaviour. The carrier of representation is often some language. In this case, a conceptual model can be considered to be an utterance with a collection speech acts. The model itself can be then build on well-formedness rules for its syntax, semantics, and pragmatics, or more general of semiotics and semiology. According to J. Searle (see [33]), a speech act consists of uttering elements, referring and predicating, requesting activities, and causing an effect. Whether at all and which language is going to be used is a matter of controversy too.

3.2 Facets of a Conceptual Model

1. The conceptual model is a result of a perception and negotiation process. The conceptual model represents mental models, especially domain-situation models or a number of perception models. Domain-situation models represent a settled perception within a context, especially an application. Perception models might differ from the domain-situation model. They are personal perceptions and judgements of a member of the community of practice. Maturity of conceptual models is reached after the community of practice negotiated different viewpoints and has found an agreement.

2. The conceptual model represents its collection of origins. Considerations about what to model and what not to model are expressed via the adequacy criteria, especially for analogy to its origins, for focusing on specifics of the origins, and also on well-formedness of the model. The conceptual model does not represent a real world or a problem domain. It is already based on perception models of users about this problem domain or on domain-situation models of a user community on this problem domain.

3. The conceptual model is an instrument. The conceptual model is used in some utilisation scenario by its users. So it functions in this utilisation scenario. It should describe in a more abstract way compared to the origins how the user conceives it and thus does not target on describing the origins.

4. The deep model underpins the conceptual model. The deep model consists of all elements that are taken for granted, are considered to be fixed, and are common within the context for the community of practice. Elements of this model are symbolic generalizations as formal or readily formalisable components or laws or law schemata, beliefs in particular heuristic and ontological models or analogies.

\(^1\)Notice however that the first introduction of conceptual data models has been oriented on a documentation scenario.
supplying the group with preferred or permissible analogies and metaphors, and values shared by the community of practice as an integral part and supporting the choice between incompatible ways of practicing their discipline. There is no need to redevelop this model. So, the normal model only display those elements that are additionally introduced for the model.

5. The conceptual modelling matrix. The modelling matrix combines the deep model with the typical utilisation scenarios that are accepted by a community of practice in a given context. It specifies a guiding question as a principal concern or scientific interest that motivates the development of a theory, and techniques as the methods an developer uses to persuade the members of the community of practice to his point of view. Although often not explicitly stated, the model matrix consists of a number of components: the objectives, inputs (or experimental factors), outputs (or responses), content requests, grounding, basis, and simplifications. The matrix sets a definitional frame for the normal model. It might support modelling by model stereotypes. The agenda of the modelling method is derived from the matrix. The matrix determines also a specific treatment of adequacy and dependability for a model.

6. The performance and quality criteria. The model is a persistent and justified artifact that satisfy a number of conditions according to its function such as empirical corroboration according to modelling objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability, and by stability and plasticity within a collection of origins. The quality characteristics bound the model to be valid, credible, feasible, parsimonious, useful, and at the same time as simple as possible and as complex as necessary.

7. The model is the main ingredient of a modelling method. Sciences and technologies have developed their specific deployment of models within their investigation, analysis, development, design etc. processes. The deep model and the matrix are often agreed. The central element of all modelling methods is the model that is used as an instrument in scenarios which have been stereotyped for the given modelling method. The modelling method typically also includes design of a representation model (or a number of such). The representation model of the (conceptual) model may be based on approaches such as diagramming and visualisation. It uses a set of predefined signs: icons, symbols, or indexes in the sense of Peirce.

3.3 Sources for Conceptual Models: Domain-Situation and Perception Models

The domain-situation model is build by a community of practice on a semantical level. It refers to the world-as-described-and-conceived-by-the-deep-model. It thus forms the deep understanding behind the conceptual model. This deep internal structure of the conceptualisation is commonly shared in the community, abstracts from accidental origins, uses a partial interpretation, exhibits (structural) hidden similarities of all origins under consideration, and presents the common understanding in the community. It gives thus a literal meaning to the domain. The context for the conceptual model is typically governed by domain-situation models. The domain-situation model is thus one source for the conceptual model.

A domain-situation model might or might not exist. It shapes, however, what is seen in an application domain and how to reason about what is seen. They represent some common negotiated understanding in the application domain. It may represent the application domain as it is or the application domain as it makes sense to be characterised, categorised or classified in one way rather than another given certain interests and aptitudes or more generally given certain background.

The second source for conceptual models is a collection of perception models that are provided and acknowledged by members of this community of practice. A perception model is one kind of epistemological mental model with its verbal, visual and other information compiled on the basis of cognitive schemata. It organises, identifies, and interprets observations made by the member. It does not need to know the deep facts or essential properties of the origins in order to succeed in communicating about them or to reason. The perception model typically follows the situation that it represents. It is however often underdetermined and thus may also partially contradictory. So it parallels and imitates parts of the reality (‘Gestalt’ notion of the model). They provide a partial understanding, refer to some aspect, may use competing sub-models about the same stuff, and may set alternatives on meaning. It is build by intuitive, discursive and evidence-backed perception, by imagination, and by comprehension. It is shaped by learning, memorisation, expectation, and attention. Perception models serve as an add-on beyond domain-situation models.

These two sources for conceptual models depend on the community of practice. So, different communities might use different kinds of verbal and nonverbal representation. Although they provide a literal meaning to the conceptual model they must not be explicitly stated within the conceptual model. They serve as the origin for the conceptual model and thus might not be explicitly incorporated into the conceptual model. The conceptual model may have its deep background, i.e. its basis and especially its grounding.

Both origins are not complete. Typically the scope of both models is not explicit. There are unknown assumptions applied for description, unknown restrictions of the model, undocumented preferences and background of the community of practice, and unknown limitations of the modelling language. Classically we observe for members of a community of practice that

• they base their design decisions on a “partial reality”, i.e. on a number of observed properties within a part of the application,
• they develop their models within a certain context,
• they reuse their experience gained in former projects and solutions known for their reference models, and
• they use a number of theories with a certain exactness and rigidity.

The conceptual model to be developed is deeply influenced by these four hidden factors.

4 Conceptualisation of Models

The domain-situation model and also partially the perception model are commonly using concepts. Conceptual models reuse such concepts from these origins and thus inherit semantics and pragmatics from these models. Further, conceptualisation may also be implicit and may use some kind of lexical semantics of these models, e.g. word semantics, within a commonly agreed name space.

4.1 Concepts and Conceptions

Various notions of concept has been introduced, for instance, by J. Akoka, P. Chen, H. Kangassalo, R. Kauppi, A. Paivio, and R. Wille (see [6, 14, 22, 20, 27]). Artificial intelligence and mathematical logics use concept frames. Ontologies combine lexicology and lexicography. Concepts are used in daily life as a communication vehicle and as a result of perception, reasoning, and comprehension. Concept definition can be given in a narrative informal form, in a formal way, by reference to some other definitions etc. Some version may be preferred over others, may be time-dependent, may have a level of rigidity, is typically usage-dependent, has levels of validity, and can only be used within certain restrictions. We also may use a large variety of semantics (see [32]), e.g., lexical or ontological, logical, or reflective.

We distinguish two different meanings of the word ‘concept’ (see [42]):

1. Concepts are general categories and thing of interest that are used for classification. Concepts thus have fuzzy boundaries. Additionally, classification depends on the context and deployment.

2. Concepts are all the knowledge that the person has, and associates with, the concept’s name. They are reasonable complete in terms of the business.

Conceptions (see [42]) are systems of explanation. They are thus more difficult to describe.

The typical definition frame we observed is based on definition items. These items can also be classified by the kind of definition. Concepts may simultaneously have different descriptions. Competing description may differently represent the same concept depending on context (e.g. time, space), validity, usage, and preferences of members of the community of practice. A concept may have elements that are necessary or sufficient, that may be of certain rigidity, importance, relevance, typicity, or Fuzziness. Based on the generalisations of the approach that has been proposed by G.L. Murphy (see [24, 35]), concepts are defined in a more sophisticated form as a tree-structured structural expression.

SpecOrderedTree(StructuralTreeExpression
(DefinitionItem, Modality(Sufficiency, Necessity),
Fuzziness, Importance, Rigidity,
Relevance, GraduationWithinExpression, Category))

Concept may be regarded as the descriptive and epistemic core units of perception and domain-situation models. These origins govern the way how a concept can be understood, defined, and used in a conceptual model. The conceptual model inherits thus concepts and their structuring within a concept space, i.e. conceptions.

4.2 Conceptualise

Conceptualisation and semantification are orthogonal concerns in modelling. Conceptual modelling is based on concepts that are used for classification of things. Concepts have fuzzy boundaries. Additionally, classification depends on the context and deployment. Conceptual4 modelling uses concepts which are systems of explanation. Semantification (see [9]) improves comprehensibility of models and explicit reasoning on elements used in models. It is based on name spaces or ontologies that are commonly accepted in the application domain. Conceptual models are models enhanced by concepts and integrated in a space of conceptions.

Conceptualisation injects concepts or conceptions into models. These enriched models reflect those concepts from commonly accepted concept space. The concept space consists of a system of conceptions (concepts, theoretical statements (axioms, laws, theorems, definitions), models, theories, and tools). A concept space also may include procedures, conceptual (knowledge) tools, and associated norms resp. rules. Is is based on paradigms which are corroborated.

4.3 Dependability of Conceptual Models

Models must be dependable, i.e. justified from one side and qualitatively certified from the other side. Justification can be based on the domain-situation and perception models and the relation of the conceptual models to these models. If however such models are not available or of low quality then justification will become an issue. Quality certification is an issue of pragmatism and of added value of the conceptual model. So, we target on a high quality conceptualisation. Conceptualisation may be based on the seven principles of Universal Design (see [29]). Typical mandatory principles are usefulness, flexibility, sim-

---

4Conceptual modelling is performed by a modeller that directs the process based on his/her experience, education, understanding, intention and attitude. Conceptual models are using/incorporating/integrating concepts (see [42]) Conceptual modelling aims at development of concepts.
5.1 Slim, Light, and Concise Versions for

time-restricted, activity-driven, and context-dependent.

dependent on deep models and thus incomplete, revisable,
models, i.e. of mental models in general. Perception is
abstraction throughout the perception and domain-situation
and learning. It is however based on understanding and

craft learning. It is however based on understanding and
abstraction throughout the perception and domain-situation
models, i.e. of mental models in general. Perception is
dependent on deep models and thus incomplete, revisable,
time-restricted, activity-driven, and context-dependent.

5 Conclusion: Towards a Notational
Frame for Conceptual Models

Conceptual modelling is not yet a science or culture. It is
rather a craft or even an art. It can be learned similar to
craft learning. It is however based on understanding and
abstraction throughout the perception and domain-situation
models, i.e. of mental models in general. Perception is
dependent on deep models and thus incomplete, revisable,
time-restricted, activity-driven, and context-dependent.

5.2 A Proposal for a Light Version: Conceptual Model ⊕ Model ⊇ Concepts

Conceptual modelling is not yet a common method in sci-
ence (see [31]). Systems can be built without any concept-
ual model. It seems that there is no need for a formal
conceptual modelling process. It seems to be too restric-
tive to require a full conceptual model. Performance and
quality criteria are not commonly agreed. The science of
conceptual modelling is still missing.

The main bottleneck is however the missing notion of
a conceptual model. The conceptual model is a specific
model and is based on conceptualisation. It might be
language-bound. It is probably the most important aspect
of system construction in computer science and computer
engineering. It is however the most difficult and least un-
derstood. Minimal justification characteristics of models
are classical viability, i.e. corroboration, validity, credi-
bility, rational coherent and conform, falsifiable, stability
against origin collection change. Minimal quality char-
acteristics of models are the one for quality in use (e.g.
usability, aptness for the function and purpose, value for
the utilisation scenario, feasibility). Minimal performance
characteristics are timely, elegant and feasible usage within
the given context for their community of practice according
to their utilisation scenario and their competencies or more
general their profiles.

So, we might conclude for a light version: A concep-
tual model is a well-formed, adequate and depend-
able instrument that functions within its specific util-
isation scenario, that represents origins, and that is
enhanced by concepts from a concept(ion) space.

Therefore, the incorporation of concepts and the concep-
tions is one main difference to the model.

5.3 Lacunas of Conceptual Modelling

Since conceptual modelling is still more an art than a sci-
ence and a culture of conceptual modelling is still beyond
the horizons, we need

• an understanding of the area of conceptual modelling;
• a theory, techniques, and engineering of conceptuali-
sation;
• an integrated multi-view approach for the needs and
the capabilities of the members of the community of
practice;
• a refifiable definition of the conceptual model with all
three versions, i.e. a simplified version, a fully fledged
version, and an assessable version;
• a working approach with intentional and thus latent
matrices and deep models for daily practice; and
• an understanding of language use in conceptual mod-
elling.
These lacunas do not limit usability, usefulness, and utility of conceptual models. Conceptual database models improve from one side system comprehension. They allow to indicate associations among system elements, reduce the effect of bad implementation, provide abstraction mechanisms, support prediction of system behaviour, provide an elegant and adequate overview of the system at various levels of abstraction, support the construction of different user views, and cross-reference multiple viewpoints. From the other side, the reduce the developers, maintainers and programmers overhead. They support a simple and free navigation through components of the database system, provide an easy deduction of various viewpoints that represent the needs of business users, support concentration and focusing in evolution and maintenance phases, display the decisions made during development, indicate opportunities for further development and system maintenance, reduce the effort by reuse of design and development decisions that have already been made, and use a comfortable and effective visualisation. So, conceptual models are not restricted to construction scenarios or to database modelling.

We realise that the development and the acceptance of a notion of conceptual model follows the 13 Commandments stated (see [5]):

1. Thou shalt choose an appropriate notation.
2. Thou shalt formalise but not overformalise.
3. Thou shalt estimate costs.
4. Thou shalt have a formal methods guru on call.
5. Thou shalt not abandon thy traditional development methods.
6. Thou shalt document sufficiently.
7. Thou shalt not compromise thy quality standards.
8. Thou shalt not be dogmatic.
9. Thou shalt test, test, and test again.
10. Thou shalt reuse.
11. Thou shall meet intentions of all members of the community of practice.
12. Thou shall provide a usable notation, i.e. for verification, validation, explanation, elaboration, and evolution.
13. Thou shall be robust against misinterpretation, errors, etc.

References

Qualitative and Quantitative Models in Data Science

Bernhard Thalheim
Christian Albrechts University Kiel, Department of Computer Science, D-24098 Kiel, Germany
thalheim@is.informatik.uni-kiel.de

Abstract

Data are considered to be the oil of the 21\textsuperscript{th} century. They are also a rich source for many sciences, especially those that use observational data for development of an understanding behind the data. They are used to gain an insight into the discipline based on observations. This insight may result in a quantitative theory offer. The main target is however a theory that explains the data. We develop a model-backed approach to theory development based on quantitative theory offers. Models are becoming the mediator between quantitative and qualitative theories. Models can be systematically developed based on a layering approach.

1 Introduction

1.1 From Empiric Sciences to Data Science

Data science is considered to be a new stage of scientific research. Data science is based on analysis of data resources. The analysis asks the right questions with efficient processing algorithms, machine learning and cognitive computing techniques, refined statistical models, and innovative visions of how to more effectively extract the relevant data assets and scrutinise them fast with more sophisticated results. It goes beyond empirical sciences, theory-oriented research, and computational research [12]. Disciplines often use a combination of empirical research that mainly describes natural phenomena, of theory-oriented research that develops concept worlds, of computational research that simulates complex phenomena and of data exploration research. Thus, Figure 1 distinguishes four stages of sciences according to [12].

Data science discovers pattern and generates insights in data sources or data proxies. It is based on raw data and build these insights based on knowledge from the scientific discipline and application domain. It provides models, recommendations, and potential

Proceedings of the XIX International Conference “Data Analytics and Management in Data Intensive Domains” (DAMDID/RCDL’2018), Moscow, Russia, October 09-12, 2018
reports of regularities, and (3) systematic explanatory schemes (theories).”  

Figure 6 displays this quantitative underpinning and a number of theoretical concepts.

![Figure 6: The current state-of-the-art in the data science situation.](image)

**Figure 2: The current state-of-the-art in the data science situation.** We start with some data, e.g. proxy data. We \((g^c)^{-1}\) derive proxy concepts (or concepts) and form some proposals for \((h^c)^{-1}\) formation of a proposal of a potential explaining theory, i.e. a theory request (or theory offer). Proxy sources can be aggregated and \((f^c)^{-1}\) condensed and thus become quantitative sources which are the basis for \((g^c)^{-1}\) formation of quantitative concepts. These quantitative concepts are \((h^c)^{-1}\) embedded into theory offers (or, resp., theory requests for proxy requests) and are the basis for a theory offer that serves as an explanation for the theory request. Quantitative concepts can be \((F^c)^{-1}\) mapped back to proxy concepts. Proxy-based research and quantitative research is well-integrated if the diagram is commuting, i.e. \(F^c(g^c(f^c(sources))) = g^c(sources)\).

Theories can be build on the basis of theoretical concepts which are supported by sources. Quantitative concepts should be associated with qualitative concepts. The association can only be developed in the case when the association among the data has been clarified. So far, the explanations that can be generated are mainly developed for explaining the observations made on the basis of proxies. We arrive therefore with the following problem:

**Research challenge:** How we can close the gap between quantitative theory offers and qualitative theories within the setting of data science?

### 1.3 A Typical Data Science Application

Investigative modelling at CRC 1266 [1, 16] aims at exploring and explaining transformations in societies as “processes leading to a substantial and enduring re-organisation” [1] of any or all aspects of the human social, cultural, economic, and environmental relations. Proxies are observations for main concepts in Figure 3. These main concepts need however a

\[\text{h}^c \to \text{theory request} \\to \text{explanation} \downarrow \text{h}^c \to \text{theory offer} \downarrow \text{h}^c \to \text{theory} \]

**Figure 3: Theoretical concepts to be investigated in the CRC 1266**

### 1.4 The Storyline of the Paper

We develop an approach to data science based on models. The ladder in Figure 1 is thus supported by models in the form depicted in Figure 4

| Models as starting point for hypotheses, investigation, pattern detection, ... |
| Models as mediator or starting point in inverse modelling, ... |
| Models for representation, communication, understanding, learning ... |
| Models for visualisation of phenomena, experiments, observations, ... |

**Figure 4: The four kinds of models in scientific research**

Models are instruments that function in utilisation scenarios. One of these scenarios might be the development of a theory for a theory offer. We will show in the sequel how this approach can be systematically applied to development of mediating models that close the gap in Figure 6. We start with a notion of model in Section 2. Six research questions are developed which are answered in Sections 3 and 4. Next we develop a model construction approach in Section 3. Finally, we apply this approach to data science and use models as mediators in Section 4.

### 2 Models

Models are widely used in life, technology and sciences. Their development is still a mastership of an artisan and not yet systematically guided and managed. The main advantage of model-based reasoning is based on two properties of models: they are focused on the issue under consideration and are thus far simpler than the application world and they are reliable instruments since both the problem and the solution to the problem can be expressed by means of the model due to its dependability. Models must be sufficiently comprehensive for the representation of the domain under consideration, efficient for the project.

---

\(^1\)We restrict the mindmap to main concepts and do not display the full concept network. For details see the website of the project.
solution computation of problems, accurate at least within the scope, and must function within an application scenario.

**Research question 1:** Can models be used for resolving the gap between theory offers in quantitative research and theories in qualitative research?

Consider for instance the CRC 1266 application: Transformation is considered in this context as a phenomenon that requires detailed description of features and hence quantitative data are necessary for descriptions by empirical models and simulations. Models mediate between quantitative theories and qualitative theories. Models are applied in hypothetical and investigative scenarios, should support causal reasoning as well as network-oriented reasoning, and are developed in an empiric framework.

### 2.1 The Notion of Model

Let us first briefly repeat our approach to the notion of model:

A model is a well-formed, adequate, and dependable instrument that represents origins and that functions in utilisation scenarios. [10, 27, 28]

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice (CoP) within some context and correspond to the functions that a model fulfills in utilisation scenarios.

The model should be well-formed according to some well-formedness criterion. As an instrument or more specifically an artifact a model comes with its background, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background is often given only in an implicit form. The background is often implicit and hidden.

A well-formed instrument is adequate for a collection of origins if it is analogous to the origins to be represented according to some analogy criterion, it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its purpose.

Well-formedness enables an instrument to be justified by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins.

The instrument is sufficient by its quality characterisation for internal quality, external quality and quality in use or through quality characteristics [26] such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

A well-formed instrument is called dependable if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

### 2.2 Functions of Models

Models are used as instruments in certain utilisation scenarios such as as communication, reflection, understanding, negotiation, explanation, exploration, learning, introspection, theory development, documentation, illustration, analysis, construction, description, and prescription. They have to fulfill a number of specific functions in these scenarios. Typical functions of models as instruments in scenarios are (a) cognition, (b) explanation and demonstration, (c) indication, (d) variation and optimisation, (e) projection and construction, (f) control, (g) substitution, and (h) experimentation [31].

### 2.3 Model $\uparrow = $ Normal Model $\times$ Deep Model

A model consists of a normal model that is combined with some deep model similar to the visible (or exterior) and invisible parts of an iceberg [16, 29, 30]. The deep model reflects (α) the intentions of the problem world, (β) the accepted understanding within the community of practice, (γ) the context of the application domain, (ε) the background that is commonly accepted in the problem and application domain, and (δ) the general restrictions to the origins that might be considered. The deep model allows to derive a part of the justification and adequacy of a model.

The normal model reflects the collection of origins that are currently under consideration. Both the deep and the normal model are dependent on the functions that a model should play in application scenarios. Development of models is often restricted to development of a normal model under the assumption that the deep model is given by the modelling method, the context, the community of practice, and the function that the model has to play in a given scenario. The modelling methods also determined the methods that are used for model development. It might also include the utilisation methods.

**Research question 2:** Can we separate the deep model from the normal model in such a way that the model can be composed of the deep model and of the normal model?

If the answer to this question is positive then we might try to consider the model as an enhancement of the deep model. In this case, the development of a model can be layered.

**Research question 3:** How can be development of a model layered into the development of a deep model followed by the development of the normal model?

We may now ask us whether this approach is universal. The answer will be negative if the notion of model also includes models with intractable deep models, e.g. for metaphors, parables, or physical representations. We might however concentrate on models in sciences and technology.
2.4 Model Suites

Models may be given as a holistic instrument that combines all aspects into one model. The approach is often too challenging. A simpler approach is the consideration of a model as a model suite (or model ensemble) [8, 25] that consists of a coherent collection of models which are representing different points of view and attention. It is extended by an explicit association or collaboration schema among the models, controllers that maintain consistency or coherence of the model suite, application schemata for explicit maintenance and evolution of the model suite, and tracers for the establishment of the coherence.

Research question 4: Exists there a systematic approach to model development that is based on a co-development of normal models and deep models? Which additional models should be integrated into the model suite?

2.5 Generic and Specific Models

Model development does not start from scratch. We often start with generic models. A generic model [16] is a model which broadly satisfies the purpose and broadly functions in the given utilisation scenario. It is later tailored to suit the particular purpose and function. Generic models can be calibrated to specific models through a process of data or situation calibration, refinement, concretisation, context enhancement, or instantiation.

Research question 5: Can we develop normal models starting with a generic model and are they still integratable with the deep model?

If the answer is positive then generic normal models can be calibrated to specific normal models through a process of data or situation calibration, refinement, concretisation, context enhancement, or instantiation.

2.6 Data Mining as a Success Story

In [16], we developed the V-model to data mining based on a separation of the data mining process into the domain perspective with its domain world of users from a community of practice, the modelling perspective with a model world, and the data perspective in a data world. Users are interested in solution of certain problems an application world, share the context and also the scientific and technological background. The classical data model mining process uses these perspectives for a stepwise development of a model that allows to solve their problems, e.g. (1) by modelling the problem and the issues under consideration, (2) by preparing the data world for development and enhancement of models, (3) by applying data mining algorithms for pattern detection and model development, and (4) by using the model for development of some solution for the problems and thus augmenting the application domain world. The model development process itself can be understood a multi-iterative guided procedure that has its flow of activities.

This approach extend the classical CRISP framework [4] and other approaches to systematic data mining, e.g. [15]. Each of these approaches has its capacity and potential as well as its threats and limitations. The question is now:

Research question 6: Can we generalise a data mining setting to model development for data science in such a way that models mediate between theory offers and theories?

Data analysis and model suite development currently inherit success stories in a similar application. These success stories follow some kind of a meta-pattern and result in a specific data mining process as an example of a modelling method or modelling mould. Data mining starts with exploring and understanding the data mining project, its data, and a general setting of principles of modelling. After the project and the nature of the data is understood, data are preprocessed and prepared for the application of algorithms. Next pattern within the data are investigated. This pattern analysis results in clusters, maps, association or collaboration schema among the models, and are used during specification. The result is another model. Finally, the models are evaluated and potentially deployed. If the evaluation does show that the models satisfy quality criteria, we revise the models.

3 A General Model Composition Approach

Engineering and software engineering (e.g. [13, 22]) distinguish between the five primary development dimensions:

Activities (‘how’) describe the way how the work is performed and the practises for the work.

Work products (‘what’) are the result of the specification and are used during specification.

Roles (‘who’) describe obligations and permissions, the involvement of actors in the specification process.

Aspects (‘where’) are used for separation of concern during the specification process.

Resources (‘on which basis’) are the basis for the specification.

Since a model (suite) is also a work product we may refine this approach to model engineering. The modelling method we will use is similar to the modelling method in mathematics [3].
3.1 Separating Modelling into Layers

Model suite development results in a number of models: deep, generic, specific, and normal models. Since any model has its deep elements we may start with development of this deep model. In many cases, we might use reference model or generic models (or tactical model frames like those we use in data mining and analysis). Let us investigate whether a layered approach can be applied within a five-layered separation of concern and aspects. The separation into layers generalises the approaches used in mathematics, e.g. separation by Craig’s interpolation theorems.

Classical modelling often intentionally presupposes the initialisation and intrinsic layers and assumes that these layers cannot be reconsidered and specifically changed according to the functions. We loose, however, the understanding of the model and cannot understand why the model is dependent without an understanding of these layers.

(I) The Initialisation Layer

The W*H specification pattern [9], can be applied to model initialisation as well as includes then the following set of statements:

- a plan, function, and purpose dimension (model as a conception: ‘wherefore’, ‘why’, ‘to what place or end’, ‘for when’, ‘for which reason’) within a scenario in which the model is going to be used as an instrument,
- a user or CoP dimension (‘who’, ‘by whom’, ‘to whom’, ‘whichever’) that describes the task portfolio in the CoP and profile of users including beliefs, desires and intentions,
- an application and a problem dimension (‘in what particular or respect’, ‘from which’, ‘for what’, ‘where’, ‘whence’), and
- additionally, the added value dimension (evaluation).

The initialisation layer may also be enhanced by a contrast space for user-related separation of a model and a relevance space that is dependent on the user [11]. The contrast and relevance spaces as a form of mind-setting also define what is not of interest.

(II) The Enabling Setup Layer

The intrinsic setup defines the opportunity space and the infrastructure for the model. The results will be from one side a deep model and from the other side a modelling framework or modelling mould that guides and govern next activities. We define the context and the most of the background (the grounding (paradigms, postulates, restrictions, theories, culture, foundations) and the basis (assumptions, concept world, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense)) of the model. The context, extrinsic, and strategic dimension answers question like ‘at or towards which’, ‘where about’, ‘to what place or situation’, and ‘when’.

Additionally, we decide which methodology and environment seem to be the most effective and purposeful. The development and deployment dimension (‘how’, ‘whence’, ‘what in’, ‘what out’, ‘where’) defines the modelling methodology, i.e. the modelling mould.

(III) The Extrinsic Source Reflection Layer

Strategem We separate the deep model elements from elements of the normal model. According to the model function, the normal model represents extrinsic elements of potential origins based on their content and thus answers questions such as ‘what’, ‘with which’, and ‘by means of which’. It reflects the extrinsic theory essentials that are necessarily to be represented, e.g. conceptions or pre-conceptions from the theory that is underpinning the application. The normal model can be build from scratch (‘greenfield’ modelling). It is more usual based on the experience gained so far. The latter case thus starts with a generic or reference model that might incorporate parameters. The extrinsic source reflection layer can be understood as a tactical layer.

(IV) The Operational Customisation Layer

Generic or general normal models are adjusted to those that a best fitted to those origins that are considered for the application. The operational customisation layer is sometimes holistically handled with extrinsic reflection. Inverse modelling is the general case however. It instantiates parameters, adapts the normal model to those origins (or data sources) that are really under consideration, prepares the model for the special use and to the special - most appropriate - solution, and integrates the deep model with the normal model. The normal model is typically pruned in order to become simpler (Solomonoff and Occam principled) more deviation- or error-prone. The (normal) model might be enhanced by concepts and thus become a conceptual model.

(V) The Delivery and Product Layer

The final result of the modelling process is a model suite that is adequate for origins, properly justified, and sufficient. We cannot expect that one singleton model is the best instrument for all members of the community of practice. A sophisticated model that integrates deep and specific normal models is delivered to some members. An informative model that is derived from this model can be better for other CoP members. Models delivered in the finalisation space are often enhanced by additional annotations, e.g. relating the model to the demands for members of the CoP by answering the ‘with’, ‘by which’, ‘by whom’, ‘to whom’, ‘whichever’, ‘what in’, and ‘what
Models are supported by data sources. We can now distinguish two views that can be integrated. The model functions thus as mediator [17]. The rendering claim that these two views can be integrated. The same time models may also render a theory. We envision that this integration can be based on layered model development. Therefore, model-backed reasoning is properly based on layered model development.

**4 Models as Mediating Instruments**

Model-backed reasoning is thus some kind of revisable reasoning depending on the stages of knowledge. Modelling as a process starting with suites of generic models and revisable refinement according to data on hand. It should support handling of uncertainties and incompleteness of any kind and must thus make use of an integrated data management. Therefore, model-backed reasoning is properly based on layered model development.

**4.1 Towards Models as Mediators between Theory Offers and Theories**

Models can be used to render the theory offer. At the same time models may also render a theory. We claim that these two views can be integrated. The model functions thus as mediator [17]. The rendering procedures are however different.

We envision that this integration can be based on the mappings in Figure 6. Models can be understood as being composed of model concepts that are supported by data sources. We can now distinguish $f_{c,q}$-mappings at the same level, $g_{c,m}$-mappings between sources and concept, $G_{c,t}$-mappings from concepts to supporting sources, and $h_{c,q}$-embedding mappings from concepts to theory offers, models, or theories. Quantitative concepts are indicators or general quantitative properties. Model concepts are already abstractions from those quantitative concepts. Theoretical concepts in Figure 3 are elements of a theory that is currently under development. The research task is the harmonisation of the two mappings $f_{c,q}$ and $f_{c,m}$. This harmonisation can be based on the mappings for supporting resources $f_{s,q}$ and $f_{s,m}$ if some commuting diagram properties are valid for model concepts and the model.

**Figure 5: The layered model development framework**

**Figure 6: Models as integrating and mediating instrument for data science**
a model has however also a feedback turn both on the
theory offer and the theory. The model is then at the
same time an instrument, a mediator, a companion, a
middle, and a medium [5]. The model thus becomes
an investigation instrument.

4.2 Evidence-Based Reasoning in Data Science

Let us finally discuss an obstacle of quanti-
tative research that results in some obsti-
nacy of models. Theory and model develop-
ment are in both cases evidence-based due to
the way how they are derived from proxies. The O(bservation)-C(aims/Hypotheses)-E(vidence)-
R(esigning) pattern [7] starts with some observa-
tions and detection of hypotheses about these observ-
ations. Hypotheses are transformed into claims and
research questions that form a research agenda. Ev-
idences are then either systematically elicited from
data, from previous investigations, or from the belief
and knowledge space. Reasoning should then connect
evidences to the claims. The results is some kind of
Bayesian formulas representing the claim with the ev-
dence. Evidence-based reasoning combines therefore
inductive and abductive reasoning. It is enhanced
by Occam’s razor approaches [19] that allow to fi-
nalise the model development. It can be combined
with Solomonoff induction [24] that enhances a re-
sult (1) by conduction of experiments that will test
the claims and (2) by provisionally accepting the
claim if the experiment confirms the claims. The re-
asoning schema follows the induction/abduction-
retrospection-observation_concepts-theory_offer
pattern. It can be combined with Epicurus’ principle
of keeping multiple explanations that allowing con-
sideration of several models and theories as long as
they are consistent with the observations.

OCER pattern are the basis for evidence-based proxy reasoning (e.g. in the CRC 1266 [1]) that fol-
low positive evidences. Evidence-based reasoning is
based on the following principles:

1. Models represent only acceptable possibilities
   (each model captures a distinct set of possibilities
to which the current description refers) which are
   consistent with the premises and the knowledge
gained so far what makes them intrinsically un-
certain because they mirror only some properties
they represent.

2. Models are proxy-driven (the structure of the
   model corresponds to the proxies it represents.
   They might also include abstractions such as
   negation.

3. Models represent only what has been observed
   and not what is false in contrast to fully explicit
   models (that represent too what is false).

4. The more proxies that are considered, and the
   richer those models are, the more accurate the

5 Conclusion

Models are one of the main instruments in science and
technology. They support reasoning in various forms,
e.g. by systematic revisable modelling based on data
and as an associated collection of models;

This paper develops an approach for development
of fully fledged models (a) with extrinsic parts similar
to usual (normal) models and (b) with intrinsic parts
which are typically hidden in the modelling approach,
in the background and context of the model, and in
the intentions behind the model. While making this
explicit, we are able to use a model as a problem de-
scription and to compute the solution of the problem
under consideration directly from the model. The
paper presents the first part of this solution. The
development of corresponding tools is the topic of a
forthcoming paper.

The presented layered approach should not be ap-
plied as a 1-2-3-4-5 waterfall sequence of activities. Rather, model development and model utilisation use
an evolutionary approach that returns to previous
steps whenever sufficiency characteristics of models
become problematic within the application domain.
The layers can however be considered as phases of de-
velopment. We notice that our layered approach also
supports model revision and model evolution. It can
also be used for model migration and model reengi-
neering. The layered approach seems to be combin-
able with modelling cultures, e.g. those that can be
observed for our first case study [14]. Figure 5 rep-
resents the ‘greenfield’ or glassbox development with
model development from scratch. A similar picture
can be given for ‘brownfield’ model development, i.e.
model redevelopment, revision, and migration. Black-
box model representation uses only the first and the
last layers.

The layered approach is based on a separation of
concern within an initialisation layer, within an in-
trinsic and this implicit setup layer, within an extrin-
sic and thus explicit source reflection layer, within
an operational customisation layer, and finally with a
model delivery layer.

We conjecture that a similar layered approach can
be developed for model utilisation. The layers will
be different but oriented on the usage of a model
as an instrument. It can also follow the solution
story initialise-prepare-investigate-do-deliver that is
used for layered model development.
The main result of the paper is a systematic approach that closes the gap between theory offers and theories in data science. The approach is based on the mediating function of models between theories and theory offers.

References


Abstract. Modelling is an essential part of information systems development. Models are used for communication between interest groups and inside development teams. Models are also used for transferring baseline artefacts between development phases. Models are mainly developed by humans, which represent certain cultures - national, enterprise, professional, team, project etc. Because of that we claim that models, as well as many other information systems related artefacts are culture dependent. The models are born in certain context and these must be also interpreted by taking the original context into account. In our earlier studies we have analysed the effect of culture in information systems development: culture related aspects in general level, in information search and interaction and in web information systems. These studies acts as a basement for this paper having focus in modelling. Because of that we shortly answer to the question “How cultures differ from each other”. This reviews and synthesisis generally accepted frameworks for cultural analysis. In addition we shortly open the results of our earlier studies. Because modelling is a human activity, as well as information systems are used by humans, we feel important to build a model of humans in information systems development an use context. The findings of culture analysis are transferred to modelling practices via our framework that defines model as an instrument transferring elements of its development context to the models - we discuss about the roles of normal models, deep models and modelling matrix. Finally we will concentrate in the problems of cross-cultural modelling using selected national cultures as an example.

Keywords. culture-dependence of modelling; deep model, modelling matrix; multi-cultural system development;

1. Cultural Differences

1.1. The Layered and Dimensions Approaches

G. Hofstede [12] defines culture adapting the layered structure of Maslov pyramide (Figure 1). He uses the term “mental program” to describe the characteristics of each layer. The lowest level - operating system - is common for all humans. The second layer - collective program - is learned and remains same in a collective group of people and indicates the culture. The “onion model” on the right side of the Figure 1 describes the main elements of the culture. Values are the core of the culture. Rituals are collective
activities that are essential in a culture and indicate the membership of the group. Heroes are highly prized examples in a culture and indicate positive values of it. Symbols are words, gestures and objects that are common for those share the culture. Practices are manifestations of all other elements of a culture.

R. Lewis [21,22] applies the pyramid model in his culture analysis (Figure 2). As seen in the Figure the culture layers between Finland and Germany are different. Both countries in Lewis’ classification (discussed later in this paper) are close to each other and belong to the group of linear-active and data-oriented cultures. In spite of that, the values and core beliefs - the core of the culture - are different.

In our paper [17] we have introduced three methods to be used in recognizing cultural differences cultural differences: the 6D model of Hofstede, Lewis “triangle model” and Hall’s high/low context culture model. In addition to these there are several other ones; most of these overlap with the former ones and do not provide additional value to the analysis. Figure 4 illustrates the classification principles of Hofstede’s model.

The model of Hofstede basis on the analysis of six cultural dimensions [11,12]:

- Power Distance (PDI): the extent to which power differences are accepted.
Figure 3. The Hofstede 6D model [11]

- Individualism / Collectivism (IDV): the extent to which a society emphasises the individual or the group.
- Masculinity / Femininity (MAS): refers to the general values in the society - hard/soft values.
- Uncertainty avoidance (UAI): refers to the extent that individuals in a culture are comfortable (or uncomfortable) with unstructured situations.
- Long-term / Short-term orientation (LTO): refers to the extent to which the delayed gratification of material, social, and emotional needs are accepted.
- Indulgence / Restraint (IVR): acceptance of enjoying life and having fun vs. controlling the life by strict social norms.

The country comparison tool provides access to the database collected by Hofstede during the decades. The data covers culture values of most countries in the world. The left side of Figure 4 indicates the similarity of Finland and Germany. Meaningful difference is in LTO and MAS values, slight difference in IDG. The German work to reach results, which are more long-range than the Finns (LTO). They appreciate material values more than Finns (MAS) and live in the atmosphere, which is more puritan than in Finland (IVR).

1.2. Interaction and Collaboration

The Lewis’ model [21,22] focuses in analyzing interaction and collaboration activities of people. The nationalities locate in the corners and sides of a triangle. The corners represent the basic stereotypes: linear-active, multi-active and reactive. Linear-active cultures are data oriented (decisions are based on facts and official sources) and can be described by terms cool, factual and decisive planners. Reactive cultures are “listeners” - they base their behavior in more rich base of information sources (oral information from social networks, family, friends, ...) than people in data oriented cultures do. In communication they are not active members of the interaction; listening and reacting is typical to them in dialogues. Terms courteous, amiable, accommodating, compromiser and good listener describe people in reactive cultures. They are also usually members of collective cultures in Hofstede’s classification. Multi-active cultures are dialogue oriented. Like

³https://www.hofstede-insights.com/product/compare-countries/
people in reactive cultures they use rich set of information sources, but especially prefer oral information. Multi-active characteristic means doing several things at once, being extrovert and being active member of dialogues. Terms warm, emotional, loquacious and impulsive describes them.

1.3. Context Dependence

The Hall’s [9] model divides cultures according to the importance of context recognition in communication. Context means the extent of “wordless” communication included in the messages. High importance of context in communication indicates the importance of the membership in a collective group - i.e. collectivistic group culture in Hofstede’s classification. In high context cultures, the meaning of the message relates to the context in which it is presented. The group members know a variety of details included in the message without explicit messaging. The low context cultures are opposite. These cultures prefer punctual and clear messaging. All information is clearly included in the message and need for knowing the context is minimal. Low importance of context indicates highly individualistic society in Hofstede’s classification. Finland, as well as all Scandinavian countries, and Germany belong to the category of low context cultures; Japan and Arab counties instead are typical high context cultures. Somewhere in the middle of the continuum are USA and England, in which it is also typical to use words and sentences with hidden meaning.

1.4. The Storyline of the Paper

In this paper we start with an investigation whether cultures have an impact on models, i.e. on model development and model utilisation. Models can, for instance, be used as a means for communication. A hypothesis could be that models are a stability kernel among different people. The opposite hypothesis states that models are culture dependent. Section 2 discusses the relationship between modelling and cultures. Section 3 introduces a general model notion and a separation between the normal model and the deep model. We illustrate the cultural dependence for different kinds of schemata. Section 4
investigates the cultural differences for modelling styles. It is shown that the models developed in different cultures might also be different although the application is the same. Culture thus determines how models are developed and how models are used.

2. Cultures Influence Human Modelling Activities

2.1. Modelling is Different Worldwide

While zapping through textbooks from different countries on database analysis, design, and development we observe that the same topics and the same application tasks result in rather different database schemata. So, what causes these differences? What styles are preferred where? Under which circumstances one model is better than the other? Which detailedness is the best? Shall we concentrate on typical structures only? How exceptional cases are handled?

We observe also different modelling pattern and styles beside language differences (ER-like, UML, ORM, NIAM, IDEF, etc.). Students who got first lectures in object-orientation develop completely different schemata than those who got introduced to functional or procedural paradigms. Some companies like Ploenzke or SAP have a completely different way of representing the same application.

Moreover, the same language paradigm is often modified and extended. For instance, there are more than 50 extensions of the entity-relationship approach. Most of them are actually incompatible. Many modelling languages exist in a large variety of dialects what makes knowledge transfer and communication difficult. One reason might be that the ways of thinking, of modelling, of controlling, of working, and of supporting are different in different communities and thus result in different environments and thus in different cultures. Another reason might be the insufficiency of a language. In this case, we can use language pluralism and develop model suites [29].

Modelling might also follow different paradigms and postulates. It might use different not combinable theories. Modelling is biased by the developers and their educational and professional background [33]. It is typically laden by concepts that are to be represented, by its community, by the context into which the model is set, and by the way of utilising a model. If we compare these factors influencing modelling with the culture notion then we realise that all these factors are culture-driven.

So, we may conclude that organisation, professional, educational, and finally national cultures influence the outlook, the content, the adequacy and dependability of a model. It is not only the behaviour of people that is governed by the cultures but also the development and utilisation of tools that is governed by the culture. Models are instruments that are used in utilisation scenarios. Communication is one of the main scenarios. Models are used similar to utterances in natural languages in this scenario.

2.2. Culture Sensitivity in Information Systems Development

We have handled the topics related to information systems (IS) development in multicultural context in several papers. The papers handle information systems development from different points of view. Culture related aspects affect in both the development and
the use of IS. The development work is made in multi-cultural distributed teams, in which it is important to recognize the dynamics of the team in decisions related to organizing the work, leading the team and managing the development project. Transfer towards cloud based ecosystems and web information systems (WIS) makes recognition of the end-user base more difficult. In requirements engineering phase we have more and more often “faceless” clients from different cultures and from different parts of the world that must be served by the WIS [15].

Our earlier studies cover general aspects in IS development [14], information and query-answer related aspects [17] and web information systems design related aspects including database design and conceptual modelling [16]. These papers provide a “handbook” type list of findings to support IS development in multi-cultural context. Our analysis applies the interpretations of human behavior using Hofstede’s dimensions and Lewis analysis. It also acts as an evidence to the applicability of culture analysis and stereotyping methods to guide IS development for multi-cultural context. The realization of the findings is included in the requirements engineering phase, which transfers them to non-functional requirements in the requirements specification of the IS.

We have approached the topic via Hofstede’s and Lewis’ models. Hall’s model provides some new aspects to the analysis, which are worth of more studies. Low context cultures are tended to demand exact communication. Our hypothesis is that IS development in such cultures indicate clear and unambiguous user interface, whereas high context cultures are tended to accept some ambiguities and complexity in it. Low context indicates linearity, high context multi-activity.

2.3. Information Systems Modelling and Culture

In IS projects models are means for communication - transferring duties and work items trough the life cycle of the IS and supporting interaction between the interest groups. We defined modelling to be a kind of solution to the problems of communication. Modelling languages are culture independent unlike natural languages. However, our hypothesis is that the use of them and the structure of the models indicates culture of its user. In IS development models transfer system related knowledge between interest groups. Because most of the modelling techniques used in practical work are semi-formal, the lacking exactness opens door for misunderstandings. In addition the sender’s and receiver’s ability to interpret the model may vary; one of the reasons is culture. Interpretation of the models is also context sensitive (i.e. in different contexts the interpretation may vary). The model itself is a construction of concepts and individuals according to their internal concept handling mechanism interpret it. In our paper [13] we introduced a hypothesis that also this mechanism is culture dependent - that what a Finn finds in a (conceptual) model would be different to the findings of a German or Japanese. In the same paper we have listed problems related to communication and collaboration in multi-cultural context: (1) behavioral patterns of people are different, (2) concept creation and handling is different, (3) language of communication is different, (4) communication includes opportunity to serious misunderstandings and (5) transformations (transferring the message from one language to other) may change the meaning of the message. All these problems fit to IS modelling, too.
2.4. Modelling of a Human Being, Team Dynamics and Organization Culture

In culture adaptable information systems development context there have been efforts to model its user. In adaptable IS the system includes a model of the user. If this “user model” is equal to the real behavior of the user, the system may adapt its operations according to the expectations of different users. In culture adaptable IS this model includes culture related factors.

One of the best-known model is MOCCA environment developed by K. Reinecke [27]. MOCCA is an application that can adapt ten different aspects of its UI with 39366 combination possibilities altogether. MOCCA acts also as an example of the technical implementation of the flexible user interface in information systems design. The user model takes into account the cultural background of the user including user’s cultural adaptation because of the influence of foreign cultures. The user profile basis on the following parameter: MAS, UAI, PDI, LTO, IDV, year of birth, political orientation, social structure, religion, education level, familiarity to certain form of education, computer literacy and gender. External dependencies cover nationality of the person, his/her mother’s and father’s nationality and language skills (mother tongue, foreign languages). Dynamics of the model basis on the former length of stay under the effect of the foreign culture.

M. Phaedra and M. Permanand [26] have introduced a student model that takes into account his/her demographic factor values. The person (student) has simple arguments: identification, age and gender. The dimensions of the model fall into five categories that describe particular contextual categories: geographical aspects, religion, ethnic background, education level (including school - note the importance of school as a root of an important source of information in dialogue oriented and reactive cultures), and particular physical environment settings and terrains. External properties cover - as in MOCCA - parent data including their occupation (social group) and native language. The model does not include any aspects creating dynamics, if changes in the parameter values not counted. The model neither includes any cultural factors derived from stereotype models.

G. Dafoulas and L. Macaulay [6] have modelled the dynamics of multi-cultural virtual software development teams. The model lists a variety of factors that to take into account in management of the team and organizing the work in it. They emphasize that each individual is a member of multiple cultures (Cultural profile category): one or more national/ethnic cultures, one or more professional cultures, a functional culture, a corporate culture, and a team culture, among others combined to individual (personal) characteristics. They have seen, especially in multi-cultural distributed teamwork the importance of professional and functional culture: “software professionals worldwide belong the computing subculture, which is stronger than any other culture”. A Russian software engineer (professional culture) would be more similar to an American peer than to a Russian marketing manager (functional culture). The model of cultural dimensions in virtual software teams does not specify the properties of an individual (professional) but a roadmap to manage the team. Human resources category includes PDI, UAI, IDV, time difference between members, trust level between team members, concept of space (Lewis) and material power (goods that create or indicate power). The required skills interact with human resources. The required skills category covers communication skills, participation activity, leadership, conflict resolution, problem solving, decision-making, goal setting and motivation. Team development category covers the improvement of required skills by taking into account first the team profile (diversity level), the role profile.
preferences) and finally task profile (requirements); the improvement is a continuous iterative process. Although this model belongs more into the category of “management and leadership models” it points out important aspects that indicate personal properties to be included in the model of a software engineer.

Our paper [14] includes a simple user model structured as a mind map. This model indicates the important factors of an individual to be taken into account in developing adaptive information systems. The personal properties category of the model covers personality profile (nine general factors and three dialogue preference related factors), work profile (six factors) and education profile (three factors). The portfolio category includes task related parameters, user involvement description, type of collaboration and restrictions to take into account.

E.G. Blanchard et al. [5] use very similar approach in their conference paper related to intercultural communication. They have found a remarkable (literature based) evidence which shows that the way people interpret and react to their environment significantly differs from one culture to another and that wide range of human activities and situations influenced by culture. In spite of that, the human-related technologies have not accounted for culture. Western context dominates in design and solutions, which are tested and validated on Western samples. In their paper Blanchard et al. (2013) introduce a simplified conceptual model of intercultural communication. Cultural elements concept class in the model covers cognitive cultural elements and cultural non-verbal communication (body language) related aspects. Non-cultural (innate) elements concept class includes behavioral primitives (gestures, postures, facial expressions) and some innate non-verbal communication elements. Additional concept classes cover the role of context, culture and cultural group, enculturated individual aspects, cultural group cohesion and a variety of descriptors.

B.S. Parumasur [25] handles the problems related to organizational development (OD). The paper states that American and European consultants have developed most of the OD practices. Because of that, cultures collide in different cultures. Contextualized and customized approach is needed: The skills readiness acquired at school varies (abstract thinking, team skills, entrepreneurship, technical, language, ...), motivation factors vary between cultures (emergent/mature), gap to the welfare plays an important role. In all change and improvement processes gap between local values and proposed interventions must be recognized. However, the evolution of the political and economic climate changes the values rapidly; globalization leads to adoption of foreign influence (see [27]). The paper concludes to a model, which indicates organisation’s readiness to changes; the model applies Hofstede’s 6D model in the following way. PDI: High PDI indicates acceptance of social and economic gaps, acceptance of inequality, acceptance of centralization, valuation and respect to authorities and hierarchical relationships. In high PDI cultures close supervision is needed. PDI indicates also suitability of participative / non participative decision making. Large PDI is associated with collectivism and (lower national wealth), small to individualism (and greater national wealth). UAI: High UAI indicates resistance to changes. Combined with high PDI it reflects the responsibility of an organization instead of individuals. UAI links to formalization, the need for formal rules and specialization. IDV: IDV indicates the acceptance of person level benefits (salary differentiation based on productivity) and the importance of interpersonal relationships. High IDV leads to a need to explain every act in terms of self-interest. IDV relates also to the directness / informality of feedback (performance improvement vs. de-
stroying the harmony in certain conflict situations) and the aim to avoid face loosing (in a group. MAS: Masculinity relates to career advancement and salary vs. social aspects of work. It indicates also differentiation of gender roles. Relatively high MAS and Weak UAI justifies the high level of achievement motivation.

2.5. Revisiting Cultural Studies

The different approaches to dimensions of cultures have been combined, systematised, and generalised in [17]. The result is a 11-dimensional Kiviat graph in Figure 5. The dimensions could be used to derive guidelines for web information system development what has been illustrated by the dimension values for Finland, Germany, and Japan.

![Figure 5. The Kiviat graph of the cultural dimensions of people](image)

The graph can be extended by dimensions from other culture models. Instead we can use this combination also for derivation of other properties. The models by Hall, Hofstede, and Lewis cannot be solely considered. Additionally, combined properties cannot be derived. For instance, the triangle model [21] does not allow to reason on the cultural distance. The cultural distance is classically the differences of cultural values and is expressed as a function of differences in values of some of these dimensions, e.g. Euclidian or Mahalanobis distances. The triangle model also mixes three dimensions in Figure 5: the kind of being active, the kind of reacting on the partner and the way how tasks are performed. If we compare the distances between German and Japan people from one side and between Japan and European Russian people then first one is small in the triangle Lewis model whereas it is larger in Figure 6. The distances between Northern German and Japan people and between Japan and European Russian people in Figure 6 match far better with observations in normal life.
Figure 6. Three-parameter space of cultural dimensions with distances

So, the eleven dimensions in Figure 5 provide a better means for supporting also cross cultures. They can be easily extended by Victor’s model [23]. Since some of the dimensions are important for some aspects and not relevant for others, we can use views for these aspects. For instance, if we concentrate on the communication and interaction level then the Lewis triangle or the three-dimensional characterisation together with high and low contexts should be taken into account. Models are also developed for communication scenario. Therefore, we can abstract from Hofstede’s approach in this case. If we consider however the modelling activities then Hofstede’s dimensions become more central.

3. Models and Modelling

3.1. A Model is an Adequate and Dependable Instrument

Modelling is a topic that has already been in the center of research in computer engineering since its beginnings. It is an old subdiscipline of most natural sciences with a history of more than 2,500 years. It is often restricted to Mathematics and mathematical models what is however to much limiting the focus and the scope.

A model is a well-formed, adequate, and dependable instrument that represents origins [31,32]. Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilisation scenarios.

As an instrument or more specifically an artifact a model comes with its background, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form. The background is often implicit and hidden.

A well-formed instrument is adequate for a collection of origins if it is analogous to the origins to be represented according to some analogy criterion, it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and
it sufficiently satisfies its *purpose*. Well-formedness enables an instrument to be *justified* by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins. The instrument is *sufficient* by its *quality* characterisation for internal quality, external quality and quality in use or through quality characteristics such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions). A well-formed instrument is called *dependable* if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics. Models are used in various scenarios, e.g. *communication*, perception, system construction, analysis, forecasting, documentation, system modernisation and optimisation, control, management, and simulation. Let us in the sequel concentrate on the first scenario.

### 3.2. Database Modelling and Cultures

We already developed a number of *stereotypes for database schemata* [15]:

- (a) strictly hierarchical (ER-like) database schemata,
- (b) schemata with local viewpoints that reflect the needs of some stakeholders (local-as-view approach),
- (c) variants of XML-schemata, Bachman diagrams,
- (d) sets of local database schemata with the requirement that the corresponding database schemata is simply the union of the set (global-as-view based on local viewpoints),
- (e) sets of personalised views based on local database schemata with some kind of coherence constraint among all views (rigid global-as-view) etc.

These schema stereotypes can directly be associated with stereotypes as shown in table 1.
Table 1. Cultural stereotypes, kinds of database schemata that are potentially preferred, and potentially useful database schema stereotypes [15]

<table>
<thead>
<tr>
<th>Cultural stereotype</th>
<th>Preferences</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>High Power Distance</td>
<td>completely specified and well-formed, easy to understand and persistent database schema</td>
<td>(a)</td>
</tr>
<tr>
<td>Low Power Distance</td>
<td>freely configurable database schema that is adaptable to current needs and preferences</td>
<td>(d)</td>
</tr>
<tr>
<td>Individualism</td>
<td>my own database schema according to my and only my preferences (work profile, education profile, personality profile, security profile)</td>
<td>(e)</td>
</tr>
<tr>
<td>Collectivism</td>
<td>commonly agreed database schema reflecting all elements within a group according to the collaboration style</td>
<td>(b)</td>
</tr>
<tr>
<td>Masculinity</td>
<td>restriction to essential elements and only those, strict structuring schema with additional and optional elements, with exploration opportunities, personalised schemata</td>
<td>(a)</td>
</tr>
<tr>
<td>Femininity</td>
<td>complete schema with all elements, hierarchical structuring, more linear, wellScoped sub-schemata with simple reference to main schema extendible schema, flexible schema style, web-like schemata</td>
<td>(a), (d)</td>
</tr>
<tr>
<td>Uncertainty avoidance</td>
<td>all potential elements are reflected as well as all viewpoints, focused (oil stain) schemata handy schemata depending on current use and smooth integration of them, decomposable schemata</td>
<td>(a), (b)</td>
</tr>
<tr>
<td>Uncertainty tolerance</td>
<td>schema with a central part containing all necessary elements and further elements that might of use in future puritanical schemata without any non-essential elements</td>
<td>(e)</td>
</tr>
<tr>
<td>Long-term culture</td>
<td>schemata with step-wise exploration of all its aspects</td>
<td>(b)</td>
</tr>
<tr>
<td>Short term culture</td>
<td>different variants of the global schema for parallel integrated work</td>
<td>(d), (c)</td>
</tr>
<tr>
<td>Indulgence</td>
<td>completely fledged schemata with all details and views for later work</td>
<td>(d)</td>
</tr>
<tr>
<td>Restrained</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Linear-active culture</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Multi-active culture</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reactive culture</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

3.3. The Normal Model, the Deep Model, and the Modelling Matrix

Model development is typically based on an explicit and rather quick description of the ‘surface’ or normal model and on the mostly unconditional acceptance of a deep model [18]. The latter one directs the modelling process and the surface or normal model. Modelling itself is often understood as development and design of the normal model. The deep model is taken for granted and accepted for a number of normal models.

The deep model can be understood as the common basis for a number of models. It consists of the grounding for modelling (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities), the outer directives (context and community of practice), and basis (assumptions, general concept space, practices, language as carrier, thought community and thought style, methodology, pattern, routines, common-sense) of modelling. It uses a collection of undisputable elements of the background as grounding and additionally a disputable and adjustable basis which is commonly accepted in the given context by the community of practice. Education on modelling starts, for instance, directly with the deep model. In this case, the deep model has to be accepted and is thus hidden and latent.

This separation into normal model and deep model provides a means to distinguish two different logical theories behind: entailment or logical consequence for normal models and semantic presupposition for deep models. The pragmatic presupposition additionally consider the relation between a model developer or user and the appropriateness
of a model in a context. Inferences become then context- and scenario-dependent. Models are thus also evaluated based on their added value and not mainly evaluated based on their validity or correctness.

A (modelling) matrix is something within or from which something else originates, develops, or takes from. The matrix is assumed to be correct for normal models. It consists of the deep model and the modelling scenarios. The modelling agenda is derived from the modelling scenario and the utilization scenarios. The modelling scenario and the deep model serve as a part of the definitional frame within a model development process. They define also the capacity and potential of a model whenever it is utilized. Deep models and the modelling matrix also define some frame for adequacy and dependability. This frame is enhanced for specific normal models. It is then used.

3.4. Why Conceptual Modelling is (Not) Acceptable

A conceptual model is an adequate and dependable artifact or instrument that

- is enhanced by concepts from a concept(ion) space,
- is formulated in a language that allows well-structured formulations,
- is based on mental/perception/situation models with their embedded concept(ion)s, and
- is oriented on a matrix that is commonly accepted.

The conceptual model of an information system consists of a conceptual schema and of a collection of conceptual views that are associated (in most cases tightly by a mapping facility) to the conceptual schema [35]. Conceptual modelling is either the activity of developing a conceptual model or the systematic and coherent collection of approaches to model, to utilise models, etc.

Conceptual modelling is not in the center of development activities in all countries. Observing the history of the ER-conferences on conceptual modelling for three decades, we discover that it is still a central and attracting topic in Europe with a movement from North to South over three decades, did not change in Middle East, lost its attraction in Northern America and partially also Southern America, and has not been a central issue in the rest of Asia. So, one might ask why this attention and changes happened. One answer could be the loosing interest and importance in this approach. Another answer could be however that development is based on rather different styles in different countries, i.e. is culture-dependent. A third answer might be that models are latent and not explicitly stated what is also culture-dependent.

4. Cultures in Modelling

4.1. Models, Languages, and the Background

P.P. Chen [3] made the observation that the entity-relationship modelling language follows specific construction rules of the Old Egyptian and the Chinese language. This modelling language can only represent simple English sentences. Later, [10] could show that the extended ER modelling language HERM [28] covers the main categories in the English language. So, languages enable and hinder modelling.

---

5"All models are wrong. But some of them are useful." (often cited as a phrase by G.E.P. Box [2])
The background and especially the grounding are often incorporated into the deep model that is not explicitly communicated. For instance, the grounding for information system models includes DBMS and CE paradigms and postulates, set semantics, database theory, DBMS solution layering, DBMS technology (theory and culture), graphics and diagrammed canonical representation, ER canon, data-first-methods-second paradigm, database-approach-as-guide, etc. The basis of the model house in Figure 7 includes also a number of specific assumptions and commonly accepted practices. For instance, database modelling can be based on specific extended ER language, hidden basic types, views as derived (algebra) expressions, concept fields, extended ER thought style, parametric generic concept field, Indo-European utterance composition, extended ER development methods, extended ER heuristic rules, transformation techniques to other deep models, and extended ER tools. Additional assumptions are Salami slice tactics, the believe that functionality comes later, a rigid separation into firstness of syntax and secondness of semantics, visualisation, well-formedness (including lazy normalisation), extended ER pattern, reuse of experienced solutions (as exemplars), and flat two-dimensional schema representation. Global-as-design is commonly accepted in the database community. The global schema is the main result. Views are then defined on top of the schema by algebraic expressions in order to cope with user viewpoints. Global-as-design has its limitations. The combination with local-as-design, e.g. for BPMN diagram suites, becomes rather difficult.

The language as an essential part of the basis, the grounding and also the other choices in the basis are acceptable in one community of practice and might be completely unacceptable for others. So, the deep model and partially also the matrix are part of the cultural setting. It is often claimed that the organisation and education cultures rule this setting. The other dimensions of culture [14] are, however, not less important.

4.2. Models as a Sufficient and Necessary Means of Communication

Communication or exchange of data/knowledge/information is one of the main scenarios where models function as a content that is communicated. Models support learning, description, prescription, prognosis scenarios as well. Communication involves several partners with their own background and culture and is based on a relationship between these partners. Each of these partners also interprets the model in a specific way based on hidden background and the specific treatment of the four directives, i.e. presupposes a specific conditional framework against which the model makes sense. The explicit part of the model is the normal model. The implicit or pragmatic part is the deep model. The matrix of the model combines the deep model and the specific ways of model usage according to the considered scenarios. We shall see in the sequel that the pragmatic part is interwoven with the culture.

Models for communication must follow felicity resp. appropriateness conditions, i.e. conditions on well-formedness. Models and especially representation models must be developed on the principles of visual communication, of visual cognition and of visual design [15,24,30]. The culture of modelling is based on a clear and well-defined design, on visual features, on ordering, effect, and delivery, and on familiarity within a user community.

The meaning of models is typically combining four parts: (1) the literal model meaning (“what is said”), (2) the conveyed model developer meaning, (3) the model user meaning, and (4) and the implicated meaning (“what is implicated”). The implicated
meaning might be conventional or non-conventional. Non-conventionality of models includes what is the implicated content within the model and what has been left aside (non-conversationally). The first one can be general or particular. These different kinds of model content influence the model informativeness. The first part is triggered by the meaning of the model constructs and the model design as a statement. The second, third and fourth meanings are human related and thus depend on the culture of the people involved. The model itself should have a holistic interpretation.

Models in communication scenarios have to follow general principles and a set of rules called **maxims of model communication**. They are, in general, communication implicatures from [7,8].

**Cooperative principle:** Make your model such as is required, at the stage at which it occurs, by the accepted purpose of the model within the communication scenarios in which it is are deployed. This general principle has several sub-principles called “Maximes of Model Communication”

**Maxim of quantity:** Make the model as informative as is required for the current purpose of the model do not make the model more informative than is required.

**Maxim of quality:** Try to make model as valid as possible do not incorporate aspect that are invalid. All constructs need an adequate evidence.

**Maxim of relation/relevance:** The model and all its elements must be relevant.

**Maxim of manner:** The model should be parsimonious and perspicuous, i.e. economic and well-formed. Any obscurity of expression and ambiguity are avoided.

**Implicated maxim of efficiency:** The maxim of quantity requires that a model should be sufficient for an understanding by the model user (I(nformation)-principle [20]. From the other side it requires that the model should contain all necessary elements for an understanding by the model user (R(elevance)-principle). The model represents as much as the modeller can and must. The M(odality)-principle assumes that non-normal, non-stereotypical situations by the model that contrast to normal situations are given in an explicit and understandable form. The P(recision)-principle requires that a model is only at a precision level according to purposefulness. The B(revity)-principle prefers smaller models over longer, complex ones even though it has to be interpreted in a vague way. In some cases, vague models might serve better its function.

These maxims are explicitly stated by the sufficiency characteristics which allow to evaluate the quality in use, the external quality and the internal quality. Based on the modelling style we are able to reason on negation. A typical, however, often impractical approach with the strongest interpretation is the closed-world assumption in modelling that allows to conclude about the meaning of missing parts in the model. This assumption follows [4] ("Dire et ne pas dire"). The maxim of efficiency is often based on ‘hidden’ sub-models (called in the sequel ‘deep model’) which are taken for granted within a context and background by a community of practice.

We observe that these maxims are accepted in different cultures in a different way. So, the pragmatics of models depends on the culture. Moreover the deep model is governed by this pragmatics. The principles cannot be satisfied at the same time. Which principle is preferred also depends on the community of practice and thus on their culture imprinting.
4.3. Cross-Culture in Modelling

The adequacy of models has been handled in a strict or flexible way. Some model notions require a mapping property as a strict form of analogy. At the same time truncation or abstraction is required instead of focus. Also well-formedness is often taken more tolerant. Purposefulness is however commonly accepted. A similar observation can be made for dependability of models which is often only implicitly assumed. All model notions analysed in [34] use an implicit deep model that is undisputable. A rather surprising difference is the explicit statement on quality characteristics which have to be satisfied. At the first glance it seems that the list is random.

Let us, however, analyse the German database or information system books which are often used for teaching and papers and books from US where the first are published in the ER conferences since 1992. We observe that there are common properties applicable to both. There are also properties that can be only observed for one side. some properties are out of scope or out of style although they are important for information systems.

To make these different style more clear we shall use Lewis’ horizons of communication [21] in Figure 8. There are general properties that are commonly accepted by the two communities. There are also properties that are out of style or out of scope. There are also typical German and US properties that can only be observed for one of the communities. For instance, the US approach to development is often based on an 80:20 principle, i.e. the schema is left open for further development. The normal case is mainly considered. The opposite is observable for the German style. The schema must be complete. Whatever is not explicitly stated in the schema is not relevant in the application.

Therefore, cross-culture projects often result in a complete mismatch although the orientation to global-as-design and the deep model are commonly accepted. In order to come to a common solution, the principles to modelling must be agreed in advance. This agreement may start with an agreement of the maxims (of quantity, quality, relevance, manner) and on the R-, I-, M-, B-principles. According to [1], the choice of the language is influenced by effectiveness (cost-effectiveness, representation effectiveness), infrastructure (especially tools), resource availability, knowledge capitalisation, and - what she calls - political factors. The latter are cultural factors.

4.4. Deep Models are Governed by Culture

The deep model combines the unchangeable part of a model and is determined by the grounding for modelling (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities), the outer directives (context and community of practice), and the basis (assumptions, general concept space, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense) of modelling.

Let us consider information systems development: The grounding includes DBMS and Computer Engineering paradigms and postulates, set semantics, database theory,

---

6 These observations only cover partially the specific styles and should be extended with other material as well. Since we are interested in the general culture-dependence of modelling and not in a complete empirical study we restrict ourselves.

7 A similar observation has been made by M. Bjeković [1] for selection of enterprise modelling languages. She investigated the role of the purpose in modelling, the choice of modelling languages, and the factors for preferring one language above the other.
DBMS solution layering, DBMS technology (theory and culture), graphics and diagrammed canonical representation, ER canon, data-first-methods-second paradigm, and DBS guiding question. The basis can be build on specific an extended ER language, on hidden basic types, on views as derived (algebra) expressions, on concept fields, on a specific extended ER thought style, on parametric generic concept field, on Indo-European utterance composition, on extended ER development methods, on extended ER heuristic rules, on transformation techniques to other deep models, and on ER tools. Typical commonsense and common practices that are applied are: global-as-design, Salami slice, functionality comes later, rigid separation into firstness syntax and secondness semantics, visualisation, well-formedness (lazy normalisation), ER pattern, experienced solutions (as exemplars), and an orientation on flat schemata. Deep adequacy uses a specific analogy, specific focus, specific purpose. Deep dependability is based on arguments from the origins, on coherence inherited, on a rigid stability, and on sufficiency on the basis of extended ER quality criteria. The deep model is extended by the four directives: (i) Perception and situation models with lexicology and lexicography (e.g. an ontology as cut-out in the concept fields); (ii) a specific communication-oriented profile; (iii) the context typical for current IT or Business Informatics; (iv) the ER community of practice. The deep model provides the interpretation and the make of the normal model.

We realise that all components of the deep model are governed by the culture of the community of practice. This culture must be accepted and is the basis for a smooth communication within this community. The culture includes the acceptance of several principles: (1) the community uses a common vocabulary (*Helsinki principle*); (2) the ‘what’ and ‘why’ of modelling is agreed (*principled universe of discourse, environment*,
and information system); (3) an individual can have more than one viewpoint, one for each subject in which he is interested or has to deal with (searchlight principle); (4) all relevant general static and dynamic aspects, i.e., all rules, laws, etc., of the universe of discourse should be described in the conceptual schema (100 % principle); (5) a conceptual schema should only include conceptually relevant aspects, both static and dynamic, of the universe of discourse, thus excluding all aspects of (external or internal) data representation, physical data organization and access, as well as all aspects of particular external user representation such as message formats, data structures, etc. (conceptualisation principle); (6) the conceptual schema for an information system in practice can be perceived as being built up like some sort of onion the inner layer of the onion being formed by the minimal conceptual schema based on the fundamentals of logic, the extensions representing the layers of the onion (onion principle); (7) development is concentrated on the what about what with paying attention to the how with what we do it (e.g. conceptual level, external level, internal level) (x-level architecture principle).

4.5. Model Matrices are Driven by Culture

According to [19], a disciplinary matrix consists of (I) symbolic generalizations as formal or readily formalisable components or laws or law schemata, (II) beliefs in particular heuristic and ontological models or analogies supplying the group with preferred or permissible analogies and metaphors, (III) values shared by the community of practice as an integral part and supporting the choice between incompatible ways of practicing their discipline, and (IV) exemplars for concrete problem solutions similar to Polya’s theory for puzzle-solving (see also Wittgenstein ‘Game’ [36]). Additionally we consider (V) a guiding question as a principal concern or scientific interest that motivates the development of a theory, and (VI) techniques as the methods an developer uses to persuade the members of the community of practice to his point of view. So, the modelling matrix includes the deep model ((I),(II),(III)) which already culture-governed and additionally.

The modelling matrix is a specific disciplinary matrix and consists of the deep model and the modelling scenarios with specific stereotypes. So it governs the development of the ‘rest’ of the model development and model utilisation. The agenda is derived from the modelling scenario and the utilisation scenarios. The modelling matrix thus provides also a specific understanding of adequacy and dependability of models.

We may now derive specific modelling matrices for information system models — mainly for the development of the normal model. The matrix is assumed to be correct for normal models. Normal modelling involves showing how systems and their models can be fitted into the elements the matrix provides. Most of this work is detail-oriented. The matrix itself is thus driven by the culture accepted by the community of practice within the given context.

5. Conclusion

The aim of this paper was to analyse the culture sensitivity of modelling and models. We see modelling as a human activity and because of that it is as culture sensitive as human behavior in general. Worldwide we use same modelling techniques and tools, which for their part unify modelling practices and models. There are also several studies that criticize the use of tools and practices developed in powerful cultures in foreign
The kernel of the criticism is that these tools and techniques transfer the elements of the origin to the culture where these are used. Big gap can be seen between Western and Eastern cultures, as well as between mature (welfare) and emergent (more poor) cultures.

In our paper we have approached the topic from the direction of culture analysis in general level and applying the results of our findings in (cross-cultural) modelling context. A modelling framework — “the model house” — is used as a basic structure. The role of normal and deep models, as well as the role of modelling matrix are parts of this framework. Culture dependent aspects in conceptual modelling provide and comparison of modelling styles of two cultures are used as applications.

Our conclusion is that we found a lot of culture sensitive aspects in modelling. Models include a lot of “wordless” information that have source in modelling languages. In this context we want to make analogy to Hall’s high and low context cultures discussed in section 1 of this paper. Modelling languages — because of the semi-formal character — leave a lot of gaps to exact specification. This gap includes always some amount of culture related aspects. In addition normal information system requirements specification includes a lot of non-functional features that are defined by still less formal language, like natural language. These features are culture sensitive and also in most cases impossible to test by normal testing practices and verification; instead of tests human validation is used - again one culture sensitive step more.

References

Models and their Foundational Framework

Bernhard Thalheim
Christian-Albrechts University at Kiel, Department of Computer Science, D-24098 Kiel

Abstract

The term model is mainly used in two meanings which are considered to be different: a model of a problem domain as a conceptualisation; a model of a set of formulas as an interpretation in which every formula within this set is true. A general theory of models has not yet been developed. H. Stachowiak proposes a phenomenal approach and ‘defines’ models by their properties of mapping, truncation and pragmatics. Meanwhile, a notion of the model has been developed. At the same time, it seems that there are rather different understandings of model in sciences and especially Mathematical Logics. Sciences treat models as reflections of origins. Mathematical logics considers models as an instantiation in which a set of statements is valid. So, mathematical model theory is often considered to be a completely different approach to modelling. We realise however that mathematical model theory is only a specific kind of modelling. We show that the treatment of models in logics and in sciences can be embedded into a more general framework. So, the theory of models is based on a separation of concern or orientation.

1 Introduction

Modelling is a topic that has implicitly been in the center of research in science and engineering since its beginnings. It has been considered as a side issue for long time. During the last 40 years it has gained more attention and becomes nowadays a subdiscipline in many disciplines. The compendium [TN15] introduces models in agriculture, archeology, arts, biology, chemistry, computer science, economics, electrotechnics, environmental sciences, farming, geosciences, historical sciences, languages, mathematics, medicine, ocean sciences, pedagogical science, philosophy, physics, political sciences, sociology, and sports. The models used in these disciplines are instruments used in certain scenarios. So, essentially it is an old subdiscipline of most natural sciences with a history of more than 2,500 years [Mü16]. It is often restricted to Mathematics and mathematical models what is however to much limiting the focus and the scope.

The modelling method is a specific science method that uses models as instruments with certain intention or goal, e.g. for solving a task. The model represents or deputes origins. The model is used instead of the origin due to its properties, esp. adequacy and dependability. The modelling method thus consists (i) of the development of ‘good’ models, (ii) of the utilisation of the model according to the goal, (iii) of the compilation of the experience gained through model utilisation according to the goal, and finally (iv) of generalisation of the experience back to the origins. So, a model must be well-build for this goal, must be enhanced by methods that support its successful deployment, and must support to draw conclusions to the world of its origins.

1.1 A Model is an Adequate and Dependable Instrument

A model is a well-formed, adequate, and dependable instrument that represents origins [Tha14, Tha17a].

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilisation scenarios.

As an instrument or more specifically an artifact a model comes with its background, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form. The background is often implicit and hidden.

1 The earliest source of systematic model consideration we know is Heraklit with his λόγος (logos). Model development and model deployment is almost as old as the mankind, however.
A well-formed instrument is adequate for a collection of origins if it is analogous to the origins to be represented according to some analogy criterion, it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its purpose. Well-formedness enables an instrument to be justified by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins. The instrument is sufficient by its quality characterisation for internal quality, external quality and quality in use or through quality characteristics such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions). Model functions determine which justification is required and which sufficiency characteristics are important. A well-formed instrument is called dependable if it is sufficient and is justified for justification properties and sufficiency characteristics.

Figure 1: The model as an instrument that is adequate and dependable for its driving directives (origins, profile (functions, purposes, goals), community of practice, context) within its background (grounding, basis) and that properly functions in utilisation scenarios as a deputy of its origins

Figure 1 represents a model of the model. The development and utilisation methods form the enabling aspects of the modelling method. Driving directives are (1) origins to be represented by the model, (2) purposes or goals or functions of models, (3) the community of its users and developers, i.e. the community of practice, and (4) the context into which the model is embedded. Models function as instruments in application or utilisation scenario. Typical functions of models are (a) cognition, (b) explanation and demonstration, (c) indication, (d) variation and optimisation, (e) projection and construction, (f) control, (g) substitution, and (h) experimentation. A model is not built on its own. It has an undisputable grounding that has to be accepted. The basis of the model - similar to the cellar - can however be disputed. Grounding and basis form the background of a model. We observe that the background is often given only in an implicit form. The same kind of concealment can also be observed for the utilisation scenario which are implicitly given by sample and generalisable case studies for the utilisation frame.

The model is not simply an image of its origins. The mapping property [Kas03, Mah09, Mah15, Sta73] might be too restrictive for models. Instead, we use analogy. Models can also be material artifacts. A model can be a model of another model. Models might follow different structuring and behaviour than the origins. Usefulness and utility according to goals govern the selection of a model instead of quality characteristics such as validity. Finally, a model comes with its background. It cannot be properly understood and used if the background is concealed. Let us distinguish the concepts of goal, of purpose, and function in the sequel. The goal of a model is in general the association between a current state and the target state that is accepted by stakeholders or – more general – by members of a community. The purpose enhances the goal by means that allow to reach the target state, e.g. methods for model development and utilisation. The function extends the purpose by practices or – more systematically – by
scenarios in which the model is used. A typical scenario is the modelling method and its specific forms.

1.2 Models in Science and Daily Life versus Models in Mathematical Logics
Models in sciences and model theory in mathematical logic are often considered to be completely different issues [Bal16]. This point of view is correct as long there is no consolidated understanding of a notion of a model. Models in model theory are instantiations of a set formulas. This set of formulas is satisfied by a model according to a logical definition frame. The model is a structure that is defined with the same signature as the set of formulas.

So, we might come to the conclusion that there are at least three different understandings of the model. We will oppose this conclusion in the sequel. It is only true for the Fuzzy or phenomenalistic view.

Models in science typically follow the modelling methods. They may be composed of a number of models and be based on other models. A model must not be true. It should however be coherent to some extent within its discipline.

The origin in science is not limited to material origins. The origin itself can be virtual or be again a model, e.g. a mental model. So, the modelling methods may also be iteratively applied.

Models often used in daily life. One kind are metaphors or parables. The typical kind is, however, a pattern for explanation, negotiation, and communication. Models carry a meaning. It is often debated whether a fashion model or a diagram or a visualisation can be considered as a specific kind of a model.

The modelling method presented so far is associated with its origins. We might however also use models for construction of other origins or models. In this case, the model is not generalised but used as a blueprint for another artifact. So, we observe that the modelling method must be extended.

1.3 Models and their Utilisation Scenarios
Models are used in various scenarios, e.g. communication, system construction, perception, analysis, forecasting, documentation, system modernisation and optimisation, control, management, and simulation. Let us in the sequel concentrate on the first three scenarios.

The extended modelling method is embedded into a more general form of activities, i.e. scenarios. The model itself is used as an instrument in a scenario or a bundle of scenarios which we call usage spectrum. It has a function or a number of functions in these scenarios. This functioning must be effectively supported by utilisation methods and is used by members of a community of practice in most cases. For instance, models of situations/states/data are often used for structuring, description, prescription, hypothetic investigation, and analysis. So, we observe that the function (or simpler the purpose or the goal) of the model is determined by the concrete way how a model is used.

A model might be oriented towards this community of practice. It can however also represent the scenarios themselves. It might represent the context of these scenarios, e.g. the scientific or engineering background, the relation to time and space, the application area insight, and the knowledge accepted by the community. It might also be oriented to representation of either a situation and state under consideration or a evolutionary change process.

The different orientations is the basis to distinguish the six concerns for models: community of practice, background/knowledge/context, application scenario and stories of model utilisation with their specific frames, situation/state/data, dynamics/evolution/change/operations, and models as representations and instruments.

Figure 2 shows the relation between the concerns and the functions a model might have2.

1.4 The Storyline of the Paper
A general theory of models, of modelling activities and of systematic modelling has not yet been developed although modelling has already attracted a large body of knowledge and research3. The notion of the model is not yet commonly accepted. Instead we know a large variety of rather different notions. Model development activities have been a concern in engineering. The process of model development has not yet attracted a lot of research. Model deployment also needs a deeper investigation. The model is mainly used as an instrument in certain application scenarios and must thus function in these scenarios. So, a model is a medium.

2 Modified and revised from [Tha17c].

3 It is not our purpose to develop a bibliography of model research. Instead we refer to bibliographies in [TN15] and the more than 5.000 entries in R. Müller’s website, e.g. [Mü16].
We have already introduced the general notion of a model as a starting point. The next step could be the development of a general theory of modelling. It is often claimed that modelling is rather different in science and engineering. So, we might conclude that there is no general theory of modelling. This paper is going to show that there is a general theory of modelling. We start with a case study in Section 2. These lessons gained in this cases study are a starting point for a general theory of models, of modelling activities, and of systematic modelling. In Section 3 first elements of this theory are developed.

2 Models in Everyday Life and Sciences: A Case Study

Analysing model notions we realise that there are at least four different approaches:

1. The general phenomenalistic definition uses properties such as mapping, truncation and pragmatic properties for the association between origins and models. Most research on models starts with this approach.

2. The axiomatic definition follows frames used in Mathematical Logics and defines models as exemplifications of formal systems and formal theories. Models thus depute and represent a certain part of reality.

3. The mapping-based definition is based on a direct homomorphic mapping between origin and model. We might have another mapping between model and implemented system that is a realisation of the model.

4. The construction-oriented definition defines a model as being a result of a modelling process by some community of practice.

There is a fifth approach to models which simply uses artifacts as models without any definition, e.g. in human communication and also in sciences. The definition given above follows, however, the mathematical way of defining things through definitional extensions.

Models are used as (a) perception models reflecting someone’s understanding, (b) mental models that combine various perception models and that make use of cognitive structures and operations in common use, (c) domain-situation models representing a commonly accepted understanding of a state of affairs within some application.

---

4One of the prominent definition is given by John von Neumann [vN55]: “The sciences do not try to explain, they hardly even try to interpret, they mainly make models. By a model is meant a mathematical construct which, with the addition of certain verbal interpretations, describes observed phenomena. The justification of such a mathematical construct is solely and precisely that it is expected to work - that is correctly to describe phenomena from a reasonably wide area. Furthermore, it must satisfy certain esthetic criteria - that is, in relation to how much it describes, it must be rather simple. I think it is worthwhile insisting on these vague terms - for instance, on the use of the word rather. One cannot tell exactly how “simple” simple is. Some of these theories that we have adopted, some of the models with which we are very happy and of which we are proud would probably not impress someone exposed to them for the first time as being particularly simple.”

Studia Metodologiczne, 2019, Poznan
domain, (d) experimentation models that guide experimentation, (e) formal model based on some kind of formalism, (f) mathematical models that are expressed in some mathematical language and based on some mathematical methods, (g) conceptual models which combine models with some concept and conception space, (h) computational models that are based on some (semi-)algorithm, (i) informative models that used to inform potential users about origins, (j) inspiration models that provide an intuitive understanding of something, (k) physical models that use some physical instrument, (l) visualisation models that provide a visualisation, (m) representation models that represent things like other models, (n) diagrammatic models that are based on some diagram language with some kind of semantics, (o) exploration models for property discovery, (p) prototype models that represent a class of similar items, (q) mould models that are used for production of artefacts, (r) heuristic models that are based on some Fuzzy, probability, plausibility etc. relationship, etc. Although this categorisation provides an entry point for a discussion of model properties, the phenomenon of being a model can be properly investigated. Each category is rather broad and combines many different aspects at the same time. We already introduced a general notion of model. In this Section we will investigate whether the general definition covers all these kind of models for science and also daily life and whether it can be supported by a holistic treatment of models.

2.1 Models in Mathematical Logics

Let us consider only one kind of logics: classical Mathematical Logic based on first-order or higher-order predicate logics. Similar observations can be drawn for other mathematical logics as well. Mathematical logic has a long tradition of model research. Model theory became its branch and has a deep theoretical foundation. The main language is the first-order predicate logic. This language is applied in a rigid form [ST08] that became a canonical form of Mathematical Logics: It uses a canonical way of associating syntactic and semantic types. Additionally, the semantic type and the syntactic type have the same signature. The expressions of syntactic types are inductively constructed starting with some basic expressions of certain construct by application of expressions of some other construct. For instance, we may start with truth values and variables. Terms and formulas are then based on these basic expressions. The context is not considered. The world of potential structures is typically not restricted. The rigidity however allowed to gain a number of good properties. For this reason, first-order predicate logics became a first-class fundament for Computer Science.

In general, a model in Mathematical Logics is defined through its relationship to a set of formulas. These formulas are valid in the model. Additionally, axioms and rules of the first order predicate logics are valid in the model since they are valid in any structure of given signature. Models are thus instantiations (or exemplifications) for a set of statements. The theory of deduction is the main basis for reasoning. Therefore, the five concerns in Figure 2 have the specific peculiarity shown in Figure 3.

\[
\Sigma_{\text{Cop}} : \emptyset
\]

\[
\Sigma_{\text{context}} : \text{axioms, rules, formulas in a theory } \Sigma_{\text{knowledge}}
\]

\[
\Sigma_{\text{frame}} : \emptyset
\]

\[
\Sigma_{\text{situation}} : \text{statements } \Sigma_{\text{world}}
\]

\[
\Sigma_{\text{dynamics}} : \emptyset
\]

\[
\bigcup_i \Sigma_i \models \mathcal{M}_L
\]

Figure 3: Models in logics for investigation of situations and expressible properties: axioms and rules form the context world; admissible states are characterised by a set of formulas; models are instances of potential systems that obey the system
The special side of the approach of Mathematical Logics to modelling is the consideration of the set of all potential models together with a given instantiation. This approach is however also taken into consideration for other model kinds as we shall in the sequel.

A model might become then an exemplar or prototype for a given theory. It can represent this theory and thus allows to reason on the given theory. It can be thus a final or an initial model (see the theory of abstract data types [Rei84, Wec92]) where the first one is the best and most detailed representation of the given theory and allows to reason on all potential negative statements as well.

We notice that classically the community of practice is not considered. Also, dynamics is not an issue. There is not really defined any reasoning frame beside the calculus itself. We are free to choose Hilbert style or Gentzen style or any other derivation style for reasoning.

A specific decision within mathematical logics is the invariance of the signature, i.e. models as structures and logical languages for theory statements share the same signature. Therefore, there is a tight mapping between terms and formulas and the properties that can be stated on the model.

This specific mapping property has also been used for the phenomenal characterisation of models as structures that a based on a mapping from the origin to the model, e.g. [Bal82, Sta73, Ste66, Ste93]. We also observe that the truncation or abstraction property is a specific property of logical models.

2.2 Mathematical Models

Mathematical models are considered to be the most prominent kind of model. A mathematical representation of another ‘donor’ or origin model is based on the mathematical language. The mathematical model is used for solving of problems that have been formulated for the origin model. The association between the mathematical model and the origin model must be problem invariant. Solution faithfulness is often not given explicitly required, i.e. the solution obtained for the mathematical model must be faithful for the origin model. Mathematical modelling presumes the existence of this origin model. So, (1) it starts with an application analysis and a formulation of the problem to be considered in the application area. Next, (2) this formulation is transformed to the origin model which allows to describe the problem. (3) This origin model is then mapped to a mathematical model. (4) The fourth phase is the development of a solution of the problem within the mathematical model. (5) The solution is verified and will be validated for faithfulness within the origin model. Finally, (6) the solution is examined for its reflection in the application area. If the solution is not of the required quality then the phases are repeated. This 6-phase circular frame [GKBF13, Pol45] is a commonly accepted scenario for mathematical modelling.

![Figure 4: The mathematical model as a representation of a origin model within the mathematical frame](image)

We observe that the mathematical model is similar to the logical reasoning frame. Main quality requirements satisfaction of the problem solving purpose, adequacy of the mathematical model, robustness against minor changes,
and potential and capacity for problem solution. The community of practice should not influence the model properties. It may influence on the selection of various representation models. The situation and its dynamics determines the appropriatedness of the mathematical language. The mathematical model is determined by some mathematical method that has shown useful in the past.

Our model notion extends the model discussion by H. Hertz [Her84, vDGOG09]. He postulates that some artefact is a model due to its analogy to origins, its dependence within an application context, its purposefulness, its correctness, its simplicity, and its potentially only implicit given background. Models have thus a validity area.

Mathematical models are specific formal models. They are based on a formalisation that can be mapped to some mathematical language. The mapping from the formal model to the mathematical model should preserve the problem, i.e. it is invariant for the problem. The mapping should additionally also allow to associate the mathematical solution to the problem with a correct or better faithful solution in the formal model and for the origins, i.e. the model is solution-faithful [BT15]. The mathematical language has not only a capacity and potential. It also restricts and biases the solution space. The calculus used for the derivation of the model is any mathematical and not restricted to logical reasoning.

2.3 Science Models

All sciences widely use models. Typical main purposes are explanation, exploration, hypothesis and theory development, and learning. Models are mediators, explainers, shortcuts, etc. We can consider models as the third dimension of sciences [BFA+16, TD16, TTF16]5. Following [Gra07], sciences may combine empirical research that mainly describes natural phenomena, theory-oriented research that develops concept worlds, computational research that simulates complex phenomena and data exploration research that unifies theory, experiment, and simulation. Models are an essential instrument in all four kinds of research. Their function, however, is different as illustrated in Figure 5 [BFA+16].

<table>
<thead>
<tr>
<th>Empirical research:</th>
<th>Models for visualization, communication, and investigation of phenomena, experiments, observations, ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Theory-oriented research:</td>
<td>Models for representation, exploration, explanation, reasoning, understanding, comprehension, learning, ...</td>
</tr>
<tr>
<td>Computational research:</td>
<td>Models as mediator or starting point in inverse modeling, ...</td>
</tr>
<tr>
<td>Data exploration research:</td>
<td>Models as starting and intermediate point for hypotheses, investigation, pattern detection, ...</td>
</tr>
</tbody>
</table>

Figure 5: Some model functions according to the kind of scientific research

Empiric research also uses a canonical modelling mould. Beside an ad-hoc mould we might use a sophisticated one: (1) define a research question (based, for instance, on the rhetoric frame (who, what, when, where, why, in what way, by what means (Hermagaros of Temnos, also Augustine’s De Rhetorica) or W*H framework [DT15]), (2) consider threats to the research, (3) choose a research model (e.g. positivistic), (4) develop an approach how facts become theories, (5) create a generic meta-model (with some level of abstraction, with independent and dependent parameters and indicators), (6) define analysis approaches (qualitative or quantitative), (7) define the research program and agenda as a specific research process, (8) select the research method, (9) analyse the capacity and potential of quantitative data, (10) design the experiment, (11) design the case study, and (12) design the outcomes survey.

The empirical research approach often combines qualitative and quantitative approaches. The quantitative approach is often oriented on observable data whereas the qualitative approach orients towards theory, on concepts and conceptions, and on a characterisation of the situations of interest. The quantitative theories are often ‘phenotypical’ approaches contrary to the ‘genotypical’ approaches used in qualitative approaches. A typical approach is used in the collaborative research centre 1266 6. It uses additionally an investigative reasoning approach. Figure 6 shows the differences between genotypical and phenotypical models. We use a planar representation of the three dimensions: (1) the composition dimension with sources, concepts, and theories; (2) the kind dimension with qualitative and quantitative reasoning, and (3) the model dimension that allows to concentrate on certain aspects of the first dimensions depending which function, purpose and goal the model should satisfy. A typical specific

---

5The title of the book [CH04] has inspired this observation.

6Scales of Transformation – Human-Environmental Interaction in Prehistoric and Archaic Societies: https://www.sfb1266.uni-kiel.de/en

Studia Metodologiczne, 2019, Poznan
treatment of concepts is applied in modelling. Since models orient on certain aspects and represent also combined representations, concepts used in models are often not directly derived from concepts in the theory. Additionally, we should distinguish between quantitative, investigative, and qualitative models. The model kind in Figure 6 uses investigative reasoning and lends some elements from quantitative and qualitative theories beside the theory offering that are used for investigative reasoning. The quantitative theory should also be reflected in the qualitative theory.

Figure 6: Models for investigative and quantitative reasoning in empirical research

A qualitative theory uses a concept or conception space that represents situations of interest (may be based on some mapping \( g^e \)). The situation can be observed and characterised by sources (may be based on some \( f^e \) mapping). Empirical research in sciences often differentiates between an investigative reasoning and quantitative reasoning. Both use phenotypical observations on proxies. Quantitative approaches aggregate and combine the source data and thus allow to reason on correlation, dependencies, time and spatial relationships. The first two reasoning approaches should be based on a commuting diagram, i.e. we assume 
\[ F^e_c(g^e_c(f^e(situation))) = g^e_c(situation) \]
for any situation considered.

Evidence-based proxy modelling and reasoning treats models in a different way.

(a) Models represent only acceptable possibilities. Each model captures a distinct set of possibilities to which the current description refers) which are consistent with the premises and the knowledge gained so far what makes them intrinsically uncertain because they mirror only some properties they represent.

(b) Models are proxy-driven. The structure of the model corresponds to the proxies it represents.

(c) Models represent only what has been observed and not what is false in each possibility in contrast to fully explicit models (also representing what is false).

(d) The more proxies that are considered, and the richer those models are, the more accurate the world view is.

(e) Additionally, we use pragmatic reasoning schemata, e.g. \( A \) causes \( B \); \( B \) prevents \( C \); therefore, \( A \) prevents \( C \). The model themselves illustrate then concepts. Therefore, sources support concepts and conceptions what inverts the mapping \( G^e_c \) instead of \( g^e_c \).

Let us now consider the theory-oriented research. The frame for empirical research is similar to communication frames in Subsection 2.5. We neglect inverse modelling [Men89] although it is an important approach to science and it has been reconsidered and generalised under various other names, e.g. [ASG13, Noa09, SV05, BST06, TT13]. Data science approaches have been considered in [KT17].

So, we arrive with the hexagon in Figure 7. Models function as instruments within the science. They are vehicles for investigation, for analysis, for discovery of alternatives, for prognosis, for exploration, for explanation, for intellectual absorption, for learning, for understanding, for scoped and focussed comprehension, for representation of certain aspects, for discussion with partners within their background, for quick illustration, etc. They are supported by various kinds of reasoning. It seems that this variety is rather broad. If we however orient our investigation
on the scenarios then we discover that the model utilisation scenarios determine the function of the model. At the same time, the background with the grounding and basis strikes through. Models are biased by their foundations, by their development and utilisation methods, their communities of interest, and their context. A specific context is the school of thought [Bab03, Fle11]. The concept space determines what could the content and the scope of a model. The MMM compendium [TN15] illustrates that models, the approach for to model, and modelling share a good number of common approaches.

2.4 Conceptual Models

Conceptual models are widely used in Computer Science and more specifically in Computer Engineering. In Computer Science and Computer Engineering, one main scenario is (1) the model-based construction of systems beside (2) the explanation and exploration of an application, (3) description of structure and behaviour of systems, and the (4) prognosis of system properties. Model-based construction might include conceptualisation. The application scenarios mainly follows the description-prescription frame. The model is used as a description of its origin and as a prescription of the system to be constructed. The notion of conceptual model is not commonly agreed however7. In a nutshell, a conceptual model is an language-determined enhancement of a model by concepts from a concept(ion) space.

The conceptual modelling method uses a canonical style of model development and utilisation. Models are instruments in perception and utilisation scenarios. They function is explicitly defined, e.g. models for design and synthesis. The scenario can incorporate a decision point that stops after understanding the perception and domain-situation models or that designs and synthesises the conceptual model after a preparation phase. The last stage support then evaluation and acceptance of the model.

So, Figure 8 displays the more specific way of conceptual modelling for information systems. The IS community with its actors \{a\} shares an IT orientation. It might however be in conflict with the business users. They reason in a different way and are often using a local-as-viewpoint approach. The global-as-design approach might not provide an appropriate support. The model development and utilisation becomes canonical after the choice of the enabling language and the modelling method. The origin models such as the perception and domain-situation models follow the style accepted in these communities. The global-as-design approach must then provide appropriate aggregations and derivations for support of local viewpoints. The community also shares the assumption of strict

---

7We know almost threescore different notions what shows the wider controversy about this notion[Tha18a]. E.g., Wikiquote (see [Wik17]) lists almost 40 notions. Faceted search for the term “conceptual model” in DBLP results in more than 5,000 hits for titles in papers (normal DBLP search also above 3,400 titles)
separation of specification into syntax and semantics with the firstness paradigm [KL13, Pei98] for structures and the secondness [Cas55] of functions and views. The model to be developed inherits all the paradigms, assumptions, biases, conceptualisations, cultures, background theories, etc.

A typical example for conceptual modelling is entity-relationship modelling. [Tha18b] observed a large number of paradigms, postulates, specific modelling cultures, commonsense, practices, and assumptions such as global-as-design (with derivation of local viewpoints), Salami slice typing (for homogenisation of object structure within a class), set semantics (instead of multi-set semantics that is used for implementation), uniqueness of names within a schema, hidden implementation assumptions, specific styles for model composition one must follow, well-formedness conditions, etc. Some approaches add also requirements such as strict binarisation of all relationship types.

The notion of conceptualisation, conceptual models, and concepts are far older than considered in Computer Science. The earliest contribution to models and their conceptualisations we are aware of is pre-socratic philosophy [Leb14].

2.5 Models for Communication and Human Interaction

Human communication heavily uses models. They are often not called models. Some models might be metaphors or prototypes. Other models might be incomplete or not really coherent or consistent. They are however used for exchange of opinions among users. Models function in communication scenario as a medium. The communication itself determines the role and thus the function and therefore the purpose of the model. Models represent in this case a common understanding of the communication partners. They are biased by these partners. Communication is based on some common understanding about the topic that is under consideration. Partner have already agreed on some background. They use this agreement within their communication. This agreement is based a common reflection and some common model. This model is taken for granted and not further discussed in communication. So, partners agree on some background or deep model. Typically, deep models [KT17, Tha17b] are not explicitly communicated. We need however an understanding of a theory of deep model and return to it in the next Section. The model is used for a shared understanding, for sense making, for reflection, for derivation of open issues, and for negotiation.

The hexagon in Figure 9 shows the differences between models in Mathematical Logics or sciences and communication model. The main difference is the explicit community dependence of such models. Each of the partners or agents a has some understanding of the world. This understanding is the main ingredient of a personal model that we call below perception model. The perception model also reflects the setting of the agent, especially the orientation and the priming. The communication might also be based on some common understanding, i.e. on a situation model. The situation model represents the common world view, shared knowledge and beliefs, and shared opinion.
community of practice

methodology
application scenario
communication acts, speech acts, dialogue frame

purpose, function:
communication, reflection, understanding, sense making, negotiation

dynamics

representation

domain-situation model, state perception models, situation observation

state

communication acts, speech acts, dialogue frame

models for shared understanding, for communication, for negotiation, for open issues

context

common world view, personal knowledge, shared knowledge and opinion

purpose, function:
orientation on common understanding, opinion exchange, CoP issues, beliefs, desire

model theory:

\[ \Sigma_{\text{CoP}} : \text{community } \{ a \}, \text{group, social, behaviour} \]

\[ \Sigma_{\text{beliefs}} : \text{deep models, priming, orientation, perception models } M_{\mathcal{P}_b} \]

\[ \Sigma_{\text{frame}} : \text{dialogue frames } \hat{\mathcal{D}}_{\text{dialogue}} \]

\[ \Sigma_{\text{situation}} : \text{deep situation model } M_{\mathcal{S}} \]

\[ \Sigma_{\text{dynamics}} : \text{deep change models } M_{\mathcal{E}} \]

\[ \bigcup_i \Sigma_i \models M_{\mathcal{U}(\text{understanding})} \]

Figure 9: Models in human interaction: development of common understanding, exchange of opinion, communication, reflection, negotiation; context on the basis of commonalities in world views as deep models; scenario based on communication acts

The modelling methods is governed by communication and human interaction. So, we might base the frame on the dialogue and interaction frame. Models play a different role. They are used for common understanding. Typical specific models for human interaction are metaphors [Lak87].

Our second case shows the differences and also commonalities between Mathematical Logics and human interaction. The model must suffice all hidden agreements within the community of practice, the context, and the specific scope and focus taken by the agents. Therefore, the logics becomes now more advanced. Mathematical Logic as the opposite is oriented on general laws and thus not oriented on one model but rather on a family of models.

2.6 Lessons Learned with the Case Studies

We may now summarise the experience we gained:

- We realise by these case studies that there exists a common framework to models, to the activities of modelling and to modelling as a systematics reflection, for development of models, and for utilisation of models.
- Models are used to represent certain issues. They are more focused and must serve its purpose. The purpose and the focus determine which kind of adequacy is appropriate.
- Models do not exist on their own. They represent something in the world. The world under consideration depends again on the modelling frame. In most cases, mental models and perception or situation models are the origins which are reflected by the model.
- The justification must be given in a way that can be accepted by its community of practice. Models are developed by some members of this community and are utilised by some – may be other – members of this community of practice. So, models must be satisfying. Therefore, we need an explicit understanding of the sufficiency and thus quality of the given model.
- Models are composed of models that reflect their background and of models that represent specific states and situations within from one side and specific dynamics.
- Models are used as instruments in certain scenarios. They have a number of specific functions in these scenarios.
- Models are typically multi-models, i.e. an association of models which are reflecting specific sides of the same issue depending on the viewpoint that is actually considered. Since such models must be coherent we may bundle them within a model suites [DT10, Tha10].
• Model development and model utilisation typically follow canonical stories. An example is mathematical modelling that consists of a six-step procedure. Similar procedures can be observed for most sciences that start with a research question, initialise a certain research agenda or problem solving program or schedule, adapt elements to be used to this program, and then solve a problem. Solution-faithfulness is assumed as a hidden quality characteristic beyond the problem invariance. Modelling is typically based on some specific method or methodology, e.g. the mathematical method. These methods are a mould for the modelling process itself, e.g. pattern, template, stereotype, work-holding attachment, and an appliance. The method itself follows a macro-model.

• Modelling is still a big challenge to science and has a lot of lacunas. The biggest lacunas seems to be the missing support for combined model-based reasoning. Conceptual modelling uses a specific kind of layered model-based reasoning with changing reasoning methods depending on the stage of model development and model utilisation, e.g. in greenfield development of conceptual models: settlement of the context and the method, transfer of mentalistic concepts to codified ones with a concept expression language, transfer of domain-situation models to raw conceptual models, language-backed negotiation and agreement on a number of conceptual models that allow reflexion of different viewpoints, maturation of these conceptual models, and proper documentation. The reasoning method changes according to the stages. The integration of all these reasoning methods into a holistic one is not required.

3 Towards a General Theory of Models

3.1 Deep Models and the Modelling Matrix

The context and methodology layer determines the set-up of the model. It is often taken for granted and as given. It makes modelling more economical and also more reliable. A number of quality characteristics can be thus satisfied without any further consideration. Model development is typically based on an explicit and rather quick description of the 'surface' or normal model and on the mostly unconditional acceptance of this set-up. In reality, this setting becomes an essential part of the model. We call it deep model [Tha18b]. It directs the modelling process and the surface or normal model. Modelling itself is often understood as development and design of the normal model. This approach has the advantage that the deep model can be used as a basis for many models.

The set-up is the modelling matrix behind the model. It consists of the grounding for modelling (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities), the outer directives (context and community of practice), and basis (assumptions, general concept space, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense) of modelling. It uses a collection of undisputable elements of the background as grounding and additionally a disputable and adjustable basis which is commonly accepted in the given context by the community of practice.

The modelling matrix is often given as a stereotype one should follow while developing the normal model. Adequacy and dependability of is partially already defined by such stereotypes. The stereotype of a modelling process is based on a general modelling situation.

Stereotypes determine the model kind, the background and way of modelling activities. They persuade the activities of modelling.

3.2 The Five Concerns and a General Approach to Modelling

The case studies led us to the conclusion that there is a common three-layer setting in modelling:

(1) Community and scenario setting: The community governs the function that a model has to serve according to their issues and scenario.

Community of practice and application cases: The community of practice has its needs and desires. It faces a number of application cases. The application case consists of tasks that should be accomplished. These tasks form the community portfolio. The application cases can be solved by a model, i.e. the model functions as an instrument. Community members determine which model functions best. The community agrees on the issues for modelling.
(2) Guiding settings: The deep model and the matrix is commonly agreed according to the setting in the first layer.

Context: Modelling has its implicit and sometimes also its explicit context. Knowledge and disciplinary schools of thought and understanding are considered to be fixed. In a similar form, the background is fixed. This context forms the deep model that underpins the entire modelling process. A typical element of the deep model is the school of thought.

Modelling methodology and application mould: Modelling follows typically practices that are accepted within the community of practice. These practices are often stereotyped. The methods that are used for model development

(3) Origins and targets: Members of the community form their personal perception models and share their domain-situation model that characterises states and dynamics in the application domain that is of interest. These models are the origins on which the normal model is formed as an extension of the deep model.

The final result is a model that combines the normal and the deep models. The representation of the final model must not show all details of the deep model.

Figure 10: The five concerns for models as a kernel for a theory of models and of modelling

This general setting takes us back to the rhetorical frame and its generalisation to the W*H specification framework [DT15]: In our case, the model ("what") incorporates the meaning of parties (semantical space; "who") during a discourse ("when") within some application with some purpose ("why") based on some modelling language.

We thus distinguish between five grounding and driving perspectives to models:

Community perspective: The community has intentionally set-up its application cases, its interests, its desires and its portfolio. The community communicates, knows languages, explains, recognizes, accept the grounding behind the models, has been introduced to the basis and is common with it. Models are used by, developed by and for, and gain a surplus value for a community of practice. They may have a different shape, form, and value for community members. They must, however, be acceptable for its community. Typical specialisations of this concern are ‘by whom’, ‘to whom’, ‘whichever’, and ‘worthiness’.

Purpose, function, goal perspective: Models and model development serve a certain purpose in some utilisation scenarios. The model has to function in these scenarios and should thus be of certain quality. At the same time it is embedded into the context and is acceptable by a community of practice with its rules and understandings.

We answer ‘why’ and ‘for which reason’ questions.

*It relates back to Hermagoras of Temnos or Cicero more than 2000 years ago., i.e. they are characterised through “who says what, when, where, why, in what way, by what means” (Quis, quid, quando, ubi, cur, quern ad modum, quibus adminiculis).
**Product perspective:** Models are products that are requested, have been developed, are delivered according to the first perspective, are potentially applicable within the scenarios, and have their merits and problems. Typical purpose characteristics are answers to ‘how-to-use’, ‘why’, ‘whereto’, ‘when’, for which reason’ and ‘wherewith’ (carrier, e.g., language) questions.

**Engineering perspective:** Models are mastered within an engineering process based on some approaches to modelling activities and to utilisation of models. Modelling is a systematically performed process that uses methods, techniques, preparations, and experience already gained in former modelling processes. The modelling method is typically given in a canonical form. It guides and steers the model development and the model utilisation processes. This guidance can be derived from the scenarios in which the model functions.

**Background and context perspective:** Model development and utilisation is a systematic, well-founded process that allows one to reason on the capacity and potential of the model, to handle adequacy and dependability of models in a proper way, and the reason on the model and its origins that it represents. A modelling culture also answers the by-what-means question beside providing the background. The background is typically considered to be given and not explicitly explained. It consists of an undisputable grounding and of a disputable and adjustable basis. The context clarifies on which basis and especially on which grounding the model has been developed and must be restricted in its utilisation. Additional context characteristics are answers to questions about the ‘whereat’, ‘whereabout’, ‘whither’, and ‘when’.

### 3.3 Model-Based Reasoning

The observation depicted in Figure 6 drives us to a multi-model approach. We build models in situations, concepts and theories in dependence on their function and purpose. The same situation-concept-theory may be the basis for a variety of models. A typical multi-model approach is the consideration of models in Physics. Models should thus be considered to be the third dimension of science [BFA^16, TN15, TTF16]. Disciplines and also human communication, human interaction, and human collaboration have developed a different understanding of the notion of model, of the function of models in scientific research and communication. Models are often considered to be artifacts. In reality, they are however instruments that are used with a certain intention. Models might also be perception models that incorporate mentalistic concepts [Jac04]. Models are used in various utilisation scenarios such as construction of systems, verification, optimization, explanation, and documentation. In these scenarios they function as instruments and thus satisfy a number of properties.

Model-based reasoning [Bre10, Mag14] is enhances classical reasoning such as reasoning mathematical calculi or logical derivation. There are several kinds of reasoning that are more appropriate and widely used:

**Evidence-based modelling and reasoning** is one of the main approaches for quantitative models. Models only represent acceptable possibilities. Each model captures a distinct set of possibilities to which the current description refers. Possibilities are consistent with the premises and the knowledge gained so far what makes...
them intrinsically uncertain because they mirror only some properties they represent. In investigative and quantitative modelling, models can be proxy-driven where the the structure of the model corresponds to the proxies it represents. They might also include abstractions such as negation which must be then stratified. Propositional evidence-based reasoning is based on monotone functions and specific interpretations for logical connectives. Models represent in this case only what has been observed and not what is false in each possibility what is different from fully explicit models which represent what is false. The more proxies are considered the richer those models are, the more accurate the world view is. Evidence-based modelling and reasoning uses pragmatic reasoning schemata, e.g. A causes B; B prevents C; therefore, A prevents C. The calculus may use several implication forms, e.g. deterministic conclusions (A cause B to occur: given A then B occurs) and ordered sets of possibilities (A enables B to occur: given A then it is possible for B to occur).

**Hypothetical and investigative modelling** considers different assumptions in order to see what follows from them, i.e. reasons about alternative possible worlds (i.e. states of the world), regardless of their resemblance to the actual world. Potential assumptions with their possible world conclusions and assertions are supported by a number of hypotheses (allowing to derive them). It is often combined with abductive reasoning. Evidence against hypothesis is performed by testing its logical consequences, i.e. exploring different alternative solutions in parallel to determine which approach or series of steps best solves a particular problem.

**Causal reasoning and modelling** is a specific variant of inductive reasoning and justification-backed truth maintenance with assertions (beliefs, background) and justifications within some context (current beliefs, justifications, arguments). It establishes the presence of causal relationships among events based on methods of agreement, difference, concomitant variation, and residues. It uses assumptions and thus avoids inconsistent sets (‘nogood’ environment). The environment consists of a set of assumptions, premises, assumed statements, and derived statements for the world view. Justifications (e.g. data-supported) represent cause. Hypotheses are not derived from evidence but are added to evidence. They direct the search for evidence. They are tested by modus tollens \((H \rightarrow I) \land \neg I \Rightarrow \neg H\).

**Network reasoning** uses models that are expressed as networks. Nodes carry justification (arguments) and status (in, out, believed, relevant, necessary, ...). Edges, hyperedges, or directed edges have an antecedent (support nodes) and conclusions. They may also be non-monotonic and enable backtracking for dependencies (causality, chronological, space, etc. Labels also express the degree of consistency and believability. Queries can be expressed as subgraphs and are evaluated by query embedding into the network.

Model-based reasoning is an interactive and iterative process that helps to digest a theory and to develop the theory. Therefore, model-based reasoning integrates many reasoning approaches, e.g. deduction, induction, abduction, Solomonoff induction, non-monotonic reasoning, and restrospective reasoning. Model refinement might also be based on inverse modelling approaches. Facets of the last one are inductive learning, data mining, data analysis, generic modelling, universal applications, systematic modelling, and pattern-based reasoning.

### 3.4 Towards Powerful Methodological Moulds

The hexagon picture and the consideration of the variety of different (reasoning) techniques might lead to the impression that a general treatment of models and a methodological support is infeasible. Sciences and humans have however developed their specific approaches and overcome the challenges of this complexity. We will illustrate resolution of complexity by two methods: Layered treatment and generic modelling. Both approaches are based on the separation of a model into a deep or core model and a normal model. A typical example of a methodology is the mathematical modelling method [BT15, GKBFI3, vDGOG09, Pol45] (see Subsection 2.2). The CRISP cycle (data selection according to generic model, data preprocessing, data transformation, data mining, model development, interpretation/evaluation) [BBHK10] and classical investigation cycles (define issues and functions of the model, hypothetically predict model properties, experiment, (re)define model, apply and validate the model against the situation) are typical methodologies. Similar methodologies are known for data mining [Jan17], data analysis [BBHK10], and systematic mathematical problem solving [Pod01]. They use a variety of reasoning techniques and layer their application of these techniques according to the stage that is currently under consideration. These modelling methods and methodologies are used similar to moulds that are commonly used in manufacturing.
Data mining [Jan17], inverse modelling [RSS +10 ], and generic modelling [TTFZ14] start with a generic model. A set of associated models (called model suite) is the result of a modelling process. We may develop a singleton model or a model suite. Figure 12 displays a variant that starts with an initialisation and setting of the modelling process. The initialisation is based on the issues that are important for the community of practice, the tasks that are on the agenda, and the injection of the context. The community of practice aims at completion of tasks from its portfolio and is bound by profiles of their members what also includes beliefs and desires shared in this community. At the same time, the methodology for modelling is already chosen. That means, the upper dimensions in Figure 10 governs the entire modelling process. A similar approach can be declared for model redevelopment model evolution instead of model development from scratch (greenfield modelling). The result of the first layer is a deep model and a matrix.

Initialisation: Utilisation scenario, task profile, community of practice and profile, setting, context

- Problem setting rules, Conditions of modelling, Adequacy & dependability, Invariance and faithfulness
- Deep model & matrix determination & selection
  - Deep model & matrix setting
    - Select
    - Control

Background: Deep model & matrix

- Stereotyping modelling generalis
  - Select
  - Control
- Model (suite) rules, Priming & orientation, Policies, Constraints, Framing rules

Ansatz: Agenda & generic normal model (suite)

- Definition frame opportunities
- Select
- Control

Methods, Support enhancers, Monitoring enhancers, Evaluation methods

- Methods library
- Select
- Control
- Well-formed rules, Model (suite) refinement, Parameter handling rules, Support

Initial normal model (suite) with its support

- Indicators, Properties, Data space
- Select
- Control

Model background, Deep model candidates, Infrastructure, Environment

- Deep model & matrix determination & selection
- Select
- Control

Problem setting rules, Conditions of modelling, Adequacy & dependability, Invariance and faithfulness

Model (suite) rules, Priming & orientation, Policies, Constraints, Framing rules

Result: Specific model (suite) and various representation models

Figure 12: Layered model (suite) development (None-iterative form, greenfield variant)

The second layer or stage uses some kind of most general and refinable model as the initial model. A generic model [BST06, TF16] is a general model which can be used for the function within a given utilisation scenario and which is not optimally adapted to some specific origin collection. It is tailored in next steps to suit the particular purpose and function. It generally represents many origins under interest, provides means to establish adequacy and dependability of the model, and establishes focus and scope of the model. Modelling is often based on some experience. This experience can be systematically collected within a number of libraries. Libraries and collections are used for collecting the most appropriate setting and model. This selection is controlled or governed by rules, restrictions, conditions, and properties. The main results of the second layer are generic models and an agenda for the next modelling steps.

The third layer sets the environment for the development of the normal model. This environment prepares model development on the basis of the generic models and under inclusion of the deep model. The section of methods might also include the selection of parts and pieces from the context, e.g. from the background and especially from theories and knowledge. The fourth layer results then in the development of a normal model that can be neatly combined with the deep model. Representation models are developed for different members of the community of practice and for different functions the model must fulfill in the utilisation scenario.

This development process is often cut down to the fourth layer assuming the results of the first, second, and third
layer as already given. This kind of implicitness has often been assumed for language utterance. The government and binding approach [Cho82, BST06] made the two-step generation of sentences explicit: we intentionally prepare the deep model and then express ourselves by an explicit statement which is build similar to a combination of a normal model and of a cutout of the deep model.

4 Conclusion

A collection of modelling approaches has been presented in [TN15]. It seems that the variety of modelling approaches, the different utilisation of model, the broad span of underpinning theories, the variety of models themselves do not allow to develop a common setting for models. We often met the claim that models used in social and natural sciences, in mathematics, in logics and in daily life are so different that a common treatment cannot exist. From the first side, logicians provided a specific understanding of models that is easy and formally to handle. They inspired model research and the notion of model, e.g. [Bal16, Kas03, Mah09, Mah15, Sta73, Ste66, Ste93]. This notion has mainly been based on properties that a model should satisfy: mapping, truncation, and pragmatic properties as phenomenalistic characterisation of the notion. From the second side, models in all sciences have been used as an artifact for solution of problems, e.g. [BT15, Her84, vDGOG09, vN55]. The model notion has been enhanced by amplification, distortion, idealisation, carrier, added value, and purpose-preservation properties. From the third side, language- and concept-based foundations of models have been developed in philosophy of science and linguistics [Blfrm[ó]–5, Bur15, Cas55, KL13, Lat15, Pei98]. From the fourth side, models in engineering [BFA+16, LH15, TD16, TTF16] are instruments for system construction. From the sixth side, models are also instruments in human interaction. They are used as metaphors, for communication, for brief reference, for depiction, as prototype, etc. For instance, the question whether a picture or a photo is a model depends on their utilisation in some interaction scenarios. We thus may conclude that a common science and culture of modelling cannot exist.

The main claim in this paper is however that a common treatment of models in science and human interaction can be developed. We base our foundational framework on a separation of concern. This separation into five governors for models provides a common treatment of models and model utilisation. We base our framework on the observation that not all concerns are considered at the same time. So, we can use some kind of stepwise procedure for model development.

Utilisation of models as instruments in scenarios is the main driving property that distinguishes something from a model. The model functions in scenarios such as communication, reflection, understanding, negotiation, explanation, exploration, learning, introspection, theory development, documentation, illustration, analysis, construction, description, and prescription. How the model functions has been illustrated in the case of model-based reasoning. Model-based reasoning goes far beyond model methods used in classical first-order predicate logics or mathematics. We use the layering approach also for model methods since the development of a general reasoning method is far beyond the horizon.

The meta-models of modelling concerns in Figures 3, 4, 7, 8, 9, 10 support the layered modelling method in Figure 12. Instead, we could separate the layers into communities and their application scenario, into background and methodology setting, into situation and theory setting, into origin calibration, and model delivery layers.

This paper has been centred around models, theories, communities, context, methodologies, state, and dynamics at the same level of abstraction. Model-driven development and architecture [MMR+17, SV05] is an orthogonal approach to this paper. It distinguishes abstraction layers for models (M1), model frames (M2) [as meta-models], model frameworks (M3) [as meta-meta-models], and model framework setting (M4). The data/information and traces/events abstraction layer (M0) underpins models. Our approach has been mainly oriented on M1. We envision that the general M0-M1-M2-M3-M4 architecture can be integrated into our approach as well.

References


Abstract. Policy development is a complex and highly dimensional process. This complexity is very difficult to comprehend due to complexity of the parameter space, multi-dependence of parameters, and the nature of process. Therefore, policy makers should be supported while considering and evaluating various alternative decisions. This paper illustrates a modeling approach for advisory and assistance in decision making for political practitioners. We describe the corresponding advisory tool supporting the interactive decision process.

Keywords: Computer-based communication tool, interactive learning between scientific models and practitioners, political decision making support

1 Introduction

Policy decision making is a complex task which comprises the understanding of possible positive or negative consequences of decisions as well as a mechanism to restore consistency of a system in the case of inappropriate decisions. Thus, even policy experts often have only a vague understanding of how policies impact on relevant outcomes. Therefore, political practitioners use simple mental models (beliefs) to understand complex impacts of policies. For this reason, a technical solution for the simulation of policy impacts can be helpful, e.g. a graph displaying the impact of parameters. Our software will work as a digital playground system with relevant decision parameters as inputs and implied outcomes (consequences of the decision) as outputs.

Nowadays it is commonly accepted that good economic policy has to be evidence-based, i.e. rest on scientific knowledge and statistically proven evidence. However, scientific modeling is often criticized by political practitioners as a purely academic exercise that fails to provide practical tools for understanding or designing optimal real-life economic processes [5]. Accordingly, scholars promote participatory policy analysis that is characterized by an interaction between economic theory and political practice to combine the ‘objective’ knowledge derived from economic theories and empirical data with the ‘subjective’ knowledge of stakeholder organizations as political practitioners ([2], [9], [5]). Moreover, inadequate communication between scientific policy analysts and political actors is proposed to be a principal cause of the limited impact of research on policymaking. For example, the ‘utilization of knowledge school’ emphasizes the fact that policy analysts and policymakers live in two separate communities [5]. Hence, to become more efficient, the relationship between scientific experts and policy actors must be redefined.

Moreover, Stiglitz argues in his highly recognized book “Whither socialism?” [14] that the market-socialist experiences in Eastern Europe failed due to the incorrect beliefs of politicians in the Arrow-Debreu concept of real market economies as a complete set of competitive markets ([14], Chapter 11). Interestingly, Stiglitz’s explanation of the failure of the market socialism experiment highlights an interesting general point: economics must be recast as something more than a constrained maximization problem to understand and design real economies. In other words, theoretical models provide a relevant benchmark for understanding real-life economic processes but require abstract scientific models and political praxis to actually change the world. Hence, as previously discussed in [6], [12], [7], identifying effective solutions for central economic problems appears to be a problem of linking abstract economic theory with feasible political practice. Accordingly, scholars of participatory policy analysis discussed innovative tools, such as participative modeling (see [5]) (i.e., improving communication in formal models by means of interactive or man-machine simulations [for example, see [1]] or decision seminars [10]).

Beyond interesting methodological ideas and concepts for assessing the role of relevant ‘objective’ scientific knowledge it is important to better understand
and design the complex communication processes between science and political practitioners in a way that combines the knowledge of both worlds to generate advanced solutions to existing economic problems, such as the transformation to a sustainable bio-economy or reaching sustainable development goals.

In this context the paper develops a computer-based tool Policy-Lab that facilitates an interactive communication and learning between political practitioners and scientific models. Figure 1 shows a graphical presentation of positioning of scientific models’ statements (scientific world), political practitioners’ beliefs (stakeholder beliefs world) and the aspired communication between these two worlds.

![Graphical Presentation](image)

**Figure 1**

### 2 Policy-Lab tool

The Policy-Lab tool has to fulfill various tasks in order to effectively support the decision-making process and facilitate the learning of stakeholders. Those tasks can be categorized as follows:

- **Input Device**: Survey policy preferences, goals and beliefs using questionnaires.
- **Report Device**: Report surveyed data back to the group. This requires the dynamic application of statistical analysis of the data.
- **Interactive Modelling Device**: Users can simulate different policies and evaluate their impact on policy goals.
- **Consensus Device**: This device provides support in finding a potential political compromise.

An integral part of those devices is the simulation of scientific models. For example, typical formulas used in such simulation and graphical visualization.

The Policy-Lab tool will be implemented in the form of a web application. The tool is now in the creation phase, for this reason the main concepts of tool development, tool requirements, and its structure will be discussed further.

#### 2.1 Theoretical concepts

Some theoretical concepts will be explained before the structure of the playground is going to be introduced.

1) What is a model from the tool’s perspective?

In the sense of the current tool, a model is a computable unit with defined input parameters, computational core, and computed output parameters, which can be shown in a graphical form. A special sub-type of a model is a questionnaire, that has input parameters and a computational core, which adds user input to a statistical model and recalculates its output. The output of recalculation is not shown to the users directly, but can be called from another view.

2) How model data will look like?
The computational core of a model is predefined by the model scientists. It can be written in R, GAMS or in other programming language. The input and output parameters depending on the language used are language specific character values, which can be saved in a database or in an external file. These parameters should be accessible to the playground.

2.2 Playground system requirements

The creation of the simulation tool begins with the comprehension of required features. Partly this information can be derived from the existing Policy-Lab tool prototype, partly from model scientists’ requirements and user expectations.

The list of requirements for the simulation tool includes the following:

- clear and comprehensible software structure
- clear and comprehensible database structure
- scalability of the system
- maintainability of the system
- efficiency of the system
- run-time reciprocative input-output system
- user-friendliness of the system

Based on the analysis of system requirements the following issues can be defined during the development of the tool:

- How to implement interactive forms for user-input and output? Which interfaces are needed?
- How input and output parameters for the models look like and how they are saved?
- How the communication between the computational module and the web interface looks like?

2.3 Playground system structure

The simulation tool should serve as a web information system for model simulations, with interactive input-output mechanisms for users. The system should have a clear structured database, expandable for new entities, since the system will describe a varying amount of models. The system should visualize a list of models and its descriptions for users. Further the system should have views for input parameters from users and possibilities for the graphical presentation of computed output. Another integral part of the system is a computational module, where the computation of output takes place.

According to the system requirements the new system should have the following components:

- Web interface for users with possible use-cases’ definition, user management functions, presentation of views related to models, including model-input-parameters and output graphics.
- Computational module with possible integration of R and GAMS sub-modules.
- Communicational interface: beside other functions web interface and computational module should be capable of interaction with each other.
- Database for the web interface
- Database for the computational module

Web interface

Web interface is a unit that contains common login, logout, and register functions, explanatory use-cases, overview of present models, view for input parameters for the models, view for the output in graphical form. Moreover, there should be a separate view for administrators to allow user management.

Computational module

Computational module is a unit that can be connected to R or GAMS sub-modules or use some other language for computation. This module should communicate with the web interface: parse user-input-parameters, convert them to input-parameters in the format of computational language depending on the model, parse computed output back to the chosen web interface format (e.g. JSON).

Database for the web interface

Database for the web interface should contain all the information about users and their management, widgets shown in the interface, and shown model views. Furthermore, for the presentation of input and output this database should have information about input and output parameters of a model.

Diagram 1 shows a fragment of a possible ER-schema for the database:
includes descriptions of models, their simulations and different types of simulation result parameters. Additionally, every interface page has specific widgets of different types depending on model being simulated, including charts and questionnaires.

**Database for the computational module**

In the case computation is produced in another application it needs its separate database.

The database for the computation should have information about models, their computational cores, and their input-output parameters.

If the computation module does not need its own database, analogical database entities are necessary.

A possible ER-schema of a computational module is shown in Diagram 2:

![Diagram 2](image)

**Communication between web interface and computational module**

Communication between these two modules is an important part of the system, the whole software structure and efficiency depends on the form of communication.

Two architectural alternatives for modules’ communication have been developed:

1) Web interface and computational modules can be placed inside of one software project, so that the division in interface and computation is only a logical notion. In this case the interface and computational parameters can be saved in the same database. The computation itself can be made, for example, with JavaScript language. In the case of JavaScript, the computation will proceed efficiently as no integration of external R and GAMS modules is needed. The communication in this case is trivial and proceeds within one application.

2) In the other case, R and GAMS modules can be stored in a separate application, if the computation needs these modules because of its complexity, as it allows to bring a modular structure to the software. In addition, the exchange of or changes in R or GAMS models are made easier, because they do not influence the execution of the web interface in a negative way. Thus, the two components are not only logically, but also physically separated from each other. The communication between prototype tool and the application where model computation takes place proceeds with HTTP-messages, containing input-output parameters for computation and information about models in JSON format.

Figure 2 illustrates, how this kind of architectural style can be implemented:

![Figure 2](image)

In the system the both ways of communication will be used, depending on the complexity of a model.

### 2.4 Advantages of the system

The described playground system has a number of advantages:

- The system is scalable and extendable, as the underlying web information system is dynamic and is built accordingly to the database contents. The expandable database allows the insertion of new visual elements and models for the simulation.
- The first architectural style for communication allows the implementation of a run-time reciprocative input-output system.
- The second architectural style for communication contributes to system’s modularity and can be approached from two different perspectives: web interface based and computation based perspective. Thus, two scientists can work simultaneously on the two components. Any changes in one of the components would not cause error or stoppage of the execution in the other component. After the adaption of communicational modules, the changes can be accepted by both components.
- The tool supports expert, model and interactive learning, moreover the learning from collective decision is implementable.
- Description of use-cases supports user-friendliness.

### 3 Conclusion

Described Policy-Lab tool facilitates political decision making by presenting an interactive playground system, that simulates a large opportunity space for policy
decisions and computes possible effects of the model simulation with the decisions made.

As a result, Policy-Lab tool for policy decision enables political practitioners to relate potential policy decisions to corresponding outcomes.

The described tool should be flexible, efficient and user-friendly, in order to be able to simulate the full complexity of the models and to assist in successful decision making.

Related work

There exist other systems, which work with interactive user input-output and use a large number of possible input parameters and calculations, beside the Policy-Lab tool prototype, the precursor of the current simulation tool, mentioned above.

Examples of agricultural frameworks are:

Another decision making GIS-based tool is ReSAKSS [13], it contains data on agricultural, socio-economic and bio-physical areas. This tool assists policy makers in developing agricultural policies.

Examples of other frameworks are:
Today one can find modeling tools which accept a wide range of parameters and simulate some complex process in order to understand the influence of these parameters on the system in medicine.

The Lives Saved Tool for Maternal and Child Health (LiST) [15], [11] is a modeling framework developed by the Institute for International Programs at Johns Hopkins Bloomberg School of Public Health with intention to estimate the effect of health coverage on maternal and child health. LiST models the status of health coverage under the influence of various factors (e.g. increasing of health care services and usage of nutrition interventions). In this tool users can estimate the impact of different kinds of health interventions in order to plan the strategies for the improvement of medical methods in maternal, newborn, and child health. The tool contains the data about the effect of some kinds of interventions on peoples’ health. Further, the data about maternal and newborn mortality rates, health coverage and interventions of a particular country or region, is collected. Thus, a user can simulate the usage of specific health care methods in a particular region and see the influence of this usage as graphical output.

The Multi-Criteria Analysis Decision framework is a modeling framework for decision making and priority setting, which elaborates on possibilities to create „an equitable, efficient, and sustainable health care system“ [15]. All possible health interventions are ranked and compared during a multi-criterion analysis. A specific web-based framework to implement this approach was developed by the EVIDEM Collaboration [3]. The EVIDEM tool is used to provide the participants of the health care process with information and to support decision making during this process. The tool simulates different factors influencing patients’ health and produces a graphical output measuring the importance of these factors or the degree of their positive or negative impact.

References


https://www.nap.edu/read/13337/chapter/5
Models and Their Functions

Hannu JAAKKOLA \textsuperscript{a,1} and Bernhard THALHEIM \textsuperscript{b,2}

\textsuperscript{a} Tampere University, P.O.Box 300, FI-28101 Pori, Finland
\textsuperscript{b} Christian-Albrechts-University Kiel, Computer Science Institute, 24098 Kiel, Germany

Abstract. Models are one of the main vehicles in everyday and scientific communication, understanding, learning, explanation, exploration, comprehension, representation, starting points for investigation, pattern detection and exploration, system development, problem solution, hypothetical reasoning, and theory development. Models are mediators, explainers, shortcuts, etc. Models are used as instruments in these scenarios. Their function varies and thus their properties.

This paper investigates the functions of models in dependence on their scenarios. We concentrate the investigation on engineering and construction scenarios which are the main model use in Computer Science and Computer Engineering. The problem solving scenarios, the science scenarios, and the social scenarios are considered as well in a brief form.

Keywords. models as utility, functions of models, instruments, scenarios

1. Introduction

Models are widely used in life, technology and sciences. Their development is still a mastership of an artisan and not yet systematically guided and managed. The main advantage of model-based reasoning is based on two properties of models: they are focused on the issue under consideration and are thus far simpler than the application world and they are reliable instruments since both the problem and the solution to the problem can be expressed by means of the model due to its dependability. Models must be sufficiently comprehensive for the representation of the domain under consideration, efficient for the solution computation of problems, accurate at least within the scope, and must function within an application scenario.

1.1. Models in Software and Information Systems Engineering

Models in Information Systems (IS) development have three roles:

1. Acting as an abstraction of the real world;
2. Acting as a knowledge base;
3. Acting as a communication tool.

\textsuperscript{1}hannu.jaakkola@tuni.fi
\textsuperscript{2}thalheim@is.informatik.uni-kiel.de
The purpose of Information Systems (IS) is to support tasks of the real world (business) processes; IS modelling creates an abstraction of that (Figure 1). The modelling process itself creates an evolution chain of models from requirements to design and further to implementation and maintenance. This evolution chain can be seen as an IS related knowledge base transferring (communicating) IS related understanding (static and dynamic properties) between the development phases and among the development teams representing a variety of interest groups of the IS.

![Figure 1. The Role of Abstractions](modified by the authors)

In Figure 1 IS models (“system world”, lower part) are the abstractions of the real-world (business) processes. Abstractions are focused on the essential characteristics; because of that there are detail gaps that provide means for misunderstanding and interpretations. For a single set of characteristics we need several individual models describing the real-world from different points of view (a viewpoint represents a view to the system under development), in which a single model provides a single view to certain system characteristics. In Figure 1 these viewpoints are presented as overlapping ovals. Overlapping binds the different viewpoints to the whole and provide means for conformity and consistency checking. Such real-world properties that are not included in the IS are represented by the external connections or excluded (based on abstraction, not seen important and essential elements of the whole).

This simplified description of IS modelling rises up several questions:

1. What should be modelled and what can be left out?
2. How many views we have to take to the system?
3. How many individual models (modelling languages) we need?
4. When something is left out (which means that the model is not a complete 1-1 representation of the real world phenomenon) how the gaps are filled?

There is no clear right answer to these questions especially if we want to keep the focus in the key issues. However, usually the problems in Information Systems relate more to
the features that are not modelled than to those that are included in the models. Models make things visible.

Views are used in ordering the individual models. There are several approaches in this issue; one of the most referred is the Kruchten's 4+1 view model [26,47]. Kruchten specifies a scenario view as a central point for the other views; it represents the visible external behavior of the system in the form of use cases and other interaction models. The four other views serve different needs: A *logical view* represents end-user functionality and is necessary information for a variety of interest groups, a *development view* is targeted for the software developers and software management, a *physical view* covers aspects important for the system engineers transferring, and the *process view* to the variety of roles responsible for the final software implementation.

Individual models are implemented by *modelling languages*. Every view covers several viewpoints, which means that different *modelling tools (languages)* are needed. In practice most commonly used modelling languages are semi-formal ones having formal syntax and semi-formal semantics. Semi-formal modelling languages provide sufficient exactness combined with reasonable easy understandability. These languages are located in the middle area of the continuum having easy-to-understand (natural) at the one end and formal exactness combined with difficulty in understanding by non-professionals. There are hundreds of modelling languages for different purposes. The effort of OMG (Object Management Group) has continued the work initiated by Grady Booch, James Rumbaugh and Ivar Jacobson in 1990ies and standardized Unified Modelling Language (UML). UML has gradually become a commonly used set of modelling languages, which have also unified principles of IS development processes (e.g. Rational Unified Process - RUP). UML has 14 diagrams divided in two groups - behaviour and structure diagrams. See the details e.g. in [48].

The problem with the modelling gaps has two sides. On the one side is the exactness and on the other side are the problems caused by the non-modelled details of the reality. The more exact the model is compared to the real world phenomenon the more complex is the model and the more effort is needed to develop it. The non-modelled gaps cause misinterpretations and misunderstandings having final manifestation in system quality - unfortunately so. However, these can be avoided by well organized quality control activities and by keeping the usage related interest groups close to the developers (Agile development). One detail not discussed above is the role of non-functional (quality) properties, assumptions and limitations. Without going to the details, we state that they are changing along the development work to functionality, system architecture, a part of the development process, or stay as they are to be verified and validated in qualitative manner.

### 1.2. The Notion of Model

Let us first briefly repeat our approach to the notion of model:

*A model is a well-formed, adequate, and dependable instrument that represents origins and that functions in utilisation scenarios.* [8,39,43]

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice (CoP) within some context and correspond to the functions that a model fulfills in utilisation scenarios.
The model should be well-formed according to some well-formedness criterion. As an instrument or more specifically an artifact a model comes with its background, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background is often given only in an implicit form. The background is often implicit and hidden.

A well-formed instrument is adequate for a collection of origins if it is analogous to the origins to be represented according to some analogy criterion, it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its purpose.

Well-formedness enables an instrument to be justified by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins.

The instrument is sufficient by its quality characterisation for internal quality, external quality and quality in use or through quality characteristics [38] such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

A well-formed instrument is called dependable if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

1.3. The Storyline of the Paper

The approach to model functions and its explicit treatment is novel. The maturity of model development generalises SPICE approaches to software development. Section 2 clarifies what are model functions, what is the journey of a model in applications, and what kind of maturity is necessary. We observe that model utilisation can be categorized in four general dimensions. Dimensions are not orthogonal. A model may be used for problem solving and engineering at the same time. This paper orients on the engineering dimension of model deployment. Section 3 elaborates this dimension in detail. We elaborate the role of models and derive which function a model has to play in order to be properly used as an instrument in Section 4.

2. Functions of Models

The function of a model is often taken for granted and seems not to be an issue for investigation. Many modelling problems are caused by the unawareness of functions that models play, of the scenarios in which models are used as instruments, of the specific maturity that is necessary for an instrument, and of the journey of a model within these scenarios. Since models become somehow more universal after their development they are used in additional scenarios and in additional functions. This section now aims at a clarification of functions of models and their necessary level of maturity for dependency and adequacy of a model.

2.1. Models are Instruments

Models are used in some application areas in order to achieve something. They are, thus, aids in accomplishing tasks in those scenarios. They become then useful, usable and used task utensils which have their capacity and potential [2].
Models are, thus, instruments\(^3\) that are adapted to facilitate a definite kind or stage of operation in these scenarios, i.e. the model serves as an instrument or tool in a number of roles. In a scenario, the use of an instrument may vary, i.e. the model can be used in some variants of a play.

We observe in science that each science has developed its specific set of approaches. Mathematics, for instance, uses the ‘mathematical method’. This method is a specific mould, i.e. reveals clearly as having a certain character, forms the flow of processing and thus application of a model, and determines a distinctive nature, character, or type of the model to be used. Engineering and especially information system development have their molds that became common practice. A similar observation can be made for almost all sciences and problem solving tasks.

As tools instruments give practical effects to and ensure of actual fulfillment by concrete goals. They combine the necessary components for this fulfillment what rules the potential structure and the potential behaviour of models in these scenarios. They are not lacking or faulty in any particular according to the purpose.

The role as an instrument in an scenario dictates which quality characteristics are essential, i.e. in which case the model is sufficient and what are the evaluation and assessment approaches. Sufficiency implies (1) the soundness and the excellence of every model component, suggests (2) a completeness or perfection characteristics that can be sought or regained by a model, implies (3) perfection deriving from integrity, soundness, or completeness, and implies (4) retention of perfection of a model in its natural or original state.

2.2. What is a Function of a Model

The function of something is determined by a characterisation what it is used for. The function justifies a something’s existence. Functioning in a scenario means that models obtain a role which clarifies the actions and activities assigned to or required or expected of a model. The utility and usefulness of a model in some scenario defines the quality of being of practical use. Quality is characterised by essential and distinguishing characterisations of something. The capacity and the potential determines specified functions. These general characterisations provide now a means for consideration of a function of a model.

We distinguish the notion of goal, purpose, and function of a model similar to [9]. These notions are often considered as synonyms. The goal of a model is in general the association between a current state and the target state that is accepted by stakeholders or – more general – by members of a community. The purpose enhances the goal by means that allow to reach the target state, e.g. methods for model development and utilisation. The function extends the purpose by practices or – more systematically – by scenarios in which the model is used. A typical scenario is the modelling method and its specific forms. The purpose is characterised by actions for a model is specially fitted or used or for which a model exists. Actions might be complex and structured. We thus might consider any of a group of related actions contributing to a larger action. The goal is then something set up as an object or end to be attained. The function is performed as expected when applied.

\(^3\) An instrument is among others (1) a means whereby something is achieved, performed, or furthered; (2) one used by another as a means or aid or tool [32].
A model function provides a characterisation (1) as being sufficient; (2) being adequate (either in quality or quantity); (3) as satisfying, fulfilling, meeting the requirements or expectations within a given scenario; (4) as being fit; (4) as conform to, meeting, and fulfilling the wants or needs or condition or restriction; and (5) provide and supply what is desired or needed.

Function, purpose and goal are interrelated. The profile of a model combines the three properties. A function implies a definite end or purpose that the one in question serves or a particular kind of scenario it is intended to perform. A purpose suggests a settled determination. A goal suggests something attained only by prolonged effort and hardship. Other relation notions are intention, intent, design, aim, end, and objective. The objective or goal means what one intends to accomplish or attain. The intention implies little more than what one has in mind to do or bring about. The intent suggests clearer formulation or greater deliberateness. The design implies a more carefully calculated model. The aim adds to these implications of effort directed toward attaining or accomplishing. The end stresses the intended effect of model utilisation often in distinction or contrast to the action or means as such. The object may equal end but more often applies to a more individually determined wish or need within a community of practice. The objective implies something tangible and immediately attainable. In the sequel of the paper we will consider mainly the function of a model. A similar investigation can be performed for the other notions.

Models are used as instruments in certain utilisation scenarios such as communication, reflection, understanding, negotiation, explanation, exploration, learning, introspection, theory development, documentation, illustration, analysis, construction, description, and prescription. They have to fulfil a number of specific functions in these scenarios. Typical functions of models as instruments in scenarios are

(a) cognition,
(b) explanation and demonstration,
(c) indication,
(d) variation and optimisation,
(e) projection and construction,
(f) control,
(g) substitution, and
(h) experimentation [45].

2.3. Functions of Models in Scenarios

In general, a scenario an outline or synopsis of an application story where a sequence of steps is performed by some members of the community of practice. Models may function as instruments in these steps. We notice that each of the functions below require a specific form of model. For instance, typical functions in information system development and maintenance scenarios (e.g. those in [1,30,37]) are typically:

Communication and negotiation scenario: The model is used for exchange of meanings through a common understanding of notations, signs and symbols within an application area. It can also be used in a back-and-forth process in which interested
parties with different interests find a way to reconcile or compromise to come up with an agreement.

The model has several functions in this scenario: (personal/public/group) *recorder of settled or arranged issues, transmitter of information, dialogue service, and pre-binding.*

**Conceptualisation scenario:** Models may be used for conceptualisation of information system terms. Conceptualisation is typically shuffled with discovery of phenomena of interest, analysis of main constructs and focus on relevant aspects within the application area. The specification incorporates concepts injected from the application domain.

The function of a model within these scenario is *semantification or meaning association* by means of concepts or conceptions. The model becomes enhanced what allows to regard the meaning in the concept.

**Description scenario:** In a description scenario, the model provides a specification how the part of the reality that is of interest is perceived and in which way augmentations of current reality are targeted. The model says what the structure of an envisioned information system is and what it will be.

The function of models in these scenarios the *representation of essential properties and qualities* in an accurate or precise form, i.e. delineation.

**System construction scenario:** Models in system construction scenarios are model suites, i.e. requirement models, informative models, description models, prescription model, and code models.

The functions in this scenario are inherited from those scenarios for models in the model suite.

**Prescription scenario:** The model functions as a blueprint for or prescription of a information system application, especially for prescribing the structures and constraints in such applications.

Typical function of such blueprint models in these scenarios are: *instruction, direction,* and *guideline.* Often diagrams such as UML diagrams are used in an *inspiration* function. This might, however, too limited.

**Documentation scenario:** Models are used for providing various concepts that have been used for structuring and functionality development of a system. They specify what will be is the system, how the system can be used, by what means, in what way, what are supporting means, and wherewith facilities of the supporting software systems. They typically describe the structure, purpose, operation, restrictions, and other requirements in a documentation scenario.

Functions of models are similar to functions of manuals, i.e. *support for use, documentary validation, and presentation of documentary evidence.*

**Explanation and discovery scenario for applications:** In early stages of database development, the developer seeks an explanation and understanding of how, when, and when which entities are of interest and should be taken under consideration. Models serve then for presentation in a form that *makes the origin intelligible* (comprehensibility of relevant ideas or objectives and understanding in an application area) or *support for hypothetical reasoning.* The first function is typical for domain-situation models [41]. The second one for mental models, especially perception models.
Explanation and discovery scenario for systems: In later stages and reorganisation of a system application or ‘brownfield’ modernisation, the modeler rediscovers which constructs have been the basis for which part of the database, which associations occur among these constructs, which general forms are behind them, and the boundaries within which associations.

Within these scenarios model serve for information extraction, providing awareness, or making the origin intelligible.

Knowledge discovery and experience propagation scenario: Models tacitly integrate knowledge and culture of design, of well-forming and well-underpinning of such models and of experience gained so far, e.g. meta-artifacts, pattern and reference models. This experience and knowledge is continuously enhanced during development and after evaluation of constructs.

Models are functioning for elaboration, exploration, detection, and acquisition of tacit knowledge behind the origins which might be products, theories, or engineering activities. They allow to understand what is behind drawn curtain.

These scenarios are typically bundled into use spectra. Information system development is mainly based on description, conceptualisation, and construction scenarios. The re-engineering and system maintenance use spectrum is based on combination of documentation scenarios with an explanation and discovery scenario on one side and communication and negotiation scenario from the other side. Models are also used for documentation scenarios, explanation and discovery scenarios for applications or systems, and for knowledge experience scenario. We concentrate here on the four scenarios.

Furthermore, we cannot handle all functions for these four scenarios. The treatment and the properties of these functions can be exemplarily explained for one of them. Since the theory and techniques for informative models have already been developed in detail in [44], we can develop in detail the function that an informative model has to serve in Section 4. The function can be characterised by verbal expressions. Informative models are characteristic for the first phase of system development and for the documentation phase. Informative models are typically used as leaflet or instructions for use.

In general, models can be considered to be the third facet of science beside situation and theories. They are an essential element in problem solving and engineering. They are widely used in daily life without calling them models. We observe that model functions vary a lot. Models can be characterised by the ‘logos’\(^5\). The logos provides a separation of scenarios into perception/utilisation (see ‘word’), concordance/acceptance (see ‘judge’), intellectual absorption and comprehension (see ‘mind’), understanding and sense-making (see ‘power’), application (see ‘deed’), and reasoning-backed application (see ‘reason’) [40]. We are going to restructure the separation of concern in [40] by developing four main dimensions of model utilisation. Model development can be investigated in a similar form but is left out in this paper.

The four main scenario dimensions of model use in Figure 2 are:

---

\(^4\)The title of the book [3] has inspired this observation.

\(^5\)We refer to J.W. Goethe poem “Faust” where Faust reasons in the study room scene on the meaning of the word ‘logos’ λόγος. This word has at least 6 meanings where Faust used only four of them: word, concept, judgement, mind, power, deed, and reason. The first and sophisticated introduction of the logos can be traced by to pre-socratic philosophy and especially the work by Heraclitus [26].
Problem solving scenarios: Problem solving is a well investigated and well organised scenario (see, for instance, [2,11,13,31,46]). It is based on (1) a problem space that allows to specify some problem in an application in an invariant form and (2) a solution space that faithfully allows to back-propagate the solution to the application. We first describe a problem, then specify the requirements for its solutions, focus on a context, describe the community of practices and more specifically the skills needed for the collaborative solution of the problem, and scope on those origins that must be considered. Next we develop a model and use this model as an instrument in the problem solving process. This instrument provides a utility for the solution of the problem. The solution developed within the model setting is then used for derivation of a solution for the given problem in the origin setting. We may distinguish three specific scenarios:

Perception & utilisation: A problem becomes understood and can be investigated in a proper form.

Understanding & sense-making: A problem is elaborated in such a way that its specifics can be critically investigated and a solution can be derived.

Making your own: The problem is profoundly understood, properly formalised, and a number of faithful solutions can be derived.

Engineering scenarios: Models are widely used in engineering. They are also one of the main instruments in software and information systems development, especially for system construction scenario. Engineering is often technology-driven problem solving by practitioners where a problem got a different shape and a solution is some material product within a given infrastructure. Engineering also considers additionally robustness, failure management, safety, human factors, regulations,
practicality, and cost. So, the scenarios add to some perspectives that are used in science scenarios techniques, methods or processes used in production of any kind of goods, e.g. services (see, for instance, [6]). Engineering is a goal-governed process of designing and making tools beyond what is already available. Typical models in engineering are trial-and-error models, inspiration or enlightenment models, and product (consideration or documentation) models.

We may distinguish three specific scenarios depending on the level of sophistication:

**Application**: Handicraft, apprenticeship, and engineering is an art of creating and manufacturing a technics-based solution. It does not have to be scientifically grounded. Their main success criterion is that the solution suffices.

**Application management**: The art of solution development might be properly organised and this organisation allows to repeatedly produce a product for similar requirements and tasks.

**Well-understood technology**: Technology of solution development supports professional outcome-oriented (e.g. indicator-based) engineering and manufacturing including mastering the whole process of development of tools, processes, machines and equipment with an integrated view on facilities and systems.

**Science scenarios**: Sciences have developed a number the distinctive form in which a scenario is organised. Sciences make wide use of mathematical modelling. The methodology of often based on specific moulds that are commonly accepted in the disciplinary community of practice, e.g. [2]. Model development is based on four phases: description, formulation, ramification, and validation. In the description phase, individual perception and situation models involved into the modelling situation, are isolated and the corresponding primary properties are identified and represented, e.g. [12]. In the formulation phase, properties are interrelated, integrated and combined into a preliminary, initial model. This model is analysed in a ramification phase in order to check whether the model is a proper solution and to interpret and to consider its implications. Finally, the model and its capability and capacity are assessed in a validation phase.

We may distinguish three specific scenarios:

**Comprehension**: Most sciences and also empirical sciences use a systematisation scenario (see, for instance, [22,23]) where individual tasks are considered first, later combined, then considered together, next potentially based on concepts and theories, and finally based on theories.

**Computation and automatic detection**: Hypothetical reasoning and data-driven discovery scenarios are essentially search scenarios (see, for instance, [17]). Data science, knowledge discovery, deep learning, and business intelligence applications typically use this scenario.

**Intellectual adsorption**: Advanced mathematics and natural sciences use deep theories that have been developed and conceptualised over several centuries. Their scenarios are based on intertwined theoretical concept systems, on conceptualisation, and on meta-reasoning.

6Engineering is the art of building with completely different success criteria (see [33]: “Scientists look at things that are and ask ‘why’; engineers dream of things that never were and ask ‘why not’.” (Theodore von Karman))
Social scenarios: Social scenarios are less investigated although cognitive linguistics (e.g. [19,21]), visualisation approaches (e.g. [34]), and communication research (e.g. [10]) have contributed a lot, e.g. by the notion of mental models and perception models. Social models might be used for the development of an understanding of the environment, for agreement on behavioural and cultural pattern, for consensus development, and for social education.

We may distinguish three specific scenarios:

Acceptance: Models support orientation. People might believe, trust or credit what is given with the model. Schools of thought widely use models for agreeing on approaches, on common understanding and thought, and on some kind of exclusivity.

Internalisation & emotional organisation: Models may also affect behaviour, understanding, communication, and social life. They are appraised in a community and might be the basis for a system of values.

Concordance & judgement: Models are often used in discourses. They allow to collaborate to certain extend, to coact, and to integrate within a society.

In Section 3 we will now consider these specific models and their functions in engineering scenarios. In most cases, a model is in reality a model suite that consists of associated models.

2.4. Maturity of Models

A Maturity Model (MM) represents a path towards increasingly organized and systematic way of doing something. The maturity is defined by capabilities essential to fill the goals of the target system. The Capability Model is a modular description of the capabilities something has and needed to execute its tasks, described in the terms of the desired outcomes. Each component of the capability model has its maturity defined by its attributes. The maturity of the whole is the maturity profile of its components.

The most commonly used and the best known maturity models are CMMI and ISO 33004:2015 [4,15]. CMMI has its roots in late 1980ies, when Watts Humphrey joined to Software Engineering Institute (SEI) of Carnegie Mellon University and published his first version CMM [14] for improving the process quality of software developing organisations. In his maturity model Humhrey applied Quality Management Grid developed by Philip Crosby [5]. Since that CMM was further developed and applied in a big variety of versions for different branches of business. The current version - CMMI v2.0 (Capability Maturity Model Integration) is published by CMMI Institute is from 2018 [4]. ISO 33000 series of standards [15] has its roots in 1990ies, when International Standardization Organization started project called SPICE (Software Process Improvement and Capability dEtermination) as an international collaborative effort to develop a Standard for Software Process Assessment under the International Committee on Software Engineering standard, ISO/IEC JTC1/SC7/WG10. Official standards developed in this project were published as a series of standards ISO/IEC 15500 having the standard 15504 as a definition of their maturity model. The series number was changed to 33000 having the standard 33004:2015 [ISO/IEC 2015] including the maturity model definition.ISO standard included in two models - organizational level maturity model (continuous) and process based staged model; in the current version these are coincided.
The CMMI maturity model is based on five levels, which are (1) Initial (unpredictable, poorly controlled and reactive), (2) Managed (characterized for projects and is often reactive), (3) Defined (characterized for the whole organization and is proactive), (4) quantitatively managed (measured and controlled) and (5) Optimizing (focus on improvement).

The ISO model has six levels, which are 0. Incomplete, 1. Performed (process performance), 2. Managed (performance and work product management), 3. Established (definition and deployment), 4. Predictable (measurement and control) and 5. Optimized (innovation and optimization). The capability of processes is measured using nine process attributes specific according to the levels.

As seen, both models, CMMI and ISO/IEC, follow the same basic principles. We have adapted the six level model that is applies these principles to characteristics and capabilities in our maturity engineering model.

3. Models in Engineering and Computer Engineering

3.1. The Maturity Level of Model Utilisation in Engineering

The three main scenarios in Figure 3 we investigate are application, application management, and industrial development based on well-understood technology. Software engineering and information system development is currently mainly based on the first two scenarios. Models function then according to system construction. Recent development on component-oriented development, pattern, and product lines can be considered as early stages of the third scenario. Web information systems [35] used the third scenario in the most advanced way.

Application: Engineering is using different approaches to development than science and daily social life. The compilation of engineering knowledge (see, for instance, the six volumes [24]). The acting and deed scenarios are typical for all handicraft and apprenticeship processes. An artisan has developed a number of habits for production. Software engineering and information systems engineering are often limited to this handicraft approach. The level of maturity may be high or ad-hoc. The process of developing a product is somehow organised but not systematic. The quality cannot be properly guaranteed however since the management of the whole process is a prerequisite.

We may distinguish the 6 levels of maturity similar to the taxonomy levels for physical or handicraft work in [22,23]:

1. Be inspired, guided, imitate: There are already similar solutions that have been mastered in the past, e.g. code that has been developed for similar application problems. These solutions can be modified and used for the current problem.
2. Deliberately apply and manipulate: The current solution collection has led to some understanding of the issues that have been developed, e.g. the code for search algorithms the first volume of [24]. New solutions can be found knowing the properties of these collections.
3. Deliberately and precisely practise and manage: The problem area may be properly organised and the algorithms can be organised within flow of work, e.g. the systematic organisation of the data mining process [20].
Figure 3. The Kiviat graph for Engineering Scenarios and their Maturity

4. **Organise and reorganise course of action**: Engineering can be reorganised whenever it is necessary, e.g. agile SCRUM-based programming inherits the goodies of classical software engineering.

5. **Mechanise**: The engineering framework has found some systematic treatment that allows to mechanically derive the solution to problems, e.g. algorithmics with the 10 classes of algorithms

6. **Refine, reorganise, best practices**: The problem area is so well understood that solutions in this area can be developed based on best practices and inheriting good solutions, e.g. on the basis of reference models valid for the entire application, refinement, assessment, and getting the most of it.

These scenarios can be based on normal models [18,42]. The underpinning deep model which incorporates the disciplinary modelling matrix (paradigms, postulates, assumptions, ...) and the practised flow of action (mould) is inherited and taken for granted.

**Application management**: ISO/IEC 33004 and CMMI [4,15] have brought an understanding of the level of maturity for development processes. Engineering as well as software and information system development have developed approaches within such scenarios but did not yet reach the highest levels of maturity. We may distinguish the 6 levels of maturity:

1. **Fully describe**: Each step of the scenario and the corresponding utilisation of a model is well specified. The definitions are given and settled.
2. **Execute well**: The scenario can be executed and documented according to the description and the model can be used as envisioned.
3. Manage properly: The scenario can be managed in a form that allows to assess its level of completion, its lacking points, and its deficiencies.

4. Establish: The scenario has already practised in a number of applications and the experience gained can be used for new projects by adaptation to minor differences.

5. Understand the entire process: The scenario is well understood. A corresponding theory for reasoning about the scenario has been developed and ready for application.

6. Optimise the process: The scenario and the corresponding models can be organised according to a number of indicators and properties.

The deep model is partially known at least for the basis part of the background. This part and the normal modelling approach can be revised, reorganised and optimised to certain extent.

Well-understood technology: Software and information systems engineering is now developing approaches to become a technology of analysis, design, and development. Civil engineering and production management are the blueprint for such naturalisation. Industrial production uses tools, provides facilities for individualisation, and applies measures for integrated quality management. The scenarios and the models become ‘naturalised’, i.e. industrialised, manufacturable, adaptable to other application cases, more natural, and lifelike.

1. Existence of tacit tools and ready-to-apply methods: The scenarios and models are supported by tools and methods which can be used to generate them.

2. Matured activity in an engineering scenario: The technology has becoming a standard and is used for manufacturing.

3. Standardised application that use pattern and are based on genericity concepts: A number of general and easy to adapt approaches have been developed and allow instantiation, context enhancement and refinement on demand.

4. Ready for deployment general techniques: The manufacturing itself has led to machine tool design that can be brought in as off-the-shelf and can be adapted to the current circumstances.

5. Processes that provide facilities for adaptation and individualisation: The scenarios and the models can be individualised according to the very specific circumstances.

6. Processes that integrate quality management: Quality characteristics are incorporated into the entire process and can be properly maintained.

The technological understanding leads to properly manageable, adaptable and optimised processes that can easily adapted to new circumstance, e.g. to changes in the indicator system. The deep and the normal models are well understood and their effect can be predicted. These scenario are based on meta-models, metascenario, and meta reasoning.

3.2. The Engineering Dimension and the Model Notion

It seems that software and information system engineering is far from the maturity levels. We claim, however, that these maturity levels can be successfully accomplished if models are developed according to the model notion. Maturity can be characterised by
We define capability attributes directly by properties of adequacy and dependability. Capability attributes for models are based on questions we can ask for a model or a model suite. The most critical are the following ones:

- What is the function of the model in which scenario? What are consequential purposes and goals? What are anti-goals and anti-purposes?
- Which origins are going to be deputed/represented? Which are not considered?
- Does the model contain all typical, relevant and important features of the origins under consideration and only those?
- Rhetoric frame: who says what, when, where, why, in what way, by what means that can be extended to the W*H frame [7].
- Is the instrument adequate and dependable within the utilisation scenario? What are the parameters for adequacy and dependability?
- How purpose-invariance and solution-faithfulness is going to be defined?
- What kind of reasoning is supported? What not? What are the limitations? Which pitfalls should be avoided?
- Do you want to have a universal model that contains all and anything? Would it be better to use a model suite where each of the models depute/represent some specific aspects? What about the non-essential aspects?

4. Functions of Models Defined by Verbal Expressions

Now we face the description problem for functions of models. Models can be classified according to their instrument properties in scenarios. Let us consider only three specific functions: inform, reconstruct, and engineer. Moreover, these kind have also sub-kinds.

![Figure 4. Separation of Functions of Models](image-url)
We arrive at a separation of models in Figure 4. This kind of separation can be combined with the categories presented in [44,45].

These kinds lead to three different kinds of models with different capabilities, i.e. three different styles (or stereotypes) of adequacy and dependability. The properties of adequacy (analogous, focussed, purposeful) and dependability (justified (corroborated, coherent/conform, validated, stable/plastic) and sufficient (quality characteristics, evaluation)) are specific for each kind. The properties can be considered as parametric characterisation of the model capability.

4.1. Functions of Models Explicitly Given by Verb Fields

We still need, however, a means to characterise the functions of models. We propose to use word fields (or semantic fields of words [27,36]) for characterisation. A word field has typically several words which are related to each other by similar meanings or through a more abstract relation. The bundle of words bears the meaning. The words in a word field all relate to the same subject or concept. A word in a word field can be considered as more general one than another in the same word field. We may thus draw tree like the one in Figure 5. For instance, the word ‘believe’ is used in two semantic fields: (1) (verb form) accept as true; taken to be true with a number of synonyms (trust, buy, infer, bank, swear, believe in, rely, swallow, accept, understand); (2) (noun form) as make-believe, the enactment of a pretense with other synonyms (feigning, pretense, pretending, simulation, pretence, pretend). It seems that the hypernym (describing a word more broad) and hyponym (more specialised and specific) association (for verbs additionally the troponym) provide a specialisation and generalisation structure for word characterisations of models. Since models are instruments we may reduce consideration

![Figure 5. Characterisation of Models by Verbs](image-url)
to verbs and verbal expressions. For instance, engineering systems may be based on three activities that are related to the three words describe, specify, and prescribe. “Prescribe” itself is related to “instruct”, “direct”, and “guide”. “Instruct” with the meaning to “give instructions or directions for some task” can be linked to either (1) “give instructions or directions for some task” or (2) (less relevant for system engineering) “impart skills or knowledge to”. The third meaning of “make aware of” can be neglected.

If we restrict the word fields to some specific ones then a functions becomes properly shaped and the capabilities which are necessary can be drawn. The corresponding representation can be given as a category network. Each word field thus describes thus the mission and the determination we have in the mindset as habitual or characteristic mental attitude that determines how you will interpret and respond to situations. The mission separates the task into important ones and less important ones where the first enable to perform properly this task in the given scenario as an assignment. The determination decides and controls the model’s outcome or nature. The determination is an explanatory statement extended by how the model is used, what is the main background behind this instrument, and why to use this model. It provides basic ideas, features, particularities, and the utilisation pattern for the model.

4.2. Model Function Extended to the Cargo

We may now derive a general specification form for the function of a model:

- **Word field** describing the mission and the determination of a model;
- **When, how in general, what for, by whom, for whom** the model is used;
- **Capability and incapability attributes** as adjective characterisation of a model;
- **In what way, with what requirements** the model has been developed.

The specification can be narrative, sign-based, or implicit. We advocate the first form and an explicit statement about the function of a model. This explicit statement allows to reason about when this artefact is not a model but simply an object, which anti-profile has a model, and which specific considerations must be taken into account. The journey of a model is given by an association to one or several scenarios. For instance, a model suite in a waterfall approach consists of an inspiration model that is used before requirements are agreed, of a declaration model describing the objectives, of a prescription model for coding, of a documentation model for the code developed, and of an educational model used for elaboration of experience gained during development.

For instance, an entity-relationship diagram may be combined with a number of view schemata and realisation templates. In this case, it functions as a prescription model in an information system greenfield development scenario if it is definitely guiding the codification as a design of structuring and derivable viewpoints. View schemata are used for representation of user viewpoints and viewpoints for system’s operating how the data can be used. Greenfield scenarios start coding from scratch. Brownfield scenario use another kind of model.

The word field “guide” has two aspects: to act as a guide to and to direct in a codification; to direct, to supervise, to influence, to superintend the codification. Partial synonyms in this word field are “lead”, “steer”, “pilot”, and “engineer”. The background comes with the kind of supporting technology and the deep model that is governed by IS technology. Adequacy requires in this case mapping properties to the origins and to
the code. The focus is given by the origins that are exemplarily considered. The purpose is related to construction. Corroboration, validation, and plasticity are determined by the origins. Conformity is determined by the standards in the information systems area. Quality in use is based on the visual representation requirements and the transformation to code. External and internal quality are those that are commonsense in the area. The tight binding provides also the evaluation of the model. The community of practice is the modeller, the technologist, and the programmer. The binding to origins determines the requirements to the model.

The cargo [29] of a model consists of the model functions and additionally of the abstract declaration of the meaning, and of the narrative explanation of the identity. The abstract declaration of the meaning mainly contains main semantic and pragmatic statements about the model. The description of value of the model is determined by the functions in the utilisation scenarios and its importance within the given settings. The identity of a model is given by the actual and communicated identity. The three other kinds of an identity (accepted, ideal, and desired identity) are often neglected. The cargo can be considered as an abstract that describes the model. The cargo describes why a model should be accepted by some community of practice and in which scenarios the model has which functions, which context and background is (implicitly).

Any instrument that is used as a model has a cargo that determines its utility value and its present utilisation value. The function is the central ingredient of this cargo.

5. Conclusion

In our paper we have introduced a framework that clarifies the essence of modelling. In practice, the development of models in a variety of application scopes is not systematically guided and managed. We have started with a case study focusing in information systems (IS) modelling the area of modelling that is best known by the authors. We pointed out aspects that make IS modelling difficult and cause problems in model quality. Based on these findings we have developed stepwise our general framework explaining the essence of different models and different modelling practices. We have wanted to show the heterogeneity of models, but also the fact that model types have same basic properties, functions of models.

In our framework models are seen as as instruments that facilitate functions. The model has a goal (having a target state) that is expected to reach by modelling, purpose that enhances the goal by means that allow to reach the target, and function that extends the purpose by executable practices represented as scenarios. The paper handles the idea of functions of models in a wide manner covering different (classified) types of models.

The first step in our framework introduces the scenario approach - we see functions as scenarios. A wide variety of scenario types are discussed, but for detailed handling we have selected four scenarios - engineering, science, social and problem solving. This four scenario approach divides each main scenario into three sub-scenarios describing the typical characteristics of them. This approach provides us means to analyse the maturity of models functions in the scale of six. This analysis approach is derived from the principles of CMMI and ISO 33004 maturity models. The detailed handling is focused in engineering scenarios. We have developed similar analysis for the three other scenarios - social, problem solving, science but because of the limited space these were left out of the paper.
The second step of our framework we face the description problem for functions of models. Models are classified in this step according to their instrument properties in scenarios. For that purpose we have constructed three specific functions - inform, reconstruct, and engineer. Two alternatives to specify the functions of models are introduced - functions of models explicitly given by verb fields and model function as the main ingredient of a cargo.

As a summary - we have introduced a framework that can be used in analysis of modelling and developing modelling methods. It provides means for analyzing the maturity and quality of models. The framework is useful both for modelling practitioners and for those who are focused in developing modelling methods.

References


Models as Programs:
The Envisioned and Principal Key to True Fifth Generation Programming

Bernhard THALHEIM\textsuperscript{a,1} and Hannu JAAKKOLA\textsuperscript{b,2}

\textsuperscript{a} Christian-Albrechts-University Kiel, Computer Science Institute, 24098 Kiel, Germany
\textsuperscript{b} Tampere University, P.O.Box 300, FI-28101 Pori, Finland

Abstract. Programming became more and more comfortable with development of third and fourth generation programming languages. Although the fifth generation project did not achieve its goals, the necessity for more comfortability is still challenging. This paper delineates the path towards true fifth generation programming, based on literate modelling with model suites that generalises model-driven development and conceptual-model programming. A model suite consists of a coherent collection of explicitly associated models. A model in the model suite is used for different purposes such as communication, documentation, conceptualisation, construction, analysis, design, explanation, and modernisation. The model suite can be used as a program of next generation and will be mapped to programs in host languages of fourth or third generation. So, we claim that models will become programs of true fifth generation programming.

Keywords. model-based programming, models are programs

1. Introduction

Programming has become a common cultural technique, esp. for non-computer specialists, engineers, and laymen where the latter already start with simple tools like MIT Scratch programming or LEGO. Programs and computers have become an essential part of modern infrastructure. Programming is nowadays a socio-material practice in most disciplines of science and engineering. Despite the detailed research knowledge gained so far, the quality of programs decreases, for instance, due to the wide variety of program applications, due to the large variety of program libraries and their constant evolution, due to the numerous languages and toolboxes, due to integration and impedence problems among already existing programs, due to application of different programming cultures, and due to missing provenance and documentation support. Programming is not the most natural kind of communication for many programmers. They reason in a different language and at different abstraction levels. They often have difficulties in understanding their own programs later on or programs developed by others. At the same time,
systems become more complex and thus less comprehensible. We are thus approaching a \textit{software crisis} 2.0 \cite{Ama16,Far16,VM11,WVH17}. Programs are still developed on the basis of the third and fourth generation although the underlying mental concepts are not expressible in these languages. Moreover, critical software components for everyday life systems, for infrastructure, for management and control are developed by non-computer-specialists who are not familiar with matters of maintainability, risk avoidance, error tolerance, precision, completeness, integration, migration, and component coherence.

However, programmers have already initially and intrinsically an idea and models how to solve their problems and how to solve them. This idea is the rationale which underlies programming, i.e. it is a mental model of the solution that is going to be developed. As long as the models behind are only intrinsic and hidden, the solution background and the program ideas become tacit secrets of programmers. It seems far better if an appropriate support for modelling, gradual improvement, and refinement of the models is provided. If this support becomes sophisticated and code can be generated from models, the need for program development is reduced to the real problematic cases which are resolved by professionals. In this case, some models in a model suite \cite{Tha10} become programs at a higher level of programming. They are compiled to classical programming languages. \textit{Therefore, models become programs at a higher and more comprehensible level. They are more efficiently and correctly developed.}

\subsection{The Path Towards True Fifth Generation Programming}

Our approach fundamentally revises, combines, and corrects two already existing approaches: (1) Model-driven development approaches (MDD) (or engineering or architecture) are the latest developments (e.g. \cite{SV05}). Users start with requirements and continue with platform-independent models which can be specialised and refined to platform-specific ones. The platform-specific models are translated to code. Yet, the mental model behind the requirements is not explicitly considered. The approach does not consider the intrinsic details of all the models. (2) Conceptual-model programming \cite{ELP11} asserts that programming activities can be carried out via models. Models are complete and holistic, are conceptual but precise, and are executable. These models can be refined at any level of abstraction. However, the underlying foundations remain incomplete thus hindering full realisation. Both approaches have so far failed to fully generate deployable systems. The path towards model-based programming has however already been tested for web information systems. A third approach, which is mathematically precise, is based on abstract state machines \cite{BR18} that offers a semantically well-defined, pseudo-code language for specification at various abstraction levels. These models provide an accurate high-level description, support quality assessment, and can be mapped to third generation programs. By combining the first two approaches with the mathematically precise description, \textit{Modelling-as-Programming} (MaP) will be a \textit{springboard for next generation programming}. Next generation programming will be the first step towards true fifth generation programming.

\subsection{The Storyline of This Paper}

The paper develops a programme for true fifth generation programming that starts with models and uses models as a program specification. It is similar to second and third gen-
eration programming where programmers are writing programs in a high-level language and rely on a compiler that translates these programs to machine code. We propose to use models instead of programs and envision that models can be translated to machine code in a similar way. This paper presents the first starting vision to such novel kind of programming. The completion and full establishment of this programme may take a decade. Information system modelling is, however, already a positive proof of this kind of programming by models. Models delivered include informative and representation models as well as the compilation of the model suite to programs in host languages. Models will thus become executable while being as precise and accurate as appropriate for the given problem case, explainable and understandable to developers and users within their tasks and focus, changeable and adaptable at different layers, validatable and verifiable, and maintainable. Therefore, we start first with a discussion what models-as-programs means. Next we discuss literate modelling as high quality modelling with model suites. Section 4 describes our envisioned realisation path.

2. Modelling - The Next Generation Programming

Model research has a long, more than 2000 years old history in sciences, engineering, and daily life (e.g. [Mü16,TN15]). Computer science and engineering uses models as the main vehicle for system construction, description of problems and systems, explanation, and system quality investigation. Computer science has developed a very large number of model notions. As investigated in [TN15], these notions mainly differ according to the model purpose, the attention of the community, the background, and especially the language setting.

2.1. Modelling is Often Only Normal Modelling

The main difference to classical programming, model-driven development, and conceptual-model programming is the explicit orientation on the extrinsic surface model called normal model (yellow color in Figure 1). By contrast, the deep model (green color in Figure 1) consists of the background, the context, the intentions behind the model, the commonly accepted practice in the community of practice (CoP), and the setup of the model. The deep model and the normal model should however be considered as a whole.

We use the model notion from [TN15,Tha18] that is depicted in Figure 1. An essential result of the interdisciplinary brainstorming seminars of the modelling commu-

---

3A model is a well-formed, adequate, and dependable instrument that represents origins and that functions in utilisation scenarios. Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its CoP within some context and correspond to the functions that a model fulfills in utilisation scenarios. The model should be well-formed according to some well-formedness criterion. As an instrument or more specifically an artefact a model comes with its background that is often given only in an implicit and hidden form and not explicitly explained. The background consists of an undisputable grounding from one side (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities) and of a disputable and adjustable basis from other side (assumptions, concepts, practices, language as carrier, thought community and thought style, methodology, pattern, routines, common sense). A well-formed instrument is adequate for a collection of origins if it is analogous to the origins to be represented according to some analogy criterion, it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its purpose. Well-formedness enables an instrument to be justified by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity.
Figure 1. The conception and notion of a model with extrinsic elements of the normal model (yellow color) and intrinsic elements of the deep model (green color)

nity at Kiel University since 2009 has been the explication of the intrinsic enthymeme-like deep model within all models used in science and technology [TN15]. Modelling is currently mainly modelling of the surface-like normal model without explicit description of the background. The normal model is bound to its deep model. It is thus not entirely understandable to anybody, e.g. outside its context (e.g. discipline) and its CoP. The concentration on normal modelling is one of the main reasons why model-driven development and conceptual-model program have not succeeded as expected. If the deep model is not known and not understood then translation or mapping to platform-specific models becomes infeasible. This situation is similar to specifying LaTeX text without the corresponding strategic setup, e.g. by .cls, .clo, .def, .bst, .sty etc. files and libraries. The separation into the intrinsic and extrinsic parts of models is also depicted in Fig. 1 where the light blue part the normal model, the yellow part represents the mixed extrinsic and intrinsic part, the green part the part that is mainly build from the deep model. The explicit description of a deep model reveals the secrets within models.

The deep model has not been considered for model-driven development (MDE, MDA, MDD) and conceptual-model programming. This non-consideration is the main source for impedance mismatches between source and host languages, crucial translation problems, and the failure of these approaches. The explicit treatment of deep models and of high-quality source models (e.g. standardised generic and reference models) is

formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins. The instrument is sufficient by its quality characterisation for internal quality, external quality and quality in use or through quality characteristics such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions). A well-formed instrument is called dependable if it is sufficient and it is justified for some of the justification properties and for some of the sufficiency characteristics.
THE essential difference of our approach. Complete knowledge about the model is THE guarantee for modelling as programming.

2.2. Models as Model Suites

Researchers and engineers often collaborate in interdisciplinary and interacting communities. A model suite [Tha10] can also incorporate viewpoints and sub-models that support interaction and exchange with collaborating partners on the basis of sub-models. I envision that model-backed collaboration is far more effectively support collaborative work and problem solving in communities.

A model may combine several facets at the same time and may thus have its structure where some facets support specific purposes and functions. A model suite is a coherent collection of well-associated models at a variety of abstraction levels, foci, and scopes. The associations are explicitly stated, enhanced to explicit maintenance schemata, and supported by tracers for the establishment of coherence. Coherence describes a fixed relationship between the models in a model suite.

Model suites support holistic and consistent description of models at numerous levels of detail, precision, completeness, foci, and scopes depending on context, function of the model, community of practice, and origins that are really of interest. They close thus the gap among ideas and intentions, requirements, conceptualisations, and realisations. Models in a model suite support various functions such as communication support, mediator for system construction, basis for problem solving, facilitator for contracting and negotiation, documentation, analysis and quality assessment, support for integration, and warrant for migration and modernisations. Representation and informative models are typical models in a model suite. The latter can be generated. Models in a model suite can also be generated from others, e.g. in order to represent viewpoints [Tha10].

Model suite development is an intellectually challenging task if we aim at a complete model suite. For this reason, MaP also incorporates toolbox support.

2.3. Models are New Generation Programs

Models are currently used as a prescription or blueprint for programs of the third or fourth generation. We envision that models themselves can be considered to be the source code, i.e. models and model suites are essentially the program source. The independence concepts (hardware, operation system, physical, and logical independence) will be extended by programming language, platform, environment, and system independence since models can be transformed to different kinds of programming languages. The translation requires sophisticated compilers including optimisation facilities. Models in a model suite can be translated to code while other models in the model suite serve as communication and collaboration means in the CoP.

MaP proposes new programming paradigms, develops novel solutions to problem solving, integrates model-based and model-backed work into current approaches, and intents to incubate true fifth generation programming.
3. Literate Modelling as Literate Programming

3.1. Towards Literate Modelling

A holistic approach entirely based on models will thus provide a better support. The model support must include multi-model treatment with coherent model ensembles at different levels of abstraction, with explicit and maintainable associations among these models, with supported intellectual management of the complexity, with explicit knowledge of details of the models, and with sophisticated quality management. Models have also their anti-profile [Tha17] that limits applicability of model-backed development. Such a holistic and general approach would be too ambitious and unrealistic. Therefore, MaP focusses its scope on selected areas of Computer Science and Engineering. We, thus, start with four application areas of model-backed development and then use the experiences gained to extend our scope.

Already literate programming [Kn84] considered a central program together with satellite programs, especially for interfacing and documenting. We generalise and extend this new paradigm of programming with GibHub, ’holon’ programming, and schemata of cognitive semantics. Projects like the Axiom project or the mathematical problem solver [Pod01] have already shown the real potential of literate programming. The association among models must become manageable and be supported by computational features. The design and development of model suites has realigned the model ensemble approach to meet this challenge. One reason that literate programming has not become a mainstream paradigm is that implicit and intrinsic components remain largely unknown. Another reason is the missing representation of models behind the code and the missing representation of thoughts of people. A third reason is the hidden representation of the intention and the application task that has been the reason for developing a program. A fourth reason is the implicit usage of experience and of generic models behind the program solution. Our approach will reveal intentions, strategic and tactical issues (see Figure 2).

Model-backed development of systems will not be a universal solution to all computational problems. It is however a solution for those application cases for which users have an idea that can be expressed as a mental model. These models can be understood as interfacing or communicated models. In this case, the mental model can be enhanced by models characterising the problem space according to the needs, interest, and intentions of users. Users have their own understanding of the problem space, their educational and work environment, and their culture as ‘programming of their mind’ [Hof01]. Different users might use different models for the same application case. That means, we support modelling as literate modelling. It frees the modeller from the inherent and implicit parts of a model as modelling is understood at present and imposed by modelling languages and means that the modeller can develop models in the order of the flow of their thoughts. A model suite also explains the model and its intrinsic components in a natural language and is interspersed with snippets of representation and realisation models. This means that models are very easy to understand, to justify, and to share, as all its details are well explained. Literate modelling is a change of the mindset by making the story of the model suite explicit. It reduces bugs, misconceptions, and flaws in a model. Models are communicated to both people and machines.

Models can be transferred to programs if all details within the model are known and the models themselves are well-structured based on a sophisticated model language, e.g.
extended entity-relationship models with stereotyped and refinable profiles and directives for realisation (among stereotypes we may select the default one) [KT16]. A general model language would be the basis for a universal solution and thus cannot exist. We can however use modern engineering approaches. Engineers already develop systems based on standardised components. They use composition pattern and some kind of quality and failure management. Components and compositions can be coherently specialised in machine tool building. They are based on standards in this case. Database and workflow models can also be built in this form [MNS+13]. Standards are in these cases generic models or reference models. We restrict our approach to this kind of models. This standard-backed approach can also be applied to model suites. All models that are not directly derived from mental models are standard-backed models. Mental models are going to be enhanced and generalised in such a way that they become the source for a generic or reference model. This harmonised treatment then supports model-backed development of programs. Thus model suites become the source for programs.

3.2. Towards Next Generation Programming as Starting Point for True Fifth
Generation Programming

The rationale behind the initial fifth generation program language project was very ambitious [Mo82]: development of a general-purpose, multilingual environment and general-purpose problem solver that also supports natural language communication and multimedia processing; support for general knowledge representation, storage, processing, and retrieval; automatic problem-solving after accurate and abstract problem specification; closing the mental and the language gap between users and computers; development of sophisticated logical and functional machines for backend computation; developing an advanced architecture for support of these features; development of sophisticated theories and technology for support; development of a distribution and collaboration architecture. However, the initial fifth generation programming languages project was never completed. It failed despite its great technological and social advances since it was too early for the hardware progress, it was highly dependent on AI technology, it did not achieve an integration of AI and human-computer interface techniques, it was oriented on one programming paradigm and on mathematical logics, it tried to provide a universal solution for any kind of programming, it routed granularity to basic program blocks, and it was oriented on one final solution instead of coherent reasoning of coherent variants of final solutions depending on the focus and scope of a user4.

MaP now aims at true fifth generation programming where models are essentially programs of next generation and models are translated to code in various third or fourth generation languages. Programs of next generation programming must at the same time be understandable by all parties involved, support abstraction, be as accurate and precise for the problem space and the issues to be solved, transferable and distributable to other parties, commonly deployable by all parties, and support quality management and reasoning.

Due to its user orientation, next generation programming cannot rely on single language paradigms. Instead, models as programs must become language independent. Languages of third and fourth generation of programming languages became already hard-

---

4Our approach has been inspired by H. Aiso [Ais88] who chaired the architecture sub-committee in the Japanese fifth generation computer project [Mo82].
ware, storage, operating system, firmware, and optimiser independent. Language and platform independence will support non-specialists to program based on their models without forcing programmers to certain style of thinking and programming.

Our approach is based on stereotypes for deep models and on generic and reference models as a starting point for normal models. Model-driven development, engineering and architecture (MDD, MDE, MDA) taught some valuable lessons reported about model-driven approaches, e.g. those in the list in [Web95]. Two main problems limited the applicability of MDA and MDD: the intrinsic and not explicitly stated deep model and the restrictions in layering development.

*Models are a more natural kind of human reasoning* than programs could be. Programs are often oriented on algorithmic thinking. Most programming languages use simple variable spaces. Object-orientation has added essentially user defined object-relational structures. Network diagrams are one of the reasons why the entity-relationship structuring became quickly popular. All these structures are, however, one-facetted, cognitively simplistic, and without multiple viewpoint representation. *Human communication* is partner-oriented, is ambiguous in structure and meaning, uses partially semantics, is culture-dependent, is oriented on parsimony instead of completeness or preciseness, uses previous communication histories, considers principles of communication such as politeness, uses background information, and incorporates ellipses and context-dependent abbreviations. Models are represented according to the communication flow and the communication partner. Models thus must have a number of faces (or contrast classes and relevance classes [Fra80]) that can be used interchangeably. *Human thinking* does not separate syntax, semantics, and pragmatics but treats them as a coherent and larger whole. It is rather based on mental models such as collections of interrelated personal perception models or environment- and culture-oriented domain-situation models. Moreover, it is complex, multi-facetted, highly adaptable to different viewpoints and opinion, multi-viewpoint oriented, and network-connected.

*Non-professional programmers* are confronted with problems of transferring their understanding and their models to algorithmic and computer-oriented environments. The transformation process from thoughts to programs is error-prone, is oriented on the normal case without consideration of the entire picture, requires one central representation, and some understanding of computing technology.

Therefore, it is far better to support *non-professionals by model-backed programming* instead of forcing them to learn and to fully understand programming in third or fourth generation languages. True fifth generation programming is a better model-based representation of problems. In this case, the model must be understood both in their extrinsic, directly represented components and their intrinsic background. The second part has not taken into consideration in model-driven development and conceptual-model programming. This second part is, however, a central necessity for model-backed next generation programming. The explicit treatment of this part will become a `silver bullets’ for the new programming.
4. The Programme and Its Realisation Path

4.1. The Layered Model Development Framework

The layered approach has already often and successfully been used in Computer Engineering. Most program language realisations follow this approach since COBOL and ALGOL 60 development (e.g. infrastructure definition; variable space; program space; interpreted or compiled code) and application development (e.g. application case; infrastructure; design; specialisation & tuning; Deliver). Layering has also been the guiding paradigm behind text processing, e.g. behind the TeX and LaTeX realisations [Knu86,Lam94] with a general setup layer, the content layer, the adaptable device-independent layer, and the delivery layer. We assume that this approach is the universal basis for treatment of models as programs at least for programming by non-specialists. The approach for professional programmers is different. It can, however, also be supported in this manner how the success of programming environments such as Eclipse has already been demonstrating. These toolboxes have become accepted for their ease of use.

The model suite will be layered in Figure 2 into models for initialisation, for strategic setup, for tactic definition, for operational adaptation, and for model delivery. At the left side the issues for the model suite are represented. The right side displays the activities and methods for the development. The corners of the octagon represent the starting and final stages as well as sources and enablers of the intermediate stages. We restrict the picture to the layered model development process. The complete model suite thus becomes the source for the code of the problem solution, and for the system to be built. Currently, one model is considered to be the final product. Model suite development results in a number of models: deep, generic, specific, and normal models. Since any model has its deep elements, we start with the development of this deep model. In many cases, we use reference models or generic models (or tactical model frames like those used in

![Figure 2](image-url)
data mining and analysis). Models have their own background that is typically not stated explicitly but intrinsically. Methods for developing and utilising models are considered to be given. The intrinsic part of a model and these methods form the deep sub-model [Tha18]. The deep model is coupled with methodologies and with moulds that govern how to develop and to utilise a model. The deep as well as the general model are starting points for developing the extrinsic or “normal” part of a model. Consideration of modelling is often only restricted to normal models similar to normal science [Kuh70]. Classical modelling often intentionally presupposes the initialisation and intrinsic layers and assumes that these layers cannot be reconsidered and specifically changed according to the functions. The developer thus loses the understanding of the model and why the model is dependent without an understanding of these layers. Model suites, however, integrate these models over all layers. Another main obstacle why model-driven development and conceptual-model programming has not yet succeeded is the non-consideration of modelling moulds.

Intention modelling extends rational-based software engineering [BCM+08]. The W*H specification pattern [DT15] can be applied to model initialisation as well as includes then the following set of statements: (1) a plan, function, and purpose dimension (model as a conception: ‘wherefore’, ‘why’, ‘to what place or end’, ‘for when’, ‘for which reason’) within a scenario in which the model is going to be used as an instrument; (2) a user or CoP dimension (‘who’, ‘by whom’, ‘to whom’, ‘whichever’) that describes the task portfolio in the CoP and profile of users including beliefs, desires and intentions; (3) an application and a problem dimension (‘in what particular or respect’, ‘from which’, ‘for what’, ‘where’, ‘whence’); the added value dimension (evaluation). The initialisation layer may also be enhanced by a contrast space for user-related separation of a model and a relevance space that is dependent on the user [Fra80]. The contrast and relevance spaces as a form of mind-setting also define what is not of interest.

The enabling intrinsic setup layer defines the opportunity space and the infrastructure for the model. The results will be on the one hand a deep model and from the other hand a modelling framework or modelling mould that guides and govern next activities. In future, the developer will define the context and the most of the background (the grounding (paradigms, postulates, restrictions, theories, culture, foundations) and the basis (assumptions, concept world, practices, language as carrier, thought community and thought style, methodology, pattern, routines, common sense)) of the model. The context, extrinsic, and strategic dimension answers question like “at or towards which”, “where about”, “to what place or situation”, and “when”. Additionally, developers decide which methodology and environment seem to be the most effective and purposeful. The development and deployment dimension (“how”, “whence”, “what in”, “what out”, “where”) defines the modelling methodology, i.e. the modelling mould.

Deep model elements will be separated from elements of the normal model at the extrinsic source reflection layer. According to the model function, the normal model represents extrinsic elements of potential origins based on their content and thus answers questions such as “what”, “with which”, and “by means of which”. It reflects the extrinsic theory essentials that are necessarily to be represented, e.g. conceptions or pre-conceptions from the theory that is underpinning the application. The normal model can be built from scratch (“greenfield” modelling). It is usually based on experience gained. The latter case thus starts with a generic or reference model that might incorporate parameters. The extrinsic source reflection layer can be understood as a tactical layer.
Generic or general normal models are adjusted to those that a best fitted to those origins that are considered for the application in the operational customisation layer. This layer is sometimes holistically handled with extrinsic reflection. Inverse modelling uses this layer for adaptation of the model to the observational data (e.g. data adaption in astrophysics or parameter instantiation in most data mining processes). In some cases, this layer seems to be trivial. It is not trivial in the general case however. It instantiates parameters, adapts the normal model to those origins (or data sources) that are really under consideration, prepares the model for the special use and to the special - most appropriate - solution, and integrates the deep model with the normal model. The normal model is typically pruned in order to become simpler based on Solomonoff and Occam principled deviation and error-prone. The (normal) model could be enhanced by concepts and thus become a conceptual model.

The final result of the modelling process is a model suite that is adequate for origins, properly justified, and sufficient at the delivery and product layer. We cannot expect that one single model is the best instrument for all members of the community of practice. A sophisticated model that integrates deep and specific normal models is delivered to some members. An informative model that is derived from this model can be better for other CoP members. Models delivered in the finalisation stage are often enhanced by additional annotations, e.g. relating the model to the demands for members of the CoP by answering the ‘with’, ‘by which’, ‘by whom’, ‘to whom’, ‘whichever’, ‘what in’, and ‘what out’ questions. At the delivery and product layer we, thus, generate a number of associated models.

Models delivered in this approach become more reliable and - in the general sense - dependable on the explication of the deep model and of the initialisation. Dependability can now be considered according to dependability that is already given by deep models from one side and by generic and reference models from the other side. For both kinds of models we use stereotypes and pattern similar to the usage of class and setup libraries in LaTeX and the special document templates that provide a specific parameterised structure for content development for LaTeX content input (e.g. .tex and .bib files). Skipping the operational layer can be an option if a single model is delivered as a program collection. The typical case, however, is adaptation, fitting, pruning, specialisation, operationalisation, and exemplification at the operational layer.

Transformation is based on standardised combinable components (not only basic elements) as pattern and templates. Generic and deep models are going to be developed on the basis of standardized stereotypes and pattern. The specialisation and combination of models is supported by a model algebra that generalises the ER algebra [MNS+13]. Each specialisation can be enhanced by directives similar to pragmas in C++.

4.2. A Path towards Modelling-as-Programming

Computer Science and Engineering has resulted in many tools for support of programming. However, we observe that most of these tools have been oriented on bottom-up representation of program language elements and constructors for programs. Some of the tools provide some kind of abstraction. Very few tools also allow introduction of components and support modularity with refinement. Many tools are based on their own language variant and own interpretation, e.g. [Kru04]. Tools should however be based on standard components that can be refined for specific purposes. This path of componenti-
sation is essentially implemented with programming languages of the second, third, and fourth generation - at least for bottom-up elements and block concepts. Many tools tend to be unnecessarily complete in order to provide the full flexibility. All tools consider syntax on its own, define semantics of elements and construction on top of syntax, and do not consider personalised pragmatics. Natural languages have however collocations for words, holistic syntactical-sentential constructions, and their special interpretation in dependence on the context and the community of practice.

We are going to partially represent generic normal models in three frameworks:

- ADOxx [KMM16] is a configurable meta-modelling development and configuration platform that supports specification in a larger variety of graphical modelling languages. It follows the MOF (meta-object-facility) approach by OMG [PM07] based on a separation of abstraction layers of specification languages: M1 as the layer of model creation and description; M2 as the layer model language specification (considered as meta-model); M3 as the layer of frames for language specification (considered as meta-meta-model). The compiler approach can be integrated into ADOxx.

- Ptolemy II [Pto18] focuses on actor-oriented modelling of complex systems. The application of Ptolemy II in our approach must, however, consider a number of specific problems and must develop solutions to them. Ptolemy is oriented on bottom-up level of components. Abstraction in specifications is still an issue. There is a high freedom for specification and thus the approach struggles still with standardisation. Generic normal model can be used for standardisation. Intrinsic strategic and tactic parts of models are not yet considered. The model suite concept fits well into Ptolemy II.

- KIELER [Kie18] provides an eclipse-based framework for diagrammatic model specification. It aims at improving comprehensibility of diagrams, in decreasing development and maintenance time, and in providing facilities for analysis of dynamic behaviour of diagrammatically represented processes. Semantics is based on sequentially constructive sequence charts. Normal models can be represented as long as they are given in diagrammatic form and as long as their semantics if based on sequence charts.

Model-based development and architecture as well as conceptual-model programming have also been bound to imperfect tools. Moreover, they fail since the deep model is not taken into consideration. They meet thus all the classical impedance mismatches. A proper transformation can only be developed if either the source and target share their deep models or the deep models are transformed as well. Above all, programs are developed by people who have their culture, esp. programming culture and who are biased and framed by their way of programming and working.

This approach is based on a number of new assumptions: models consist of specialised and refined components that are combined via construction expressions; model components can be stereotyped and refined based on a specialisation approach; interdependence of refinement can be handled by attribute grammar constructions; construction expressions can also be stereotyped; stereotypes form the strategic layer of description; stereotypes can be specialised to pattern at the tactical layer and to templates at the operational layer; stereotypes, pattern and templates form semiotic units with their own specific syntax and with their fully integrated semantics.
Each sub-discipline in Computer Science has developed its specific style of modelling. This style is based on specific languages which have their specific grammar. Following the Eugenia [PKP14] framework, attribute grammars can be developed for these languages. In this case compiler-compiler approaches become applicable [BL74].

The Kiel team has been participating in tool development for database design, database engineering, and database performance management. Starting in the 1980s with the RADD (Rapid Application and Database Design) workbench (e.g. [AAB+95]), we systematically extended the domain of structure specification by database programming (finally with the VisualSQL tool (e.g. [JT03])), by performance management and tuning (e.g. [TT11]), by integration of workflow specification (e.g. [BR18]), by integrating web information systems design (e.g. [ST04]), and by code design (e.g. [Tha04]). These specification methods have been extended to a methodological framework (e.g. [JMTV05]) that finally reached maturity level 3 in SPICE in 2005 in one of our collaboration projects. We have also developed the translation tools for transformation of (conceptual) models to code.

4.3. A Proposal for the Realisation Approach

The implementation approach to MaP is inspired by four projects.

(a) **Transformers and compilers for conceptual database models**: The RADD toolbox (Rapid Database Development) is based on the conceptual entity-relationship modelling language. This graph-based language supports conceptual development, documentation, reasoning, and requirements engineering for database analysis, design, and development. The graph-oriented approach has been compiled on the basis of graph grammars [AAB+95, Run94, Tha00] combined with attribute grammar approaches. The conceptual schema formulated in this language or enhancements of this language can be used for derivation and compilation of realisation schemata, especially for object-relational and XML platforms. It is enhanced by view suites, visual query systems (VisualSQL), and by performance optimisers. Currently, the development is transferred to ADOxx [KMM16] from OMIlab. The compilation approach is presented in [KMM16].

(b) **DEPOT-MS (DrEsdner PrOgrammTransformation)** [BL74]: DEPOT system is a compiler-compiler for domain-specific languages (DSL) (historically: little languages, application-domain languages (Fachsprache)) that has been used to compile specific language programs to programs in the mediator language (first BESM6/ALGOL, later PASCAL, finally PL/1 [GHL+85]) which can then be translated to executable code. The approach integrates the multi-language approach [Ers81], attribute grammars [GRW77, RF87], and theory of grammars [Hut86, Tha75]. The system is similar to the MetaCASE toolbox [Dah97] or development environments, e.g. Ptolemy II.

(c) **LaTeX and TeX** [Knu86, Lam94]: The TeX and LaTeX approach is based on a strict onion or layered approach with (1) an internal layer for formatting and general initialisation (e.g. .fmt, .tfm, .fd, .def, .ltx, .dat, .afm, .cfg, .clo files), (2) a structure-style-language layer (e.g. .cls, .sty, .ldf, .bst files) that includes many additional library packages, (3) the input document suite (mainly .tex, .bib, .ist files), (4) the internal supporting and generated layer (e.g. .aux, .log, .lof, .bbl, .ind, .toc, .lof, .log files) that also support related applications, (5) a generic intermediate output layer (especially .dvi files), and (6) a delivery layer (e.g. .pdf, .ps, .html, specific printer files) for multiple output variants.

(d) **Libraries of reference models** (e.g. [BKV17, FL07, KMM16]): Libraries and toolboxes of solutions and programs are widely used in science and engineering. Specific
reference models are universal models [MJ04, Sil01] as well as generic models [Tro16]. Universal and enhanced models may be algebraically combined [MNS+13] and refined [dRE98] based on a model calculus. Models may also be enhanced by metadata descriptions and by informative models [DT12,Kra18]. Models and model suites may be evaluated based on their potential and capacity [BT15].

The integration of these four technologies is very ambitious. Generation of programs from models extends the models@runtime initiative [BFCA14] by direct compilation of programs from given models instead of enhancing runtime environments by models and abstractions. The proposed layering might however provide a solution for this integration and the necessary harmonisation. The variety of application-domain languages is as large (an estimation stated about 2,500 such languages in 1985) as the one of DSL [Fra11] or multi-level languages. Our layered architecture for models is going to be combined with the abstraction/refinement approach [Bör07,dRE98]. The layerd architecture became a common culture in Computer Science. Modern systems have been built on this kind of layering for system development (e.g. by layering into application case - infrastructure - design - specialisation & tuning - delivering), for problem solving frameworks (e.g. by task ordering ((1) problem case, (2) setting, (3) incubation, (4) enlightening, (5) finalising), for data analysis (e.g. by workflow pattern ((1) define & identify, (2) select solution class, (3) select solution pattern, (4) derive parameter values, (4) fit & prune, (5) finalize), and for engineering (based on general approaches ((1) know it, (2) understand it, (3) construct it, (4) configure it, (5) use it)).

The first three inspiring projects are based on compiler technology, attribute and graph grammars, pattern and stereotype architectures [ANT14], and principles of programming languages (starting with early thoughts [Lan73] to more advanced ones [GGZ04,SGM02,Wir96]). We oriented model transformation on macro-level, component-oriented, and refinable translation [FG11] instead of meso- or micro-level transformations used for most syntax-oriented translators. Models typically consist of associated and bundled components that have their inner specialisation.

The translator is also used for generation of warnings and error messages for systematic improvement of models. Since we start with development of generic normal models and deep models, we concentrate on quality assessment and improvement for normal models. Normal models should be as adequate and dependable for the given application. Later on, quality establishment is extended to the strategic and tactical layers.

5. Conclusion

MaP aims at true fifth generation programming as a new programming paradigm where models are essentially programs of next generation and models are translated to code in various third or fourth generation languages. Users program by model development and rely on the compilation of these models into the most appropriate environment.

5.1. The Intended Outcome of Our Approach

The main outcome of MaP is a proposal for true fifth generation programming as programming by models. The capacity and the potential of MaP are evaluated. The strengths, weaknesses, opportunities, and the threats are demonstrated in the four application areas in such a way that they can be used for an extrapolation to other application areas.
An essential outcome of MaP is the *layered architecture* and a realisation of modelling as programming. Model suite are used as a foundation for literate programming. Quality and literate models are understandable, transferable, distributable, and commonly usable. MaP users may design their own modelling styles, templates, stereotypes, and configuration. They also may concentrate on development of normal models while inheriting the initialisation and configuration, the deep models, the methodology, and the techniques for model realisation and model representation.

The MaP approach supports *programming by everybody at any time*. Models become the main means for collaboration among partners. Models may evolve and therefore evolution and modernisation are less painful tasks. For non-specialists in programming, models are typically of higher quality than the programs. Therefore, the generated programs are of higher quality. Models can also be used for communication and exchange of experience. Modelling as programming thus support sustainability of developed solutions. The model is then the code. The compiler assures that program execution corresponds to the conceptual specification thus making the model directly executable. At the same time, model suites treat the model in an explicit, complete and holistic way without any intrinsic and hidden details. Elements of a model suite are conceptualisations of the thoughts and understanding of developers. They are precisely defined and commonly agreed with the concepts in the application space. I will thus attain a good level of parsimony for model and therefore program developers. Furthermore, application evolution is going to occur at the level of the model suite. Modernisation, migration, and evolution occurs at the level of the model suite and does not require consideration of lower level details.

MaP supports model-based reasoning as a natural kind of reasoning. Solutions can be developed in a large variety of reasoning styles including hypothetical, abductive, inductive, deductive, and other advanced reasoning methods. Models can be refined. MaP thus also supports inverse modelling.

### 5.2. Are You Still Programming or Are You Already Modelling as Programming?

Programming has become a central technique in science and engineering. Software systems are often developed by non-specialists in programming without a detailed knowledge and skills, without an insight into the culture of computer science, and without plans for systematic development. These systems and programs often have a poor structure and architecture, little documentation, and lost their insight and knowledge of specific solutions.

Programs of the future must be understandable by all parties involved, must be accurate and precise enough for the task they support, and must support reasoning and controlled realisation and evolution at all levels of abstraction. Programming languages are currently languages of third or fourth generation. Those generations have so far provided hardware independence, linker independence, operating system independence, and execution code independence. Programs are nowadays compiled or at least interpreted and do not require system knowledge by the ordinary programmer.

In the past, the fifth generation computing project sought to develop systems and programs that are closer to people in their communication and knowledge processing capabilities. It should have been a shift to a new paradigm of human-oriented computing in the sense of T. Kuhn [Kuh70]. This world-wide project failed despite its great technolog-
ical and social changes because it was too early, it was highly dependent on AI technology, it did not achieve an integration of AI and human-computer interface techniques, it was oriented on one programming paradigm and on mathematical logics, it routed granularity to basic program blocks, and it was oriented on one final solution instead of coherent reasoning of coherent variants of final solutions depending on the focus and scope of a user.

5.3. Envisioned Deliverables of MaP

This paper develops the general approach to true fifth generation programming. The realisability of the approach has already been demonstrated for database development [KT16]. Database specification follows the global-as-design approach. BPMN specification follows the local-as-design approach. This approach requires view schema specification for data support of the workflow diagrams. The co-design approach [Tha04] is the basis for integration of workflow specification to database specification.

The general proof of concept is however a task of the future. Our programme can be based on development of the following deliverables:

Model suite description language: The model language consists of an associated bundle of languages for model elements that users may modify depending on their needs or simply reuse them as already established sub-cultures. These elements are different from ordinary programs because they are essentially declarative rather than imperative. Similar to UML stereotypes, MaP model class and model style languages are ready for application, can be extended, combined, and adapted. Users don’t have to work on the details of the models as programs. The system takes over the integration and composition work as it deduces the consequences of the model. It also provides a new discipline of modelling according to which principles of a particular modelling language design can be stated precisely. The underlying intelligence does not remain the secret of the modellers. It is spelled out in the style language and based on the model class language. Thus, coherence and consistency can readily be obtained where they are desirable. New model elements can readily be extended to new elements that are compatible with the existing ones. The model suite description language is developed as a collection of grammars, grammar-aware theories and software, and techniques for implementation.

Technologies, techniques, methodologies, and modelling moulds: The development of models in a model suite is based on a model library with models that can be used as an inherited or initial model for systematic composition of the model suite. The approach to model construction is canonised on the basis of methodologies and modelling moulds which systematically combine known and novel techniques and technologies for model development. Modelling moulds enable the modeller to reuse systematics and theories that have been successfully deployed in the past. They enable us to start with application space models, with deep models, with generic models, and with reference models without explicit reinvestigation of these models. The explicit agreement on a given mould eases, enables, and supports an economic development process.

Environment for an extension towards modelware as next generation literate modelling: Our approach aims at development of a general infrastructure for treatment of models as programs. This infrastructure includes also standardised solutions that can be reused in other applications. These solutions are based on application space models and deep models that are typically less volatile than normal models. Therefore, the library
allows quick and well-based modelling by non-specialists which may concentrate on development of normal models instead of developing the entire holistic model. They may accept the library models as a basis and then use generic and reference models as a starting point for normal model development. The model suite is also transformed to programs in programming languages of third or fourth generations. This approach disentangles modellers from programming and allows them to concentrate on the model development. The model is then the product. The transformed program is of a higher quality and more liable.

Compiler-compiler approach to model realisation for models as programs: The modelling infra-structure is an essential element for realisation of model suites and for treatment of models as programs of next generation programming. The metamodel represented in Fig. 1 is a model of a model. Its components and its associations are expressed as attribute grammar rules. The compiler-compiler approach is enhanced by the layered handling of models according to Fig. 2. All components of a model in the model suite are explicit and become thus transformable to representation models and to programs of third or fourth generation of programming. This generation is the basis for language and platform independence of the models themselves. Modellers thus become programmers of next generation programming languages. The quality of the generated programs is therefore higher than a non-specialised programmer could achieve. Compilation allows the integration of standards. Model suite libraries become then the kernel for modelware. Models become directly executable.

Tested, verified, and validated approaches for MaP: The MaP approach is gradually developed in four application areas for which I and my collaboration partners have sufficient experience. It will be assessed, evaluated, analysed, questioned, scrutinised and generalised in such a way that I will open the path to an extension of the approach to other application space, to other CoP with different interest and intentions, to other problem spaces, and to other concept space. This extension will be developed in our scientific network.

References

References


Abstract. There is no common agreement which artifact should (not) be considered to be a conceptual model although the term ‘conceptual model’ is used for more than for five decades in computer science and for more than one century in science and engineering. A team from all faculties at our university has been able to develop a notion of model that covers all model notions known in the disciplines of this team. We now introduce three notions of conceptual model in this paper: light, slim, and concise versions of the notion of conceptual model. The paper answers the following questions: Are all models also conceptual models? What is a conceptual model? Is there a formal notion of a conceptual model? What is not yet a conceptual model? What will never be a conceptual model? What is a concept? Which philosophical and scientific foundations we should consider while modelling? Is the existence of an ontology a necessary prerequisite for the being as conceptual model?

Keywords: conceptual model · concept · model theory.

1 The Model

Humans have learned to use instruments for handling their issues, tasks, and problems in daily life. Sciences and engineering also widely use instruments. Human evolution, sciences, and engineering have been enabled by wide instrument utilisation. The language is one of these instruments – often seen as one of the main. Models are another main instrument in modern computer science and computer engineering (CS&CE). They are often material artifacts. They might, however, also be immaterial or virtual.

It is surprising that models and modelling (and its variants such as conceptual models) have not yet properly founded. This paper contributes to close this lacuna.

1.1 Models are Main Artifacts and Universal Instruments

Models have become one of the main artifacts in CS&CE. This wide usage has not led to a common agreement about the notion of a model. The same observation can be made for other scientific disciplines, for engineering, and for daily life. In our area models became as artifacts the main instrument for system and software construction.
Models might be combined with other artifacts\textsuperscript{1}. Concept and conception development might be integrated into models. In this case, models might be considered as conceptual models.

A Notion of Model

What is a model? According to \cite{7, 27, 29} we define the model notion as follows:

"A model is a well-formed, adequate, and dependable instrument that represents origins and that functions in utilisation scenarios.

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice (CoP) within some context and correspond to the functions that a model fulfills in utilisation scenarios."

Well-formedness is often considered as a specific modelling language requirement. The criteria for adequacy are analogy (as a generalisation of the mapping property that forms a tight kind of analogy), being focused (as a generalisation of truncation or abstraction), and satisfying the purpose (as a generalisation of classical pragmatics properties). The generalisation of \cite{10, 15, 17, 24} is necessary for consideration of model-being.

The model has another constituents that are often taken for granted. The model is based on a background, represents origins, is accepted by a community of practice, and follows the accepted context. The model thus becomes dependable, i.e. it is justified or viable and has a sufficient quality. Justification includes empirical corroboration, rational coherence, falsifiability (in our area often treated as validation or verification), and relative stability. In our area the quality characteristics can be based on software quality characteristics and procedures for evaluating these characteristics.

Scenarios determine functions of models as instruments. A model is utilised. This utilisation is bound to scenarios in which a model functions. Typical scenarios are system construction (with description, prescription, and coding sub-scenarios), communication, negotiation, conceptualisation, documentation, and learning. The model might have several functions in complex scenarios. For instance, a model functions as a blueprint for realisation in a prescription scenario. Other typical functioning are the usage as an informative means, as a companion, as a guide for development. The quality of a model must be sufficient for this usage. Therefore, models used for description and models used for prescription might be different.

The main qualification of models is the potential utilisation as an instrument. This utilisation is based on methods which are developed in the discipline.

\textsuperscript{1} Due to the utilisation of artifacts as instrument we will concentrate on the instrument being of artifacts. This approach allows us to additionally consider virtual ‘artifacts’ such as mental models. An artifact is “something created by humans, usually for practical purpose. It is a product of artificial character due to extraneous (as human) agency”. \cite{3}. Furthermore, models can be real artifacts as well as thoughts. An additional difficulty is the negative usage of “artifact” in engineering as artificially introduced change (e.g. in presentation, miss or imperfection).
Models are used in Sciences, Engineering, and Daily Life

*Models and model suites.* There is no CS&CE subdiscipline that does not use models. Since models are abstractions and more generally are focused they are far better to use for investigation and system development. They are used in problem solving, in social, in engineering, and in science scenarios in a wide variety of forms. Often, models either consist of sub-models or form a model suite what is a well associated ensemble of sub-models. The models in a model suite [5] coexist, co-evolve, and support solutions of subtasks.

*Models are one of the first instruments before languages.* [9] Daily life utilisation of models is often unconscious, subconscious or preconscious. One of the first models that is learned by everybody is the ‘model of mother’. It is used before we spell the word ‘mother’. It has is variety of interpretations depending on the kind of behaviour of the mother. Models might be perception models that allow to summarise observations.

*Matrices and deep models.* Models typically consist of a relative stable part and of a part that is dependent on the actual circumstances. Typical modelling languages in CS&CE are predefined, use a limited vocabulary, have a relatively fixed – at least lexical – semantics, and allow to express certain aspects. They use their own techniques in some stereotype way, i.e. their utilisation follows some mould. Origins of models are often mental models such as perception or domain-situation models. The model background forms the deep sub-model. The current model is then the ‘rest’, i.e. a normal model. The utilisation and the mould form the matrix of the model.

*Memes as basic and deep models.* Humans reason, memorise, and express their thoughts based on memory chunks. Some of the chunks are relatively persistent and become memes [2, 25] which are then units of cultural evolution and selection. These memes are combined with some identification facilities. They represent a number of properties. They may be combined with other memes. They may be activated and deactivated. They can be grouped. They become reasoning instruments. Memes are thus already models, in most cases primitive or basic mental ones.

1.2 Why there is no Commonly Accepted Notion of a Conceptual Model: 1001 Notions and 101 Scenarios

*Why the large variety of notions of model?* Already [28] discusses 60 different notions of conceptual model. The variety of notions of model in CS&CE is far larger. Each of these notions concentrates on some aspects and implicitly assumes other properties. One reason for the disagreement on a common notion is the concentration on one utilisation scenario.

The implicit and hidden usage of deep models and the corresponding matrices of one – if not the main – cause for the manifold of model notions in CS&CE.
May we develop a common understanding of the notion of model? The two main sources for the variety of notions allow a systematic harmonisation of model notions. The definition given above is a result of a discussion on models in agriculture, archeology, arts, biology, chemistry, computer science, economics, electrotechnics, environmental sciences, farming, geosciences, historical sciences, languages, mathematics, medicine, ocean sciences, pedagogical science, philosophy, physics, political sciences, sociology, and sports at Kiel university that continues now for almost 10 years. The discussion is summarised in the compendium [30]. We got a shared understanding of the notion of model, of model activities and of modelling. So, we can envision that a common understanding and a coherent collection of notions of model can be developed. The collection supports a coherent notion that allows to concentrate on the specific utilisation scenario and the specific functions that a model has to fulfill.

1.3 The Storyline of the Paper and Our Agenda

Tasks and foundations of a theory of conceptual models. This paper bases the notion of conceptual models on four observations: (I) Conceptual models integrate concept(ion)s from a conceptualisation into a models. A notion of conceptual model might be a slim, light, or concise one depending on the level of detail we need in model utilisation. (II) A conceptualisation is based on a collection of concepts. (III) Origins of conceptual models are perception models and domain-situation models. (IV) These origins are formed by our understanding of the world, i.e. our observations and our compilations of these observations. We shall answer questions 2, 3, 6-8 from the abstract in the sections and use these answers for answering questions 1, 4, and 5 at the end.

We start with the last observation that leads us back to Ancient Greece. Next we develop an enhanced theory of concepts for an understanding of a conceptualisation. We may now define what are the components of perception and domain-situation models. Finally we arrive with three notions of conceptual model. We thus head forward to a science and culture of modelling in Figure 1.

Towards a science and culture of models, modelling activities, and modelling. Modelling is currently a creative art, i.e. a skill acquired by experience and observation, and may potentially be enhanced by study. The art extends daily life intelligence that is a part which lays the foundations for modelling, conditions, and socialises. The first level of modelling is based on daily life intelligence between humans or of humans with their environment. Humans become introduced to the deep model and especially the background – in most cases at some preconscious level.

Model science is based on a system of knowledge that is concerned with modelling art and that entails unbiased observations and systematic experimentation. The foundation we envision orients on fundamental laws. We understand, establish, deliberately apply the knowledge, and formalise it. Modelling culture is shared in a community of practice, is based on well-developed principles and methods as well as on established guidelines and practices. Wisdom requires the sapience of schools and matured modelling.
2 Model Theory and its Philosophical Foundations

The earliest source of systematic model consideration we know is Heraclitus [14] with his concept of λόγος (logos). Explicit model development and model deployment is almost as old as the mankind, however. For instance, Ancient Egyptians already made sophisticated use of models [6]. So, essentially, it is an old subdiscipline of most natural sciences and engineering with a history of more than 5000 years [18]. The notion of model has not explicitly used at that historic time. It was, however, the basis of understanding, manipulation, and engineering.

2.1 Plato’s three Analogies

Plato’s Republica (for a survey on Politeia see [1] or Y. Lafrance) uses in the sixth book three analogies which can essentially be understood as the underpinning of the concept of model. We follow here the Lattmann’s [13] investigation that led to a deep revision of the interpretation by Aristoteles.

The three analogies provide a general understanding of the model-being, of models, and of modelling. We may only observe phenomena of reality, form then trusts in beliefs and observations, next develop conjectures, might next judge and hypothise, and finally provide explanations.

The analogy of the sun distinguishes the ‘good’ visible world and the intelligible world. The sun stands for the visible things (i.e. ‘empirical’, ‘physical’, and ‘material’ world) and gives the light. The opposite worlds is the world of reasoning, of beliefs, conjectures, ideas, and explanations.

The analogy of the divided line distinguishes the visible world and the reasoning about this world as the thinkable (called intelligible world). The visible world can be separated into the physical things themselves (beliefs (pistis) about physical
things) and the reflections and observations about them (called shadows) (eikasia as the illusion of human experience). The intelligible world consists of (mathematical) reasoning and thought (dianoia) and of deep understanding (moesis).

Plato represented these four dimensions by a line (reflections(AB) - physical things (BC) - thoughts(CD) - understanding(DE)). We shall see in the sequel that a four plane representation allows deeper understanding.

The analogy of the cave explains why humans can only interpret the world based on their observations, i.e. shadows that we can see. The reality cannot be observed.

2.2 Revisiting the Analogies for Understanding the Model World

The four segment presentation in Plato’s analogies can be transferred to a four plane meta-model with

![Diagram](image)

Fig. 2. The model world with the separation of concern into visible, mental, and intelligible worlds

- a separation into a quantitative area and a qualitative area for meaning and opinion (doxa) and a qualitative area for thoughts (noesis) from one dimension and
- a separation into a perception and pre-image area and an area for conceptualisation from the other dimension.

The intelligible world thus consists of a mental world and the real intelligible world. This observation allows us to reconsider the model-being in the approach depicted in Figure 2.
The visible model world mainly reflects the observations and their perception as phenomena. So, we can consider models of the visible world as models of the first generation.

The mental model world consists already of compilations to perception models that reflect someone’s understanding or domain-situation models that represent a commonly accepted understanding of a state of affairs within some application domain.

The intelligible model world includes conceptualisations and theory development.

3 Concepts and Conceptualisations

The separation of model worlds in Figure 2 provides a means to distinguish clearly between models and conceptual models. With this distinction we may now neglect the hypothesis [20] that any model is a conceptual model. We thus solved the demarcation problem for distinction of models into perception, domain-situation, conceptual, and theoretical models.

3.1 Conceptualisation

Conceptualisation is a reflection and understanding of the world on the basis of concepts from some commonly accepted concept spaces. Similar to ontologies, a conceptualisation is never unique. There is no conceptualisation such that every other one can be transformed from it. Conceptualisation means to find adequate concepts and conceptions for representation of a visible and mental world. It aims at the development of knowledge about these worlds. It is based on derivation of abstract concepts and experience, of (scientific) understanding and perception that can be applied in similar worlds, of (pragmatical) experience for modelling, and of reference models for model-driven development (MDD) approaches.

Weakening the Rigidity of Classical Concept Theory

The word ‘conceptual’ is linked to concepts and conceptions. Conceptual means that a thing, e.g., artifact is characterised by concepts or their conceptions. The word ‘conceptional’ associates a thing as being or of the nature of a notion or concept. Therefore, we distinguish the ‘conceptual model’ from ‘conceptional modelling’. Classical concept theory and concept systems in mathematical logics are based on a Galois relationship between extensions and intentions of concepts, i.e. a concept is defined as a pair of an intention and of an extension where the intention is fully characterised by the extension and the extension is fully described by the intention. Each definition of a concept is a logical equation consisting of a definiendum and a definiens. We will use here an extension of the classical theory of concepts (e.g. [19]) by R. Kauppi’s theory of concept properties [11, 26]).
Brentano, Bolzano and Twardowski (e.g. [4, 16]) distinguish three kinds of mental phenomena and inner consciousness: ideas, judgements, and volitions. Concepts and conceptions might be based on prototypes that allow to partially characterise the current understanding of the intention but do not provide a complete characterisation. Mental phenomena, beliefs, intuitions have their prototype view and a representation through best (counter-)examples that a person has been observing. Moreover, they can be represented by an exemplary view that characterises exemplars through similarity relation with measures, weights, and stimuli for their acceptance.

Conceptions are systems or networks of explanation. R.T. White [32] has already observed that concepts are not the same as conceptions. Concepts can be used in the meaning of classification and as an abstraction of a set of knowledge a person associates with the concept’s name. Conceptions are however systems or networks of explanation. Conceptions are thus far more complex and difficult to define than the either meanings of the concept.

Conceptional modelling is modelling with associations to concepts. A conceptual model incorporates concepts into the model. Conceptual structures include conceptions (concepts, theoretical statements (axioms, laws, theorems, definitions), models, theories, and tools). Concepts are linked together in a complex multi-dimensional network (is-a-kind-of, is-a-part-of, ...). The links are of variable strength.

3.2 Concepts for Conceptualisation

An advanced concept notion must allow to define a concept in a variety of ways. Some definitions might be preferred over others. They can be application and time dependent, might have different level of rigidity, have their validity area, and can only be used with a number of restrictions. We combine R. Kauppi’s theory of concept features with the concept treatment by G.L. Murphy [19].

The definition frame for concepts [23]: Concepts are given by tree-structured structural expression of the following form

\[
\text{ConceptTree}(\text{StructuralTreeExpression}(\text{Feature, Modality(Sufficiency, Necessity), Fuzziness, Importance, Rigidity, GraduationWithinExpression, Category}))).
\]

Features are elements of a concept with some modality, some Fuzziness, importance, rigidity, some graduation and some category. A feature is either a basic feature or is a concept.

Conceptions are typically hierarchically ordered and can thus be layered. We assume that this ordering is strictly hierarchical and that the concept space can be depicted by a set of concept trees.

A concept might be given by several definitions. A concept is also dependent on the community that prefers this concept. Consider, for instance, the mathematical concept of a set by an enumeration of its elements, by inductive definition of its elements, by an algorithm for the construction of the set, or by explicit description of the properties of the set. Which of the definitions is more appropriate depends on the application domain.
Interleaved meta-hypergraphs form hyper-networks of concepts: Our definition frame has the advantage that concepts which share features can be decomposed into the shared feature collection and the rest. Therefore, we may base our concept collection on a number of basic concepts.

The network of concepts is a meta-hypergraph \([21]\]

\[
MG = (MG^V, MG^{MV}, MG^E, F^{MG}, \Sigma^{MG})
\]

(1)

with a set of meta-hypergraph vertices \(MG^V\), a set of meta-hypergraph meta-vertices \(MG^{MV}\) which are subsets of meta-hypergraph vertices, a set of meta-hypergraph edges \(MG^E\) connecting vertices. A vertex and an edge is described by a set of features \(F^{MG}\). The semantic restrictions are given by \(\Sigma^{MG}\). An example of a meta-hypergraph is displayed in Figure 3\(^2\).

**Fig. 3.** A meta-hypergraph with vertices \(v_1, \ldots, v_5\), meta-vertices \(mv_1, mv_2, mv_3\), and edges \(e_1, \ldots, e_7\) without explicit features.

Conceptions can now be defined as a layered ensemble of meta-hypergraphs. We start with a primary network at layer 0 and associate next layer networks by embedding mappings to a hyper-simplex from networks at lower layer. A simple example is displayed in Figure 4.

### 3.3 Concept Granules as Basic Constructs of Conceptualisations

Concept granules are collections of concepts and/or conceptions given as meta-hypergraphs and ensembles with specified typicality of features (typical, moderately typical, atypical, borderline), with specified relevance of concept features, and with assigned importance of concept features.

Conceptualisation enhancements of a given model consist of

1. a context given for various aspects in dependence on the matrix,
2. a concept granule with several interrelated expressions as alternatives (competing, ...), with abstracts, with extensions (motivation, explanation, ...), and
3. witnesses as collections of illustrating best (counter-)examples (potentially with several concept trees) mainly based on images/observations on origins.

\(^2\) We acknowledge the communication with J. E. Gapanyuk from Bauman Moscow State Technical University (10.10.2018) who proposed this illustrations in Figures 3 and 4.
Fig. 4. A meta-hypergraph ensemble associating a simple primary network simplex $PS$ and a first-order network simplex that associates via $\Phi_1$ the vertex $v_5$ with a hyper-simplex of vertices $v_4, v_3$.

4 Conceptual Models

Mental models and their elements may be associated to concepts. The elements of a model are interpreted by concepts and conceptions. This interpretation is based on a judgement by somebody that conceive model elements as concept(ion)s within a certain scenario. If the scenario changes then the association to concepts changes as well. [28] categorises more than 50 notions of conceptual model depending on the function that a conceptual model has in a given scenario. We use this categorisation and develop now three integrated notions of conceptual model. Which one is used depends on the complexity of consideration.

4.1 Perception and Domain-Situation Models as Origins

Perception and domain-situation models [28] in Figure 2 are specific mental models either of one member or of the community of practice within one application area. It is not the real world or the reality what is represented in a perception model. It is the common consensus, world view and perception what is represented. Perception models are dependent on the observations, imaginations, and comprehension a human has made. Domain-situation models describe the understanding, observation, and perception of an application domain. The description is commonly accepted within a community of practice.

4.2 The Notion of Conceptual Model

The large variety of notions of conceptual model is caused by the scope of modelling, by the application case under consideration, by the main scenario in which the model functions, by the variety of origins that are represented by the conceptual model, by modelling languages, by the stand-alone orientation instead of integration into a model suite, and by the focus on normal models without mentioning the underpinning by a deep model. It is now our goal to consolidate three versions in such a way that they form a view depending on the level of detail and abstraction. The notions can be refined to an application domain,
e.g. to database modelling: “A conceptual database model is a conceptual model that represents the structure and the integrity constraints of a database within a given database system environment.” [29]

The Slim, Light, and Concise Notion of Conceptual Model

*Slim version: Conceptual Model ⊒ Model ⊎ Concept(ion)s* [29]: A conceptual model incorporates concepts into the model.

That means that models are enhanced by concepts from a number of concept(ion) spaces.

*Light version: Conceptual Model ⊒ Model ⊕ Concept(ion)s* [28]: A conceptual model is a concise and function-oriented model of a (or a number of) perception and/or domain-situation model(s) that uses a concept(ion) space.

This notion generalises and enhances a notion that is used in simulation research [22]: “A conceptual model is a concise and precise consolidation of all goal-relevant structural and behavioural features of a system under investigation presented in a predefined format.”

*Concise version: Conceptual Model ⊒ (Model ⊕ Concept(ion)s) ◂ Enabler* [8]: A conceptual model is a model that is enhanced by concept(ion)s from a concept(ion) space, is formulated in a language that allows well-structured formulations, is based on mental/perception/situation models with their embedded concept(ion)s, and is oriented on a matrix that is commonly accepted.

The conceptual model of an information system consists of a conceptual schema and of a collection of conceptual views that are associated (in most cases tightly by a mapping facility) to the conceptual schema [31]. Conceptual modelling is either the activity of developing a conceptual model or the systematic and coherent collection of approaches to model, to utilise models, etc.

Literate programming [12] considers a central program together with it satellite programs, esp. for interfacing and documenting. This paradigm has become the basis for GitHub and model suites. Conceptual modelling is typically explicit modelling by a model suite. Association of conceptual and other models in a model suite might follow the layered approach to model coherence maintenance and to co-evolution of models.

Descriptive and Prescriptive Conceptual Models.

A model functions in a number of scenarios. For instance, the conceptual model is used in documentation, negotiation, learning, communication, explanation, discovery, inspiration, modernisation, reflection, and experience propagation scenarios. We may categorise and enhance the notion of conceptual model depending on given scenarios. The system construction scenario integrates description and prescription scenarios beside specification and coding scenarios.
One main scenario for conceptual database models is the description scenario. A conceptual model as a descriptive conceptual model is a deliverable of an understandable (may be, ready to apply or to practise) and formalised (or well-formed) [concept-based], unconditionally acceptable conceptualisation of perception and domain-situation models for interaction and discourses.

For database applications it is thus a model suite consisting of a conceptual database model (or schema), of a collection of conceptual views for support of business users, and of a collection of commonly accepted domain-situation models with explicit associations to views (see [31]).

The second main scenario for conceptual database models is the prescription scenario. A conceptual model as a prescriptive conceptual model is a coding supporter as an analysed or synthesised, ready-to-apply blueprint because it can be deployed, it is unconditionally accepted, and appraised in a deliberately and precise practice as a tacit tool which provides notion explanations [from descriptive conceptual models].

For database applications it is thus a model suite consisting of a conceptual database model (or schema), of a collection of views for both support of business users and system operating, and of realisation templates (see [31]).

4.3 Models, Languages, and Ontologies

The major goal of an ontology [16] is to determine what exists and what not. It is independent of humans to conceive it and what kinds of existing things there are. It is independent of perception models although it can be shared among humans. Languages might be textual, visual or audio ones. The classical modelling approach often assumes artificial or partially formal languages.

Languages as enablers for conceptual models: Most models are language based. The language is an instrument similar to models. Moreover, the first models that a human develops are preconscious or subconscious, e.g. the model of a ‘mother’. Languages are however enablers since the words in languages can be used for denoting concepts. Many conceptual languages integrate several languages, e.g. ER modelling uses the vocabulary from a domain and a graphical language for schema representation.

Conceptual models must not be based on an ontology: The notion of ontology is overloaded similar to the notion of model. Ontologies are considered as shared and commonly agreed vocabularies.

A controlled and thus matured ontology must combine a controlled vocabulary, a thesaurus, a dictionary, and a glossary. There is not real need for associating such ontologies with models.

Languages are not necessary preconditions for conceptual models: Social models are often used for teaching human behaviour. They are based on concepts which might also be not explicit or integrated into the deep model. They are thus conceptual models. We observe however that in most cases conceptual normal models use some language.
5 Conclusion

We developed an approach to conceptual modelling with an explicit integration of concepts into the model. This explicit integration is based on a theory of concepts, conceptions, and conceptualisation. Concepts are developed for our understanding of the world we observe. Therefore, perception and domain-situation models become the origins of our conceptual models.

There are models that are not conceptual models: Sciences and engineering use models without explicit integration of concepts. It is often also difficult to use concepts within the model. A model performs a function in a scenario. Explicit conceptualisation would make the model more complex and thus less useful.

What is not yet a conceptual model: Middle-range theories are essentially mediator models. They are used for mediation between qualitative theories (e.g. their conceptualisations) and quantitative observations. For instance, sciences such as archeology make use of modern or medieval concepts without having yet an appropriate concept for prehistoric time, e.g. the concept of settlement or a village. Another typical model that might be enhanced by concepts is the graph model for the König’s bridge problem that uses pathes within a graph for solving this problem. The topographical model for the bridge problem uses the concepts of islands and bridges and thus allows to explain the solution.

What will never be a conceptual model: Most life situations do not need conscious models since we can live with what we have learned. Preconscious, unconscious, and subconscious models guide life, emotions, and intuitions. Conscious models require efforts and thus must have an explicit need. Concept(ion)s must not be explicated since there might be no necessity in that.

References

Usage Models Mapped to Programs

András J. Molnár\textsuperscript{1,2}[0000−0002−2194−0320] and Bernhard Thalheim\textsuperscript{1}[0000−0002−7909−7786]

\textsuperscript{1} Christian-Albrechts-University Kiel, Computer Science Institute, D-24098 Kiel, Germany \\ {ajm,thalheim}@is.informatik.uni-kiel.de \\
\textsuperscript{2} MTA-SZTAKI Institute for Computer Science and Control, Hungarian Academy of Sciences, H-1111 Budapest, Hungary \\ modras@ilab.sztaki.hu

Abstract. Model-based programming can replace classical programming based on compilation and systematic development of models as well on explicit consideration of all model components without hiding intrinsic details and assumptions. A key element of model-based programming is the proper definition and management of model suites, by which multiple, interrelated models can be transformed from one another and their consistency is ensured after modifications. A usage model is based on the specification of user roles and types, together with an interaction space described in a form of a storyboard, showing which activities are supported, in which order, by which actors. A workflow model is an extended, well-formed declaration of how specific processes should be carried out. It can directly be translated to program code, using a proper workflow or process engine. A novel way of programming is being opened up by usage modeling, which is being investigated in this paper: given a storyboard with supported usage scenarios, it is possible to derive a workflow model from it. We present our two translation methods using a working example, identifying guidelines as requirements for model refinement and normalization, rules for model translation, and propose considerations towards improved methods and model specifications.

Keywords: Model-centered programming · Model to program · Model suite · Model transformation · Storyboard · Process model.

1 Introduction

1.1 Programming by Modeling

Programming is nowadays a socio-technical practice in most disciplines of science and engineering. Software systems are often developed by non-programmers or non-computer scientists, without background knowledge and skills, or insight into the culture of computer science, without plans for systematic development. Maintenance, extension, porting, integration, evolution, migration, and modernisation become an obstacle and are already causing problems similar to the software crisis 1.0, since such systems often have a poor structure, architecture,
documentation, with a lost insight of specific solutions. Programs of the future must be understandable by all involved parties and must support reasoning and controlled realisation and evolution at all levels of abstraction.

Our envisioned true fifth generation programming [13] is a new programming paradigm where models are essentially programs of next generation and models are translated to code in various third or fourth generation languages. Programming is done by model development, relying on the compilation of these models into the most appropriate environment.

Application engineers and scientists are going to develop and use models instead of old-style programming, supported by templates from their application area. They can thus concentrate on how to find a correct solution to their problems, managing the complexity of software intensive systems. The process will be supported by model-backed reasoning techniques, as developers will appreciate and properly evaluate the model suite at the desired level of abstraction.

1.2 Usage Models and Workflow Models
In our study we are considering the case of web information system development.

A usage model of a web-is consists of specification of user roles and types, their associated goals and tasks, and an interaction space. The latter can be expressed as a graph, called a storyboard, describing what activities are supported and in which possible order, by which actors [9]. Supported interaction playouts can be formulated as scenarios (exact graph paths), story algebra expressions (path schemata), or more generally, subgraphs of the storyboard, including actorspecific views. The usage model is developed by a global-as-design approach.

A workflow model is an extended, well-formed declaration of how specific processes should be carried out, in a notation that is readily understandable by all stakeholders, including business analysts, technical developers and people who manage and monitor those processes [7]. A de facto standard is BPMN [7], but it is possible to use another workflow description language. The workflow model can directly be translated to software process components, using a proper workflow or process engine (e.g. [3]). This opens up a novel way of programming by usage modeling, via intermediate translation to a workflow model.

1.3 Related Work
Our current contribution can be related – amongst others – to the following previous works. Notions of models are discussed in [12]. [11] introduces model suites consisting of multiple, explicitly associated models, where the association uses maintenance modes, similar to integrity support in databases [15]. Amongst others, MetaCASE tools [1] were developed to support the definition of metamodel packages and the creation and customization of CASE tools based on them. The models as programs – true fifth generation programming agenda is proposed in [13]. For data structuring, translation of entity-relationship models to relational database schemata is well-known [4]. We are proposing a similar approach for the dynamics of functionality, motivated by compilers [8]: phases of preprocessing, parsing and syntax checking is followed by semantic analysis resulting an intermediate structure, and finally a possible optimization phase of the resulting,
translated model. [10] discusses *proceses-driven applications* and *model-driven execution* in terms of BPMN [7] diagrams. [2] elaborates a generative approach to the functionality of interactive information systems. [14] introduces dynamically combinable *mini-stories* to handle workflow cases with large flexibility. Although these latter works consider steps and ideas we can apply here, our currently addressed problem of usage model translation to workflow model is not explicitly discussed in any of the publications known to us.

### 1.4 Goal and Outline of the Paper

Our general vision is to generate running program code based on a usage model specification. We investigate on a particular sub-case in this paper: given a usage model as a storyboard with supported scenarios [9], is there a formalizable method to derive a workflow model in BPMN [7] from it. We present our proposed path and general framework for modeling as next generation programming in Section 2, based on [13]. Section 3 introduces our target case of workflow model elicitation from a usage model, illustrated by a working example, with general guidelines for model refinement and enhancement, rules and two different methods for translation. We conclude and close with future issues in Section 4.

### 2 Modeling and Programming Based on Model Suites and Layering

Models are universal instruments for communication and other human activities. Ideas and thought chunks can be presented to those who share a similar culture and understanding without the pressure to be scientifically grounded. A model is an adequate (i.e. analogous, focused, purposeful) and dependable (i.e. justified, sufficient in quality) instrument that represents origins and performs functions in some deployment scenario [12]. As an instrument, the model has its own background (i.e. grounding, basis) and should be well-formed. Models are more abstract than programs, but can be as precise and appropriate as programs. They support understanding, construction of system components, communication, reflection, analysis, quality management, exploration, explanation, etc. Models can be translated to programs to a certain extent, therefore, models can be used as higher-level, abstract, and effective programs. They are, however, independent of programming languages and environments. Models encapsulate, represent and formulate ideas both as of something comprehended and as a plan. Models declare what exactly to build and can be understandable by all stakeholders involved in software system development. They become general and accurate enough, and can be calibrated to the degree of precision that is necessary for high quality [13].

A *model suite* [11] consists of a coherent collection of explicitly associated models. A model in the model suite is used for different purposes such as communication, documentation, conceptualisation, construction, analysis, design, explanation, and modernisation. The model suite can be used as a program of next generation and will be mapped to programs in host languages of fourth or
third generation. Models delivered include informative and representation models as well as the compilation of the model suite to programs in host languages. Consistency can be ensured similarly to relational databases [15]. Models will thus become executable while being as precise and accurate as appropriate for the given problem case, explainable and understandable to developers and users within their tasks and focus, changeable and adaptable at different layers, validatable and verifiable, and maintainable.

Similarly to database modeling, layering has already often and successfully been used, including most program language realisations and application development methodologies. We assume a general layered approach as the universal basis for treatment of models as programs [13]. Layering has also been the guiding paradigm of the TeX and LaTeX text processing realisations [5, 6].

![Diagram of model suite development and program generation](image)

Fig. 1. The layered approach to model suite development and program generation

Model suite development and deployment will be based on separation of concern into extrinsic and intrinsic parts of models. Models typically consist on the one side of a normal model that displays all obviously relevant and important aspects of a model and on the other side of a deep model that intrinsically reflects commonly accepted intentions, the accepted understanding, the context, the background that is commonly accepted, and restrictions for the model. The model suite will be layered into models as shown in Fig. 1. Taking it as basis, we can formulate our proposed agenda for usage and workflow models.

The initialisation layer is given by the application and the scenarios in which models are used, by the problem characterisation, by background elements of the CoP and especially commonly accepted concepts in this community, and additionally by interest, intentions, and the value. In our case, it consists of a declaration that a website is needed for a specific application, analogously to selecting a \texttt{documentclass} in \LaTeX. It determines the possible syntax and semantics of the underlying layers.

The enabling strategic setup layer defines the opportunity space and especially the hidden background for the model. Its main result is the deep model
that is typically assumed to be given (normal models are not entirely developed from scratch). In our case, it will correspond to what a website means, what are the side conditions and underlying infrastructure of it and the selected application domain. It gives an opportunity space and can impose requirements or proposals for the way of system development.

The tactic definition layer starts with some generalisation, i.e. select a ground generic model that will be customised and adapted to become the normal model. It can be, for example, a generalization of a previous storyboard development, or a configurable storyboard composed of best-practice patterns. Decision of the modeling framework or language (here, the use of storyboarding, with or without story algebra usage, in which format) must have been taken. Generic modeling must be supported by meta-models assumed to be available as (re)usable packages. Further model contents are interpreted based on the selected packages.

The operational customisation layer fits, calibrates and prunes the model suite to the problem space. This is where the actual design is made, forming a normal model (here: the generic storyboard is customized as needed or allowed by the generic model: missing parameters are set up, defaults can be overridden). Requirements for an acceptable normal model must have been given in the generic model or the metamodel, in order to ensure well-formedness and consistency, and to allow proper model transformations possible on the delivery layer. The normal model(s) must be validated according to these requirements.

Finally, the model is delivered in various variants depending on the interest and the viewpoints of the CoP members. It is elicited from the normal model using a model translation, extraction or enhancement method. The target model language (here, BPMN) must be given with the selection and customization of the available translation methods. Interrelations and consistency management between the normal and the delivered model can be further declared.

The complete model suite thus becomes the source for the code of the problem solution, and for the system to be built [13].

3 Elicitation of Workflow Models from Usage Models

One of the main challenges for model translation is the existence of different intentions behind the two modeling languages. BPMN is stricter than storyboarding, while a proper translation needs to make use of the inherent flexibility of the storyboard. Therefore, besides a direct and full translation option, we are proposing a way for constructing BPMN workflows by formulating story path schemata, based on selected parts of the storyboard.

3.1 An Application Case and Its Usage Model

We are assuming the development an information system for a touristic and recreational trail network, providing guidance for visitors, as well as facility management of the trails and related field assets. A map interface is being provided with planning and navigation features along the designated trails, connected to an issue tracking system for reporting and managing trail and asset defects.
The high-level usage model is given as a storyboard on Fig. 2. Abstract usage locations represented by graph nodes are called scenes. Users can navigate between these scenes, by performing actions associated to directed transition links. Some indicative action names are given for the reflexive links (which are in fact, denote multiple links, one by named action). The entry point is marked with a filled black circle. An end point may also be added as a double circle (by default, each scene is assumed to be a potential end point).

![High-level storyboard graph for a sample trail management system](image)

**Fig. 2.** High-level storyboard graph for a sample trail management system

We declare three actor roles: visitor, trail manager and trail crew. A set of authorized actors are pointed to the bottom of restricted scenes by vertical arrows [9, p. 410]. By default, all actors are allowed to enter a scene.

The storyboard is about to represent supported normal scenarios, as specific means the users can accomplish given tasks. Context-loosing random navigations (e.g. back to the main page at any time) can be treated as breaking or canceling the started scenario, and starting a new scenario with a new context. These moves are not explicitly modeled so the focus can be kept on meaningful issues.

We are limiting our current discourse for one-session, one-actor scenarios.

The storyboard can be enhanced with input-output content specifications for each scene. We use the notation of [9, p. 410] so that input and output content for a scene is displayed using a short horizontal arrow on the left and the right side, respectively. Input-output content is named and an output content is assumed to be delivered as an input content to the next scene along each transition link, where the content names are equal. Square brackets denote optional input or output. Content names can be prefixed by generic database operation names.

### 3.2 Refinement and Normalization of the Storyboard

The top-level storyboard (Fig. 2) has to be refined and enhanced, so that actual scenarios as paths in the graph will be self-descriptive and consistent, and the graph is formally sound and contains enough details for a working and meaningful translation into workflow model(s). We state the following semantical
considerations and guidelines for developing the refined usage model. If all these
criteria are met, and guidelines have considered, we call the storyboard *normalized*. This is only partially verifiable formally – for the items marked with (*) –
and refers to a quality and stage of model development:

- Complex scenes must be decomposed into atomic sub-scenes, each having
  a single, well-defined action, task or activity which is fully authorized by a
  given set of actor roles. The interaction paths must be modeled by directed
  links between the sub-scenes and directly connected to outside (sub)scenes.
- Each transition link with active actor participation (action) must be replaced
  by a link-scene-link combination, where the action or activity is performed
  at the scene and the new links are only for navigation. This new scene can
  be handled and parametrized together with other scenes in a unified way.
- No parallel links between two scenes are allowed (*). They must either be
  translated using separate scenes (see above), or merged into one link, or their
  source or target scenes must be decomposed to separate sub-scenes.
- The routing decision (which link to follow after a scene) is assumed to be
  taken as part of the activity inside a scene, by default. If it is not intended,
  then only one outgoing link is allowed and an extra routing decision scene
  must be explicitly introduced after the original scene as necessary (this may
  be later optimized out).
- Unique names are assumed for all scenes and links (except that two or more
  links pointing to the same target scene can have the same name) (*).
- There must be a unique start node (entry point) with a single link to an
  initial scene and either a unique end node or a default rule declaring which
  scenes can be places for story completion (*).
- Each scene must be enhanced with a set of authorized actor roles. Without
  that, a default rule must be supplied. There must be no (normal) links
  between scenes without at least one common authorized actor role. (*)
- Input and output content is to be specified by symbolic names for each scene
  wherever applicable. Content names will be matched along the links (*): For
  each input content of a scene \( s \) there must be an output content with the
  same name provided by the source scene of each link directed to \( s \). Optional
  content is written in square brackets.
- Input and output content names can be prefixed by database operations:
  SELECT is allowed for input, while INSERT, UPDATE, and DELETE are
  allowed for output content. Without detailed semantics of these operations,
  a single central application database is assumed by default.

### 3.3 View Generation by Actor Roles

Given a selected actor role, a specific storyboard view can be generated for it
as a basis of role-specific workflow models, by removing unauthorized scenes
for a selected role with their links, resulting a cut-out of the storyboard, with
reachable scenes by actors of the chosen role. An enhanced, normalized version
of the visitors’ storyboard view is shown on Fig. 3, with multiple sub-scenes.
Links are denoted by italic numbers. An explicit end node is placed additionally,
reachable from chosen scenes.
3.4 Graph-Based, Direct Translation Method
At this point, a default translation algorithm we have developed, can be applied to generate a BPMN process flow diagram, based on the graph connectivity of the storyboard. Details of the algorithm are omitted due to space limitations, but the result of the translation of Fig. 3 is shown on Fig. 4 as a demonstrative example. The translation process can continue with enhancements of Section 3.8.

3.5 Modeling Supported Scenarios by Story Algebra Expressions
Alternatively to the previous method, a more sophisticated and targeted method is developed, if specific scenarios, which are intended to be supported by the system, are collected and expressed as patterns in a story algebra.

A particular playout of system usage becomes a path in the storyboard and is called a scenario. A set of possible scenarios can be modeled as using the story algebra SiteLang [9, p. 76], similar to regular expressions. Such a scenario schema can be a pattern for generating a workflow model. The original notation uses link names for description. We found that using scene names in the story algebra more naturally supports the translation to workflow models.

For example, a scenario schema of a visitor can be modeled out of the following variations: a visitor looks at the map, selects a destination point. The scenario may continue by reporting an issue for the selected point, or by planning a trip, navigating along it, and maybe at certain points, reporting an issue on-site. Each of these variants correspond to different scenarios the system should support and can be summarized as one or more scenario schemata.

Using abbreviated scene names (by first letters of words, e.g. mb stands for map browse), the above mentioned visitor scenarios can be modeled by the following story algebra expression (semicolon is used for denoting sequential steps, plus sign for at-least-once iteration, square brackets for optionality and box for expressing alternatives):
Fig. 4. Direct translation of visitor usage to BPMN, based on the storyboard graph. The full connectivity of the usage model is represented as possible process flow paths. Scenes become tasks. Numbers denote choices based on scene transition edges. Bracketed numbers are only for information, referring to original transition edges without alternatives. The grey-colored gates can be removed by merging their connections. Further refinements and optimizations are possible.

\[ mb; ps; (ir \Box (tp; (tsn; [ir])^+)) \]  (1)

Given a storyboard (view) specification, a scenario schema must be compatible with the given scene transitions, which means the following: Atoms of the story algebra expression must match to authorized scenes of the storyboard. The defined scenarios must correspond to valid directed paths within the storyboard (view). The defined scenarios must start with the marked initial scene and finish at the defined (or default) end scene(s).

Expression (1) is compatible with the visitors’ storyboard view (Fig. 3). Consistency of the input-output content declarations can also be checked along the possible playouts. Note the link 6 will not be available if the visitor is coming from link 7 (there is no navigated route to go back to).

3.6 Decomposition Into Mini-Stories
A scenario or story schema might contain semantically meaningful, reusable patterns of scene transition playouts, which can be combined with each other flexibly. Story algebra expressions, however, may be too complex and hard to handle by human modelers, and such semantical information remains hidden. A possible solution is to take the union of the relevant scenarios and decompose them into mini stories [14] (or, at least, extract some mini-stories from it).

A mini-story is a semantically meaningful, self-contained unit, which can be used flexibly in different scenarios, sometimes by possibly different actors. It can be modeled explicitly and translated as a reusable subprocess in the workflow model. Syntactic hints or heuristics can reveal possible mini-story candidates, but at the end the modeler has to explicitly define or verify them.
In our case, given the story algebra expression (1), candidate mini-stories can be recognized by maximal, non-atomic subexpressions with none of its non-trivial parts appearing elsewhere. Based on modeler decision taking into account semantics as well, we define the following two mini-stories, and substitute them in the story algebra expression (in a real case, with more scenarios, their reusability could be better verified): 1. \( \text{Select location from map}: \text{Slfm} \equiv \text{mb}; \text{ps} \) and 2. \( \text{Navigate along trip (with reporting issues)}: \text{Nat} := (\text{tsn}; [\text{ir}])^+. \) We keep referring to scenes \( \text{ir} \) and \( \text{tp} \) as atomic mini-stories. The resulting story algebra expression with the above mini-story substitutions of (1) becomes:

\[
\text{Slfm;} (\text{ir} \sqcap (\text{tp}; \text{Nat})) \quad (2)
\]

### 3.7 The Story-Based Translation Method

After the storyboard (viewed by an actor role, refined and normalized) and the desired story schemata (story algebra expressions) are given as above, with the mini-stories modeled, the workflow model in BPMN for each story schema can be elicitated the following, inductive way:

- Translate atomic mini-stories,
- Translate compound mini-stories based on their story algebra expressions (which are not translated yet),
- Compose the complex workflow based on the full story algebra expression.

Translation can be hierarchically carried over using structural recursion along the story algebra atoms and connectives, as displayed on Fig. 5. A choice for rule alternatives is proposed, with given defaults. The modeler can either leave the defaults as they are, or utilize the alternatives by applying pragma-like declarations to the usage model, or stereotypes associated to story algebra elements or subexpressions. For example, compound mini-stories can be translated as subprocesses, or connected using the link event notation. Conditionals for process flow gates match the names of corresponding storyboard edges (based on user choice) or their associated conditions or triggers (if such conditions are given for links of the storyboard).

### 3.8 Enhancement of the Translated Model

Transition link names (here, numbers) can be added to the workflow model, as well as input-output content as data objects and database connections associated to the workflow tasks and subprocesses (see the additional rules of Fig. 5).

Fig. 6 displays a result of the refined, normalized visitors’ storyboard view (Fig. 3) being translated to BPMN, based on story algebra expression (2) and mini-stories of Section 3.6, using rules of Fig. 5.

Model translation may be guided by additional information in forms of scene or link stereotypes.BPMN provides a variety of assets and some of them could be directly elicitated.Stereotypes offer more semantic information such as data or user-driven navigation, cancellation or rollback of started transactions, etc., to be mapped to native BPMN constructs.
Fig. 5. Translation rules for story algebra expressions and additional assets based on storyboard. Dotted-lined rectangles denote arbitrary workflow model parts already translated from story subexpressions. There is a default translation for each construct, with possible alternatives that can explicitly be chosen by the modeler.

Fig. 6. BPMN workflow translation of visitors’ view usage model, based on story algebra expression (2)
The modeling process is based on laying out default values for model formats, start/end scenes, authorized actor roles, context objects containing scenario history, handling of exceptions and invalid routing, stereotypes and other semantical or transformative guidance (e.g. how to connect mini-stories together, how to translate iteracted sub-processes). Defaults should work for conventional modeling cases. For customized, more sophisticated modeling, defaults can be overwritten. A possible post-translation optimization phase can improve the workflow model in each case. Most of these issues are left for future investigation.

4 Conclusion and Future Work

Model-based programming can be the true fifth generation programming, supported by sound foundation and appropriate tools, based on model suites of explicitly interrelated models. Models have their specific functions, viewpoints and can be given in various levels of details. Ensuring coherence, consistence and translatability among them is a crucial issue. In this paper, we have presented a general, layered modeling framework as a basis, and showed its feasibility by giving methods and guidelines for model development and translation between two specific types of models: the usage model (expressed by storyboard graphs and story algebra expressions) and the workflow model (expressed by BPMN).

The workflow model is claimed to be directly translatable to program code [10,3]. We have introduced the concept of user view and the normalization of the storyboard, providing guidelines to the modeler to refine an initial, top-level usage model. We gave two methods for translating the refined usage model to workflow models, and successfully applied the mini-story concept for semantically structured and flexible workflow elicitation. Translation is based on default rules, while alternatives can be chosen explicitly by the modeler.

The method is ready to be tested with more examples or prototype implementations. Future issues include actor collaboration modeling, defining stereotypes and pragmas determining model semantics and translations. The metamodeler has to implement packages of generic models and add-ons, enrich generic models with pre-defined patterns and templates. The actual application modeler can choose among them or let the modeling system decide on which defaults it uses for which cases. It points towards a generic model-suite framework, which is, in our view, essential for truly working general model-based programming.

References

7. OMG: Business process model and notation (BPMN) version 2.0 (2010)
Models: The Main Tool of True Fifth Generation Programming

Igor Fiodorov\textsuperscript{1} [ORCID number ???], Alexander Sotnikov\textsuperscript{2} [ORCID number ???], and Bernhard Thalheim \textsuperscript{3} [0000-0002-7989-7786]

\textsuperscript{1} Plekhanov Russian University of Economics, Russia
\textsuperscript{2} Joint Supercomputer Center, Russian Academy of Sciences, Russia
\textsuperscript{3} Christian-Albrechts-University Kiel, Computer Science Department, Germany

Igor.Fiodorov@mail.ru, ASotnikov@jscc.ru, thalheim@is.informatik.uni-kiel.de

Abstract. Models are one of the main and most commonly used instruments in Computer Science and Computer Engineering. They have reached a maturity for deployment as the main tool for description, prescription, and system specification. They can be directly translated to code what enables us to consider models as the main tool for modern software development. Models are the power unit towards new programming paradigms such as true fifth generation programming. This paper introduces model-centered programming as one of the main ingredients and main tool of true fifth generation programming.

Keywords: models, true fifth generation programming, model-centered programming.

1 Introduction

1.1 Towards New Programming Paradigms

Programming has become a technique for everybody, especially for non-computer scientists. Programs became an essential part of modern infrastructure. Programming is nowadays a socio-material practice in most disciplines of science and engineering. Solution development for real life complex systems becomes however an obstacle course due to the huge variety of languages and frameworks used, due to impedance mismatches among libraries and environments, due to vanishing programming expert knowledge, due to novel and partially understood paradigms such as componentization and app programming, due to the inherent tremendous complexity, due to programming-in-the-large and programming-in-the-web, and due to legacy and integration problems.

Programming languages have evolved since early 1950's. This evolution has resulted in a thousand of different languages being invented and used in the industry. First generation languages – although low-level and machine-oriented at micro-code
level - are still used for instruction-based programming. Second generation languages are assembly languages that can be translated to machine language by an assembler. Third-generation languages provide abstractions and features such as modules, variables, flow constructs, error handling, support packages, many different kinds of statements etc. Fourth generation languages are more user friendly, are portable and independent of operating systems, are usable by non-programmers, and have intelligent default. The fifth generation project has been oriented on logic programming and did not result in a wide acceptance and usage. The main supporting feature for programming is however that programs written in these languages are translated by compilers to programs in low-level machine languages.

Programs became an infrastructure of the modern society. At the same time, we face a lot of problem for such infrastructure. Its maintenance, extension, porting, integration, evolution, migration, and modernization become an obstacle and are already causing problems similar to the software crisis 1.0. Programs are developed in a variety of infrastructures and languages that are partially incompatible, in teams with members who do not entirely share paradigms and background knowledge, at a longer period of time without considering legacy problems at a later point of time, without development strategies and tactics, and with a focus on currently urgent issues. A crucial point is the development of critical software by non-professionals. Programming has already changed to programming-in-the-large beyond programming-in-the-small and is going to change now to programming-in-the-mind. Moreover, systems become more complex and less and less understandable by team members. The software crisis 2.0 (e.g. [15]) is also be exacerbated by understandability, communication, comprehension, complexity, and provenance problems.

We thus need better and more abstract techniques for development of our programs. We envision that true fifth generation programming can be based on models and model suites which can be automatically transformed to corresponding programs without additional programming.

1.2 Models as Programs

Our notion and understanding of models and model suites is based on the compendium on models in sciences and engineering [11].

A model is a well-formed, adequate and dependable instrument that effectively and successfully functions in utilization scenarios. It is adequate if it is analogous to the origins to be represented according to some analogy criterion, if is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and if it sufficiently satisfies its purpose. Well-formedness enables an instrument to be justified by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins. The instrument is sufficient by its quality characterization (internal quality, external quality and quality in) such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions). A well-formed
instrument is called dependable if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics. A model comes with its background, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background its often given only in an implicit form.

A model reflects only some focus and scope. We thus use model suites that consists of a set of models, an explicit association or collaboration schema among the models, controllers that maintain consistency or coherence of the model suite, application schemata for explicit maintenance and evolution of the model suite, and tracers for the establishment of the coherence.

A typical model suite is used for co-design of information systems that is based on models for structuring, models for functionality, models for interactivity, and models for distribution. This model suite uses the structure model as the lead model for functionality specification. Views are based on both models. They are one kernel element for interactivity specification. Distribution models are additionally based on collaboration models.

Model-centered development is used in many branches of modern computer science and computer engineering. Model-as-Programs approaches will become the traction machine characterized by slow beginning at present and a progressive increase in speed. In the sequel we discuss this change of paradigms for database development. A similar approach has already been practiced for editing systems such as literate programming and as the LaTeX environment or such as compiler-compiler approaches for domain-specific languages.

1.3 The Storyline of this Paper

Models and model suites became easy-to-use and easy-to-develop instruments that are used by everybody and therefore also by non-programmers. We envision that modern programming could be based on model suites that are translated to programs. As discussed in Section 2, this vision is already real for users that use advanced database development techniques. However, model-based database programming is used only in a less sophisticated and rather implicit form. Investigating the more advanced approach, we develop a path towards true fifth generation programming that is based on model suite development in Section 3. The entire framework is inspired by and can be considered as a generalization model-centric database development and modern specification approaches.

2 Case Study: Model Suites Direct Database Specifications

2.1 Data Specification for Database Applications with Conceptual Models

Conceptual schemata and models are widely used for database structure specification and as a means for derivation of user viewpoints [10,13]. These models are used for concept-backed description of the application domain or of thoughts, for prescription of the realization and thus system construction, for negotiation and iterative develop-
ment of the model decisions, and for documentation and explanation of the decisions made in the modelling process.

Viewpoints can be represented by view schemata that are defined on the main database schema by expressions given in an advanced algebra. Interaction models for business users are the third kind of models that are used in a database model suite. Collaboration models can be specified in a similar form and are based on viewpoints.

The usage of a conceptual model as a description model of thoughts and understanding in an application area is commonsense today. The usage for system realization must be based on specific properties of the database management platform and requires thus a lot of additional information. We thus enhance conceptual modelling by additional information. Pragmas and directives are essential elements that we use for enhancement of conceptual models for system realization. Pragmas have originally introduced for C and C++. Directives have been used as additional control units for compilation.

### 2.2 Transformation of Conceptual Models to Logical and Physical Models

Conceptual models and schemata are often taken as an initial structure for logical and physical schemata. The transformation is still often based on some brute-force interpreter approach that requires corrective specification for integrity maintenance and for performance management at a later stage by experienced database operators. The transformation of integrity constraints is not yet automatically enhanced by enforcement mechanisms and control techniques. Procedural enhancement on the basis of triggers and stored procedures is still a challenge for database programmers. Performance support includes at the first step CRUD supporting indexing. Support for querying can be based on hints.

The transformation approach can however be based on rule-based compilation. Essentials of rule-based transformation are in a nutshell: syntactical and semantic analysis of the models and schemata according quality characteristics within the platform setting; preprocessing of the models and schemata to intermediate normalized models and schemata; extension of the models by support models for performance support; derivation of integrity maintenance and other support schemes; derivation of association schemata for models in the model suite; derivation of tracers for coherence maintenance; rule-based transformation of models; optimization after transformation.

It does not surprise that this approach follows classical four-layer compiler technologies (lexical and syntactical analysis, derivation of intermediate models, preparation for optimization, translation, performance management) [14]. It is enhanced by a compiler configuration pragmas according to the profile of the DBMS. The models must be complete for performance consideration. Therefore, a number of directives have to be added to all models in the model suite: treatment of hierarchies, redundancy control, constraint treatment, realization conventions, and quantity matrices for all larger classes. Directives and pragmas must not be fully described. Instead we may use templates, defaults and stereotypes, e.g. realization style and tactics, default configuration parameters (coding, services, policies, handlers), generic operations, hints
for realization of the database, strategies for matching performance expectations, constraint enforcement policies, and support features for the system realization.

This transformation approach is already state-of-the-art for challenging applications. Advanced database programming is based on such techniques. Web information systems development uses such transformations [10]. However, it is currently the professional secret of database operators and administrators.

2.3 Generalizing the Approach for Database Programming

The specification and transformation approach has already becoming common practice for database structuring development. The Higher-Order Entity-Relationship Modelling (HERM) language [13] is the basis for development of conceptual database schemata and for specification of derivable view schemata. The latter are used for support of viewpoints for a given database system user community. Derivation is based on the HERM algebra that allows specification of user schemata. The specification of a database schema follows the disciplinary matrix of database development, e.g. approaches such as global-as-design and viewpoint support as derivable structures. This structuring may be enhanced by generic or reference models which are essentially package for modelling. The foundation and the models background is supported by the HERM theory. The entire schemata development is based on tools, e.g. ADOxx [4,6] as a specification environment. An essential element of this environment is a compiler for compilation of the specifications to logical schemata. The database developer specifies the schemata within this environment as HERM schemata, view schemata, and pragmas and directives as an additional description for database performance.

![Diagram](image)

**Fig. 1.** The model-centric database structure development with automatic mapping of conceptual models to logical models for HERM models.
Fig. 1 displays this approach. The database developer uses the development environment within the environment of the ADOxx workbench, the model definitions provided for HERM, the packages for schemata (e.g. a generic reference model for booking applications), the foundations provided by the HERM theory, and the transformation features embedded into the ADOxx generator [6].

3 Model Suites Used As Programs

3.1 Towards New Programming Paradigms

Modern programming languages provide as much as possible comfort to programmers.

Models are a universal instrument for communication and other human activities. Thought chunks can be presented to those who share a similar culture and understanding without the pressure to be scientifically grounded. Models encapsulate, represent and formulate ideas both of something comprehended and as a plan. They are more abstract than programs. They can be as precise and appropriate as computer programs.

They support understanding, construction of system components, communication, reflection, analysis, quality management, exploration, explanation, etc. From the other side, models can be translated to programs to a certain extent. So, models can be used as higher-level, abstract, and effective programs. Models are however independent of concrete programming languages and environments, i.e. programming language and environment independence is achieved. Models declare what exactly to build. They can be developed to be understandable by all main parties involved in system development. They become general enough and accurate enough. They can be calibrated to the degree of precision that is necessary for high quality.

Model-based programming can then replace classical programming based on compilation and systematic development of models as well on explicit consideration of all model components without hiding intrinsic details and assumptions. Our approach to true fifth generation programming will be extendable to all areas of computer science and engineering beside the chosen four exemplary ones (information system models; horizontally and vertical layered models; adaptable and evolving models; service line models). This paper develops a general framework to true fifth generation programming for everybody.

The framework is based on model suites since the user interface models and the collaboration models must be an integral part of modelling. Interface and collaboration treatment generalizes literate programming [5] to literate modelling as ‘holon’ programming that is combined with schemes of cognitive reasoning. Model suites enable the programmer of the future to develop their programs in a multi-faceted way. They can reason in a coherent and holistic way at the same time on representation models as the new interfaces, on computing and supporting models, on infrastructure models, on mediating models for integration with other systems, etc.

Application engineers and scientists are going to develop and to use models instead of programming in the old style. They will be supported by templates from their application area, can thus concentrate on how to find a correct solution to their prob-
lems, can manage the complexity of software intensive systems, will be supported by model-backed reasoning techniques, and will appreciate and properly evaluate the model suite at their level of abstraction. Literate modelling with model suites supports all members of a community of practice (CoP) by reflecting their needs and demands in a given situation and scenario by an appropriate model in the model suite. It becomes thus an effective and efficient means of communication and interaction for users depending on their beliefs, desires, needs, and intentions.

The generalization of the database approach is depicted in Fig. 2. We use a similar form as the experienced one that is displayed in Fig. 1.

![Fig. 2](image)

**Fig. 2.** The general approach to true fifth generation development based on models

Our approach proposes new programming paradigms, develops novel solutions to problem solving, integrates model-based and model-backed work into current approaches, and intents to incubate true fifth generation programming. This new kind of programming enhances human capabilities and could become the kernel of new industrial developments. Models are thus programs of the next generation.

### 3.2 The Layered Approach to Modelling

Our approach is based on model suites as the source, on systematic development of model suites in a layered approach, on compilers for transformation to programs in third or fourth generation, and on quality assurance for the model as a program. The notion of the model suite is based on [11]. Model suites generalize approaches developed for model-driven development from one side and conceptual-model programming from the other side. Model suite development and deployment will be based on separation of concern into intrinsic and extrinsic parts of models. Models typically
consist on the one side of a normal model that displays all obviously relevant and important aspects of a model and on the other side of a deep model that intrinsically reflects commonly accepted intentions, the accepted understanding, the context, the background that is commonly accepted, and restrictions for the model. The model suite will be layered into models for initialization, for strategic setup, for tactic definition, for operational adaptation, and for model delivery (see Fig. 3).

Model development can be layered in a form that is similar to the onion structure in Figures 1 and 2. We use essentially five layers for true fifth generation programming as shown in Figure 3:

1. an internal layer for general initialization,
2. an application definition language layer that includes many additional library packages,
3. the internal supporting and generated layer with its generic and reference libraries,
4. the input model suite that reflects the application and which is essentially the main task for an application engineer, and
5. the generic intermediate output layer, and its delivery layer for multiple output variants depending on the target programming language.

![Fig. 3. The five layers to model development (initialize, setup, reflection, customize, delivery)](image)

The model suite will be layered into models for initialization, strategic setup as an intrinsic setup, tactic definition as an extrinsic reflection, customization and operationalization as the main program development layer and as operational adaptation, and for model delivery. The complete model suite thus becomes the source for the code of the problem solution, and for the system to be built. Currently, a model is considered to be the final product. Models have their own background that is typically not given explicitly but intrinsically. Currently, methods for developing and utilizing
models are accepted as to be given. The intrinsic part of a model and these methods form is called deep sub-model. The deep model is coupled with methodologies and with moulds that govern how to develop and to utilize a model. The deep as well as the general model are starting points for developing the extrinsic or “normal” part of a model. Consideration of modelling is often only restricted to normal models similar to normal science. Model suites integrate however these model kinds. The main obstacle why model-driven development and of conceptual-model programming has not yet succeed is the non-consideration of the deep model and of modelling moulds.

4 Conclusion

We envision that true fifth generation programming can be based on development of high-level program descriptions that can be mapped to third-generation or fourth-generation programs. These programs may then be directly executed within the corresponding environment. This approach has already been the essential idea and its generalization behind a system for translation of domain-specific languages in the 80ies. The DEPOT-MS (DrEsdner PrOgrammTransformation) [7] was a compiler-compiler for domain-specific languages (historically: little languages, application-domain languages (Fachsprache)) that has been used to compile specific language programs to executable programs in the mediator language (first BESM6/ALGOL, later PASCAL, finally PL/1 [2]). The approach integrates the multi-language approach [1], the theory of attribute grammars [9], and theory of grammars [3,12].

A second source for true fifth generation programming is literate programming [5] that considered a central program together with satellite programs, especially for interfacing and documenting. This approach can be generalized and extended by new paradigms of programming (e.g. GibHub, 'holon' programming, schemata of cognitive semantics, and projects like the Axiom project or the mathematical problem solver [8] have already shown the real potential of literate programming. Our approach extends literate programming to model suites which are sets of models with well-specified and maintainable associations.

The developed framework, its theoretical underpinning and the realization approach is novel, targets at new programming styles, supports programmers from applications without requiring from them a deep program language knowledge and skills, and is going to overcome current limitations of programming. Layering is one of the great success stories in computer engineering. Already early languages such as COBOL used layered programs (division-section-paragraph-sentence-statement-command; ICCO: initialize-configuration-content_enhancement-operationalisation; environment-declaration-program). Our approach continues and generalizes this approach and will be thus the basis for true fifth generation programming.

A model in the model suite is used for different purposes such as communication, documentation, conceptualization, construction, analysis, design, explanation, and modernization. The model suite can be used as a program of next generation and will be mapped to programs in host languages of fourth or third generation. Models will become programs of true fifth generation programming.
Models delivered include informative and representation models as well as the compilation of the model suite to programs in host languages. Models will thus become executable while being as precise and accurate as appropriate for the given problem case, explainable and understandable to developers and users within their tasks and focus, changeable and adaptable at different layers, validatable and verifiable, and maintainable.

References

Towards a Great Design of Conceptual Modelling

Yasushi KIYOKI\textsuperscript{a,1}, Bernhard THALHEIM\textsuperscript{b}, Marie DUŽI\textsuperscript{c}, Hannu JAAKKOLA\textsuperscript{d}, Petchporn CHAWAKITCHAREON\textsuperscript{e} and Anneli HEIMBÜRGER\textsuperscript{d,f}

\textsuperscript{a}Graduate School of Media and Governance, Keio University SFC, Japan  
\textsuperscript{b}Christian-Albrechts-University Kiel, Germany  
\textsuperscript{c}Technical University of Ostrava, Czech Republic  
\textsuperscript{d}Tampere University, Pori, Finland  
\textsuperscript{e}Environmental Engineering Department, Faculty of Engineering, Chulalongkorn University, Thailand  
\textsuperscript{f}University of Jyväskylä, Faculty of Information Technology, Finland

\textbf{Abstract.} Humankind faces a most crucial mission; we must endeavor, on a global scale, to restore and improve our natural and social environments. This is a big challenge for global information systems development and for their modelling. In this paper, we discuss on different aspects of conceptual modelling in global environmental context. The paper is the summary of the panel session "The Future of Conceptual Modelling" in the 29\textsuperscript{th} International Conference on Information Modelling and Knowledge Bases.

\textbf{Keywords.} Conceptual modelling, model suites, multi-agent system, artificial intelligence, machine learning, semantic computing, data mining, 5D World Map System, context computing, environmental ICT, globalization.

\textbf{Introduction}

Conceptual modelling has been one of the essential academic subjects in the computer science area and includes highly significant topics not only in academic communities related to information systems, but also in the area of environmental and globalization studies.

Humankind faces the essential and indispensable mission; we must endeavor on a global scale to perpetually restore and improve our natural and social environments. One of the essential research activities in environmental study is conceptual modelling to express, share, analyze and visualize the environmental and social phenomena of various situations. It is essentially significant to create new conceptual modelling for making appropriate and urgent solutions to various environmental changes and social situations in the nature and society.

The nature and society are expecting our activities to cover environmental research areas, towards “Environmental Artificial Intelligence” with sensing-data processing, big-data analysis, machine learning, deep learning, spatio-temporal computing, GIS

\footnote{1Corresponding author.}
(Geographical Information Systems) processing and semantic computing. From the viewpoint of conceptual modelling, much research activity should focus on environmental issues for realizing sustainable nature and society.

To promote discussion on the future trends and challenges of conceptual modelling, we organized a panel session on “The Future of Conceptual Modelling” during the 29th International Conference on Information Modelling and Knowledge Bases (EJC2019). The panelists were Professor Yasushi Kiyoki (panel moderator and chair), Professor Bernhard Thalheim, Professor Marie Duži Professor Hannu Jaakkola and Professor Petchporn Chawakitchareon. In the panel session, we focused on discussions on new conceptual modelling towards Environmental Artificial Intelligence. Open questions to this aim are:

- How to give actual interpretations and understandings to the nature and societies?
  Nature cannot interpret the meanings of situations/phenomena by itself. Only the human can interpret the meaning of nature’s situation by our senses with brain.
- How to give meanings to environmental phenomena in computational processes?

The paper is based on the presentations of the panelists’ own viewpoints on the panel session topic. The paper is organized as follows. In Section 1, Professor Thalheim introduces model suites as a maintained collection of associated models. In Section 2, Professor Duži describes communication in a multi-agent world based on the Transparent Intensional Logic (TIL). In Section 3, Professor Jaakkola discusses on artificial intelligent (AI) in modelling landscape and technological changes in conceptual modelling. In Section 4, Professor Kiyoki and Professor Chawakitchareon present the 5D World Map System and its applications to global environmental engineering.

1. Model Suites as a Maintained Collection of Associated Models

Most disciplines simultaneously integrate a variety of models or a society of models. The theory of model suite has been developed in [1, 2, 3]. A model suite is essentially a well-associated and coherent ensemble of models. The models in a model suite coexist, co-evolve, and support solutions of subtasks. A model suite [3] consists:

- of set of models which are defined within a common language understanding on the basis of several modelling languages,
- of an association or collaboration schema among the models,
- of controllers that maintain consistency or coherence of the model suite,
- of application schemata for explicit maintenance and evolution of the model suite, and
- of tracers for the establishment of the coherence.

We observe three opportunities of building a model suite:

- **Horizontal model suites**: Horizontal model suites use the same level of abstraction and reflect different but integratable viewpoints or foci or scopes on the basis of different languages. Computer science and engineering modelling is mainly modelling at the same abstraction layer. The model ensemble used in UML (Unified Modelling Language) separates modelling into several concerns such as use case, classes, interaction, packaging, and collaboration. Model-based engineering can be based on a five-level model suite [4]. Business (layer) data models and conceptual (layer) data models are a typical example of a horizontal
model suite since the first one is typically business-oriented and the second one can be considered to be a refinement of the first one. The binding among these models is often implicit. We may however enhance the two models by a mapping that maps the first model to the second one. This mapping combines and harmonizes the different views that are used at the business user layer. A good example is a model suite consisting of a global conceptual model and a rather large number of conceptual viewpoints that reflect the needs of database system users.

- **Vertical model suites**: Vertical model suites combine models that have different abstraction levels, that vary in their level of detail and complexity, and that reflect different time and space abstractions. At the same time they are coupled through some kind of mapping mechanism and within a specific coupling style. Typical well-known vertical model suites are: (a) strategic, tactical and operational models used in business informatics, (b) OLTP-OLAP-Data_Mart decision support systems (OLTP= On-line Transaction Processing, OLAP= On-line Analytical Processing) and (c) database structure pattern. The OSI (Open Systems Interconnection Reference Model) layering model is a good example of a well-associated model suite. Another example is the vertical model suite consisting of models for micro-data, meso-data, and meta-data for instance in decision support systems. Data streaming data and big data applications might become another example of model suite support. These applications can use a data stream profile, a task model that allows to derive the data collection portfolio, a model of analysis-driven data exploration, and a model for data collectors according to the analysis space. A typical example of a sophisticated model suite is the model suite of the human heart. It consists of a 5-layer model of the heart. At the genes layer the networks of genes are given by molecular functions. Proteins form the elementary units, define the chemistry, and their composition. Cell structures are the basis elements for explanation of functions and key organizational unit with biological processes and pathway models. The tissue model describes the structure and function and with cellular components. The human heart as an element of the body is described by a system of myocardial activation.

- **Collection of models at some abstraction layers**: Model suites may also consist of associated models at various abstraction layers. These models are combinations of observations or thought models. A categorization of models at abstraction layers is given in Figure 1.

The third opportunity is less obvious. We thus consider the wide class of mental, semantical, and semantical scientific models as displayed in Figure 1. Mental models can be graphical (iconic, representation) ones. Semantical models are internal ones (perception, cogitative) or external ones (linguistic, meta-, meta-meta-, meta-meta-meta-models). Semantic scientific models are formal scientific, empirical scientific, technological, or praxeological ones [5]. The categorization may be extended to technical models (physico-technical, other technical or engineering models) which are not considered in [5].
The eight kinds of models in Figure 1 show a categorization of model kinds that generalizes the classification in [6, 7]. The classification has been developed after analysis of classical Greek thoughts about models. The main sources are Platon’s Politeia [8] and C. Lattmann’s [9] and our analysis of the analogies of sun, of the divided line, and of the cave. It allows a categorization into a combination of observation or empirical quantitative models and qualitative intelligible models of thought. Observations are some kind of reflection, i.e. shadows according to [8]. Intelligible models are based on different modes and qualities of reasoning first of all by one thinker (or a community of thinkers) after perceiving, believing, and digesting observations or beliefs. They are also based on proper and matured thoughts including rationality, formation and reconsideration of ideas. The eight models in Figure 1 are summarized as follows:

- **Impression models**: Humans are observing their environment and summarize their observations on things in models according to their interest, their tasks, their knowledge, and their profile.
- **Phenomena models**: Humans are considering their observations on their system environment, use pragmatic reasoning schemata, and internalize the entities around them. Phenomena models only represent what has been observed and not what is false, in each possibility.
- **Suggestion or experimental models**: Humans might have concluded that some of their thoughts so far and develop based on that suggestion models that might be the basis for a falsification process esp. on the basis of experiments.
- **Perception models** are reflecting human understanding on entities observed by a human and are based on the setting of a human, esp. the orientation and the priming. They combine mentalistic concepts that are intuitively formed according to some (empirical) human understanding.
- **Domain-situation models** represent the common worldview on systems that are commonly observed, are governed by shared knowledge and beliefs, and reflect a shared opinion within a community of practice. The modelling method is governed by communication and human interaction.
• **Hypothetical models** mediate between quantitative theories and qualitative theories. They are applied to hypothetical and investigative scenarios, should support causal reasoning as well as network-oriented reasoning, and are developed in an empirical framework.

• **Conceptual models** are models that are enhanced by concepts from concept spaces, are formulated in a language that allows well-structured formulations, are based on mental/perception/situation models with their embedded concept(ion)s, and are oriented on a modelling matrix that is commonly accepted [10].

• **Conception-based models**: Conceptions are consolidated systems of explanation. A model is enhanced by such systems of explanation and provides a generalizing and consolidated viewpoint.

Humans synchronously use a number of models according to their tasks, according to the model functions in task resolving scenarios, according to the collaboration needs, according to their actual context, and according to their partners. Models must not be coherent in the general case. Humans use various models for various purposes. The models might be contradicting and inconsistent as a model society. A model suite however must however be coherent. It is then concise and precise consolidation of all function-relevant structural and behavioral features of systems under investigation. It is represented in a number of predefined formats, e.g. modelling languages such as diagrammatic languages.

### 2. Communication in a Multi-Agent World

We learn, communicate and think by means of concepts; and regardless of the way in which the meaning of an expression is encoded, the meaning is a concept2. Yet in our background theory, i.e. Transparent Intensional Logic (TIL), we do not define concepts within the classical set-theoretical framework. Instead, we explicate concepts as abstract procedures that can be assigned to expressions as their structured meaning3. In particular, complex meanings, which structurally match complex expressions, are complex procedures whose parts are sub-procedures. The moral suggested here is this. Concepts are not flat sets that cannot be executed and lack a structure; rather, concepts are algorithmically structured abstract procedures. Unlike sets, concepts have constituent parts, i.e. sub-procedures that can be executed in order to arrive at the product the procedure is typed to produce. Not only particular parts of a concept matter, but also the way of combining these parts into one whole ‘instruction’ that can be followed, understood, executed, learnt, etc., matters4.

Having accepted semantic conception as described above, fundamental questions arise. How to reach those abstract procedures? How to examine their structure, how to derive what is entailed by them and not to derive what is not entailed? How to compute

---

2 To avoid misunderstanding, we also explicate the meaning of a sentence as the concept of a proposition denoted by the sentence. For more arguments in favour of structured procedural concepts as the means of our communication, see [11].

3 For details, see [12] and [13, §2.2].

4 This had been known already to Bernard Bolzano who criticized the classical Port-Royal school and the law of inverse proportion between the content (intension) and extent (extension) of a concept. The content itself does not determine the concept, the way of combining its parts matters; see [14: §120].
their products? There are two possibilities. Either not to do it and instead just specify axioms and rules of using them. Or, do it in a systematic way using a language fine-grained enough. In TIL, we vote for the second strategy. To this end, we apply the language of TIL λ-terms that denote these procedures and mirror their structure in an isomorphic way. To give just a hint of our conception, in Figure 2 there is a simple example of a valid argument formalized in TIL.

\[
\begin{align*}
\text{Tilman is seeking an abominable snowman} \\
\text{Tilman is seeking something abominable} \\
\lambda w \lambda t \left[ '\text{Seek}_{at} \ 'Tilman \ ['\text{Abominable} \ '\text{Snowman}] \right] \\
\lambda w \lambda t \exists x \left[ '\text{Seek}_{at} \ 'Tilman \ ['\text{Abominable} \ x] \right]
\end{align*}
\]

Figure 2. An example of a valid argument formalized in TIL.

All the entities of TIL ontology receive a type within a ramified hierarchy of types, which makes it possible to distinguish different levels of abstraction. In our example, the variable \(x\) must not range over individuals; this would turn logic into magic, and we would prove the existence of yetis. Instead, \(x\) ranges over properties of individuals. In addition, TIL is a typed, hyperintensional partial \(\lambda\)-calculus. Our procedures (concepts) are structured wholes that can occur in two fundamentally distinct modes, namely executed and displayed (for details, see [15]). When dealing with natural language, we need to operate not only within an extensional or intensional level, but also within hyperintensional level where substitution of analytically equivalent terms fails, because the very meaning procedure is displayed as the object of predication. Hyperintensional context is introduced inter alia by agents’ attitudes like knowing, believing, designing, seeking and finding, computing, and many others\(^5\). Thus we come to the second issue, which is a cogent argumentation in favour of multi-agent systems.

In [17] the authors introduce the system for disaster resilience. Protecting nature, environment and people against disasters is very important and primary goal, of course. Yet, there are critical situations such as an unexpected air disaster, natural disaster, traffic collapse, and so like, in which we can hardly do any more but to minimalize damages and harms by providing well-organized, professional disaster relief. In these situations, multi-agent systems are successfully applied.

Multi-agent systems are dynamic, distributed applications that run on many computers over the network. There can be thousands of agents who are active in their perceiving environment and acting in order to achieve their individual as well as collective goals. In general, there is no central dispatcher and the system is driven only by messaging. The agents communicate with their fellow agents by exchanging messages and they learn by experience. They are resource bounded, yet less-or-more intelligent and rational.

A multi-agent system should be designed in such a way that it is apt for handling critical situations where a centralised system is prone to a chaotic behaviour or even collapse. While behaviour of a centralised system heavily depends on the centralised

\(^{5}\) For a summary on hyperintensionality, see the Introduction to the special issue of Synthese [16].
control so that its fail causes the fall of the whole system, in a multi-agent system there are still some other agents who can act reasonably even in a very critical situation; in the worst case they can at least send a warning message to the public.

The theory formalizing reasoning of agents has to be able to ‘talk about’ and quantify over the objects of agents’ attitudes, i.e. structured meanings of the embedded clauses, iterate attitudes of distinct agents, express self-referential statements, respect different inferential abilities of resource bounded agents. While this is beyond the capacity of first-order logic systems, we have the theory at hand; it is Transparent Intensional Logic (TIL). Thus, the content of agents’ messages is formalized in TIL so that all the above issues are successfully dealt with. Each active agent has its own ontology and knowledge base. While an agent’s knowledge base usually contains dynamic empirical facts, formal ontology is a result of the conceptualization of a given domain. It contains definitions (i.e., complex concepts) of the most important entities, forms a conceptual hierarchy together with the most important attributes and relations between entities. Due to TIL ramified hierarchy of types, the agents can reason about concepts themselves, learn new compound concepts via refinement of less complex concepts and exhibit an adequate dynamic behaviour.

In the multi-agent and multi-cultural world procedurally structured concepts are central to our communication. We model such concepts as TIL procedures, coined constructions. Flexible systems that we need to deal with critical situations in our rapidly changing dynamic world are best modelled and implemented as multi-agent systems composed of autonomous, resource-bounded yet less or more rational agents who communicate by messaging. In our systems, the content of a message is formalized in terms of concepts, i.e. in the language of TIL constructions.

Acknowledgements. This research has been supported by the Grant Agency of the Czech Republic, project No. GA18-23891S “Hyperintensional Reasoning over Natural Language Texts”.

3. AI in Modelling Landscape - Technological Changes in Conceptual Modelling

The role of information modelling as a part of information systems (IS) development is to transfer human knowledge of the requirements of the system to the IS implementation. Along the development path, first step is to create a conceptual model. It represents the joint view of the interest groups of the IS’s data in a structured way. This structure is manipulated in the evolution path of the IS first to reflect to the needs of requirements engineering, further to include the architectural design related aspects, then technical aspects coming from technical design and finally the elements of implementation. During the whole life cycle it is question on the data model of the same IS, but from different points of view. The purpose of the models is to transfer IS related decisions through the life cycle from phase to phase and from interest group to interest group. In addition, it is the key issue for communication in IS development.

Data modelling is always done in its context. The context covers technologies available, tools, development processes – in general, a huge amount of environmental issues of modelling. If we look at the history of computing and IS development as a part of it, it is easy to see dramatical changes in technologies; IS development and data modelling at the principal level have remained the same, the purpose of modelling has remained the same, but technological progress has changed the environment of it. The key enabler in the progress of ICT (Information and Communication Technology) is the
improvement of VLSI technology. According to Moore’s Law [18] the packing density of VLSI circuits doubles every 15 months. It reflects directly to the processing capacity of computers (doubles in 18 months), memory capacity (doubles in 15 months), data transmission capacity (doubles in 20 months) and mass memories (capacity doubles in 18 months). If we compare the time of birth of systematic data modelling (from early 1960s) to the situation today, we have currently the computing environment having processing capacity of $2^{40}$, memory size of computers $2^{49}$, mass memory size $2^{40}$ and data transmission capacity $2^{36}$-times compared with the computing environment of early 1960s. In spite of this progress, it is acceptable to say that data modelling has more or less remained the same over the decades – or has it?

The progress introduced above is handled in more detail in [19] from the point of view of AI (Artificial Intelligence). AI has one of the key roles in the current era of data modelling. In [20] the era of systematic data modelling is divided (by binding it to the progress of database management) in four phases (quoted from the original article):

- **Phase I** (from roughly the 1960s to 1999) included the development of Database Management Systems (DBMS) known as hierarchical, inverted list, network, and during the 1990s, object-oriented Database Management Systems.
- **Phase II** (starting about 1990) relates to relational databases, SQL and SQL products (plus a few nonSQL products).
- **Phase III** (starting also around 1990 simultaneously to the Phase II) supported Online Analytical Processing (OLAP), along with specialized DBMSs.
- **Phase IV** (started 2008) introduced NoSQL and supported the use of Big Data, non-relational data, graphs, etc.

If we simply analyze the progress above, it is easy to notice that data modelling varies over the phases. Development tools are an essential part in IS development. In the development process, we aim to look at the system structure “through the glasses” of the tool. Because of that it is easy to agree that also tools create a part of the IS development context and aim to guide the data modelling.

One interesting view to the changes in data modelling can be found in the traditional classification of data related concepts. The DIKW pyramid\(^6\) (the good overview of it is available in [21]) represents structural and/or functional relationships between the concepts data, information, knowledge, and wisdom:

- **Data**: facts and figures relaying something in a non-organized way.
- **Information**: Contextualized, categorized, organized data.
- **Knowledge**: know-how, understanding, experienced, insight, intuition, contextualized information.
- **Wisdom**: Knowledge applied in action.

The hierarchy, in addition to help understanding about the role of the data in information systems, gives a view to the progress in data modelling. Without hesitance, it is possible to say that during the decades it is easy to see the transfer of the modelling target from lower towards upper levels. Nowadays, in the new era of AI, we should be able to model the connections between items of wisdom, instead of the traditional (data) concepts.

---

\(^6\) The origin of the DIKW Pyramid is not unanimously specified. The classification is handled in several articles; based on a simple literature review the author reviewed e.g.: [http://www.infogineering.net/data-information-knowledge.htm](http://www.infogineering.net/data-information-knowledge.htm); [http://www.knowledge-management-tools.net/knowledge-information-data.html](http://www.knowledge-management-tools.net/knowledge-information-data.html); [https://www.ontotext.com/knowledgehub/fundamentals/dikw-pyramid/](https://www.ontotext.com/knowledgehub/fundamentals/dikw-pyramid/)
What are the current trends in data and information systems modelling? This topic is handled in [22]. The author of the article points out the important role of cloud platforms, coupled with Big Data and IoT technologies powered by AI and Machine Learning (ML), providing professional means for non-professionals - citizen data analysts. The article gives nice scene to the data modelling of 2019. We list some of the most important aspects of it as follows:

- Tools embedding AI and ML change the modelling landscape. Built in intelligence in the tools decrease the amount of human work. Model analytics decreases the opportunity for human errors and increases the quality of models. Automatic model generation decreases the amount of human work.
- Gartner’s predicts 40 percent automation in data science tasks. In data analytics, the role of citizen data analysts is growing. It indicates the changing role of the experts. Two types of data models are needed: one for data professionals and one for citizen users to be used for quick solutions in a plug and play manner.
- There is transfer from problem specific to problem area specific instruments and towards framework dominance. It indicates higher abstraction of the IS models.
- The increasing role of Robotic Process Automation (RPA; Software Robotics) indicates the growing importance of business process modelling.
- Transfer out of relational databases, especially in new types of applications. New technologies - NoSQL databases, data lakes, algorithmic intelligence, self-describing data formats, standardized data models - initiate new challenges for and take place of data modelling. Cloud dominance affects in data structures.
- The growing role of "on-line" continuous data handled dynamically without knowing its structure in advance.
- Globalization of IS business indicates complexity in the Data Management ecosystem and unknown Data Governance issues. Data landscape becomes distributed, having a wide variety of data sources in applications. The importance of interoperability issues runs towards commonly used standardized data structures and models. Importance of interfaces and interface modelling will be an essential part of data modelling.

As a summary, it is easy to see the growth of modelling complexity, transfer of data (modelling) related tasks from professionals to end-users and AI to support human work. In basic level, data modelling remains as it has always been, but in practice, a lot of new challenges will appear.

4. Applications of AI in Global Environmental Engineering

4.1. Conceptual Modelling with Semantic Computing for Environmental Analysis

Humankind, the dominant species on Earth, faces the most essential and indispensable mission; we must endeavor on a global scale to perpetually restore and improve our natural and social environments. It is essentially significant to apply conceptual modelling and knowledge computing to global environment-analysis for finding out difference and diversity of nature and livings with a large amount of information resources in terms of global environments.

The 5D World Map (5DWM) System has introduced the concept of “SPA (Sensing, Processing and Analytical Actuation Functions)” for realizing a global environmental
The 5DWM system has been proposed by Kiyoki and Sasaki in [24, 25], and its architecture has been implemented as a multi-visualized and dynamic knowledge representation system. The 5DWM is a system for visualizing the data resources to the map, which can be analyzed with multi-dimensional axes. Environmental Knowledge Base creation with 5DWorld Map is implemented for sharing, analyzing and visualizing various information resources to the map, which can display and facilitate the comparisons in multidimensional axes.

This system realizes Physical-Cyber integration, as shown in Figure 3, to detect environmental phenomena with real data resources in a physical-space (real space), map them to the cyber-space to make knowledge bases and analytical computing, and actuate the computed results to the real space with visualization for expressing environmental phenomena, causalities and influences.

The 5D World Map System and its applications create new analytical circumstances with the SPA concept (Sensing, Processing and Analytical Actuation) for sharing, analyzing and visualizing natural and social environmental aspects, as shown in Figure 4. This system realizes “environmental analysis and situation-recognition” which will be essential for finding out solutions for global environmental issues. The 5D World Map System collects and facilitates many environmental information resources, which are characteristics of ocean species, disasters, water-quality and deforestation.

As conceptual modelling for making appropriate and urgent solutions to global environment changes in terms of short and long-term changes, “six functional-pillars” are essentially important with “environmental knowledge-base creation” for sharing, analyzing and visualizing various environmental phenomena and changes in a real world: (1) Cyber & Physical Space Integration, (2) SPA-function, (3) Spatio-Temporal computing, (4) Semantic computing, (5) World map-based visualization, and (6) Warning message propagation.

As an actual implementation of the SPA architecture, 5D World Map System Project has presented a new concept of “Water-quality Analysis Semantic-Space for Ocean-environment” for realizing global water-environmental analysis [26]. The semantic space and the computing method have been implemented with knowledge-base creation for water-quality-analysis sensors for analyzing and interpreting environmental phenomena and changes occurring in the oceans in the world. We have focused on sea-water quality data, as an experimental study for creating “Water-quality Analysis Semantic-Space for Ocean-environment” [26].

4.2 International Collaborative Research Activities with SPA-based 5D World Map System

The 5D World Map System focuses on sharing, analyzing and visualizing various environmental influences and changes caused by natural phenomena and disasters in global environments with “environmental multimedia data resources.” As a new meta-level system of international collaborative environment analysis, this system creates a remote, interactive and real-time academic-research exchange in global scopes and areas [24, 25]. Applications of 5DWorld Map for global sharing analysis of environmental situations and changes were studied as case studies in the international collaborative research activities with KEIO University (Japan) and Chulalongkorn University (Thailand).
An actual international and collaborative research project on the 5D World Map System started in 2011 with Chulalongkorn University [26, 27, 28]. This project focuses
on the global coral-analysis with multi-visualized knowledge sharing with 5D World Map System, applied to “coral-health-level analysis with images and water-quality data”.

This project [27] stared at Sichang Island, Thailand. Those coral information resources and research results have been mapped onto the 5D World Map system [24, 25]. Three species of corals at Sichang Island i.e. *Acropora sp.*, *Goniopora sp.* and *Pavona sp.* were subjected to a stress test with low salinity and normal salinity at concentration 10, 20 and 30 psu, respectively. Under water photographs and eye observation of coral activity were recorded at 12, 24 and 48 hours. The entropy or surface roughness and percent polyp activity were analyzed with comparison to eye observation of coral activity. The experiment was carried out under continuous water temperature and underwater light intensity controlled. The results indicated that “Healthy” entropy values for *Acropora sp.* are 1.57-1.62 and for *Goniopora sp.* are 4.26-4.46. In contrast, for *Pavona sp.*, short polyp coral, there was no “Healthy” entropy value resulted from any photographic assessment in this study. The “Healthy” value of *Acropora sp.* evaluated from percent active polyp was more than 52.4.

This collaborative project focuses on the effects of temperature and ammonia to coral health levels on *Acropora sp.*, *Turbinaria sp.*, *Porites sp.* by coral health levels evaluation with Coral Health Chart [28]. It was a standardized color reference card, which is a flexible tool that anyone can use for rapid, wide-area assessment of changing coral condition. The acute toxicity of ammonia concentration that affects to bleach coral more than 50% (50% Lethal Concentration: LC50) was calculated by Probit analysis and coral bleaching analysis by polyp image analysis.

In addition to coral-analysis in this place, this project integrated a global sharing analysis and visualization of water quality analysis [29], with 5D World Map system. This integration is a typical and advantageous result to be realized with 5D World Map system, as typical and effective integration between different subjects with high relationships each other. The data resources in this research were collected from Sichang Island, Chonburi province, Thailand during 1990 to 2002. Six input parameters of water quality i.e. chlorophyll a, ammonia, nitrite, nitrate, phosphate and silicate were collected and displayed in 5D World Map system. The total location-sites were 21 stations, which situated around Sichang Island. All data of water quality were added and displayed with 5D World Map system in order to visualize and share the water quality from 1990 to 2002. Our results showed that 5D World Map system integrates environmental analysis with the coral-analysis subject related to the coral health level. We apply the dynamic evaluation and mapping functions of multiple views of temporal-spatial metrics, and integrate the results of semantic evaluation to analyze environmental multimedia information resources. 5D World Map System for world-wide viewing of global environmental analysis with coral-analysis and water quality around Sichang Island in Thailand was reported in this study.

To conclude, we have introduced a conceptual modelling methodology for realizing global environmental analysis with “5D World Map System.” This methodology is essential to make appropriate and urgent solutions to global environment changes in terms of short and long-term changes. We have applied this methodology to “international and collaborative environmental-system research and education” as a new platform of environmental computing. This platform realizes remote, interactive and real-time academic research exchanging for international and collaborative research activities, and this system is currently utilized as an international and environmental research platform. This system is also expected to create several new original approaches to global environmental-knowledge sharing, analysis and visualization with “spatio-
temporal & semantic computing.” This system concept will be a basic structure to create new international and collaborative research and education for making important solutions and knowledge sharing on global environmental issues in the world-wide scope.

Acknowledgements: We would like to appreciate Dr. Xing Chen, Dr. Shiori Sasaki, Dr. Asako Uraki, Dr. Chalisa Veesommai and Dr. Irene Eryln Wina Rachmawan, “5D World Map System Project” members and Dr. Sompop Rungsupa for their significant discussions and experimental studies.

References


Models for Communication, Understanding, Search, and Analysis

Bernhard Thalheim

Christian-Albrechts University at Kiel, Dept. of Computer Science, D-24098 Kiel
thalheim@is.informatik.uni-kiel.de
http://www.is.informatik.uni-kiel.de/~thalheim

Abstract. Models are one of the universal instruments of humans. They are equally important as languages. Models often use languages for their representation. Other models are conscious, subconscious or preconscious and have no proper language representation. The wide use in all kinds of human activities allows to distinguish different kinds of models in dependence on their utilisation scenarios. In this keynote we consider only four specific utilisation scenarios for models. We show that these scenarios can be properly supported by a number of model construction conceptions. The development of proper and well-applicable models can be governed by various methodologies in dependence on the specific objectives and aims of model utilisation.

1 Introduction

Models are widely used in life, technology and sciences. Their development is still a mastership of an artisan and not yet systematically guided and managed. The main advantage of model-based reasoning is based on two properties of models: they are focused on the issue under consideration and are thus far simpler than the application world and they are reliable instruments since both the problem and the solution to the problem can be expressed by means of the model due to its dependability. Models must be sufficiently comprehensive for the representation of the domain under consideration, efficient for the solution computation of problems, accurate at least within the scope, and must function within an application scenario.

The Notion of Model

Let us first briefly repeat our approach to the notion of model:

A model is a well-formed, adequate, and dependable instrument that represents origins and that functions in utilisation scenarios. [6, 24, 25]

Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice within some context and correspond to the functions that a model fulfills in utilisation scenarios.
The model should be well-formed according to some well-formedness criterion. As an instrument or more specifically an artifact a model comes with its background, e.g. paradigms, assumptions, postulates, language, thought community, etc. The background is often given only in an implicit form. The background is often implicit and hidden.

A well-formed instrument is adequate for a collection of origins if it is analogous to the origins to be represented according to some analogy criterion, it is more focused (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled, and it sufficiently satisfies its purpose.

Well-formedness enables an instrument to be justified by an empirical corroboration according to its objectives, by rational coherence and conformity explicitly stated through conformity formulas or statements, by falsifiability or validation, and by stability and plasticity within a collection of origins.

The instrument is sufficient by its quality characterisation for internal quality, external quality and quality in use or through quality characteristics [23] such as correctness, generality, usefulness, comprehensibility, parsimony, robustness, novelty etc. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

A well-formed instrument is called dependable if it is sufficient and is justified for some of the justification properties and some of the sufficiency characteristics.

Model Deployment Scenarios are Multi-Facetted

The model notion can be seen as an initialisation for more concrete notions. We observe that model utilisation follows mainly four different kinds of scenarios (see Figure 1). The four scenarios do not occur in its pure and undiffused form they are interleaved. We can however distinguish between:

**Problem solving scenarios:** Problem solving is a well investigated and well organised scenario (see, for instance, [1, 8]). It is based on (1) a problem space that allows to specify some problem in an application in an invariant form and (2) a solution space that faithfully allows to back-propagate the solution to the application. We may distinguish three specific scenarios: perception & utilisation; understanding & sense-making, and making your own.

**Engineering scenarios:** Models are widely used in engineering. They are also one of the main instruments in software and information systems development, especially for system construction scenario. We may distinguish three specific scenarios depending on the level of sophistication: direct application:, managed application, and application according to well-understood technology.

**Science scenarios:** Sciences have developed a number the distinctive form in which a scenario is organised. Sciences make wide use of mathematical modelling. The methodology of often based on specific moulds that are commonly accepted in the disciplinary community of practice, e.g. [1]. We may distinguish three specific scenarios: comprehension, computation and automatic detection for instance in data science, and intellectual adsorption.
**Social scenarios:** Social scenarios are less investigated although cognitive linguistics, visualisation approaches, and communication research have contributed a lot. Social models might be used for the development of an understanding of the environment, for agreement on behavioural and cultural pattern, for consensus development, and for social education.

We may distinguish three specific scenarios: development of social acceptance, internalisation & emotional organisation, and concordance & judgement.

**Fig. 1.** The four aspects of model scenarios: problem solving, engineering, science, social scenarios. Each of the scenarios combines initialisation by exploring the landscape, strategy, tactics, operational, and delivery layers. Each of these layers adds quality characteristics and specific activities to the previous layer in dependence on the aspects considered.

The notion of model mainly reflects the initialisation or landscape layer. Depending on the needs and demands to model utilisation we may distinguish various layers from initialisation towards delivery. The strategy, tactics, operational, and delivery layers are essentially refinements and extensions of the initialisation. The dependability and especially the sufficiency are based on other criteria while the landscape layer is permanent for all models due to the consideration of the concern, the issue, and the specific adaptation to the community of practice. The strategy layer is governed by the context (e.g. the discipline) and the mould for model utilisation, and the matrix (including methodologies and commonly
accepted approaches to modelling). The tactics layer depends on the settlement of the strategy and initialisation layers considers the well-acknowledged experience (e.g. generic approaches), the school of thought or more generally the background, and the framing of the modelling. Which origin(al)s are reflected and which are of less importance is determined in the operational layer that orients on the design and on mastering the modelling process. Finally the model is delivered and form for its application in scenarios that are considered. We thus observe various specific quality characteristics for each of these aspects and layers.

Do We Need a Science of Models and Modelling?

Since everybody is using models and has developed a specific approach to models and modelling within the tasks to be solved, it seems that the answer is “no”. From the other side we deeply depend on decisions and understandings that are based on models. We thus might ask a number of questions ourselves. Can models be misleading, wrong, or indoctrinating? Astrophysics uses a Standard Model that has not been essentially changed during the last half century. Shall we revise this model? When? What was really wrong with the previous models? Many sciences use modelling languages in religious manner, e.g. think about UML and other language wars. What is the potential and capacity of a modelling languages? What not? What are their restrictions and hidden assumptions? Why climate models have been deeply changing and gave opposite results compared to the previous ones? Why we should limit our research on impacts of substances to a singleton substance? Many sciences use modelling languages in religious manner, e.g. think about UML and other language wars. What is the potential and capacity of a modelling languages? What not? What are their restrictions and hidden assumptions? Why climate models have been deeply changing and gave opposite results compared to the previous ones? Why we should limit our research on impacts of substances to a singleton substance? What is the impact of engineering in this case? What has been wrong with the two models on post-evolution of open cool mines after deployment in Germany which led to the decision that revegetation is far better than water flooding? Why was iron manuring a disaster decision for the Humboldt stream ocean engineering? What will be the impact of the IPCC/NGO/EDF/TWAS proposal for Solar Radiation Management (SRM) for substantial stratosphere obscuration for some centuries on the basis of reflection aerosols (on silver, sulfate, photophoretic etc. basis)? Why reasoning on metaphors as annotations to models may mislead? Are “all models wrong”?

Developing a science of models and modelling would allow us to answer questions like the following one: What is a model in which science under which conditions for whom for which usage at a given time frame? What are necessary and sufficient criteria for an artefact to become a model? What is the difference between models and not-yet-models or pre-models? What is not yet a model? How are models definable in sciences, engineering, culture, ...? Under which conditions we can rely on and believe in models? Logical reasoning: which calculus? Similarity, regularity, fruitfulness, simplicity, what else (Carnap)? Treatment, development, deployment of models: is there something general in common? Models should be useful! What does it mean? Is there any handling of usage,

---

1 “All models are wrong. ... Obtain a ‘correct’ one. ... Alert to what is importantly wrong.” [2] We claim: Models might be ‘wrong’. But they are useful.
usefulness, and utility? What is the difference between an object, a model, and a pre-model? What might be then wrong with mathematical models? What is the problem in digging results through data mining methods?

**The Storyline of this Paper**

Models are the first reasoning and comprehension instruments of humans. Later other instruments are developed. The main one is language. Models then often become language-based if they have to be used for collaboration. Others will remain to be conscious, preconscious or subconscious. Based on the clarification of the given notion of model and a clarification of the model-being we explore in this paper what are the constituents of models, how models are composed, and what are conceptions for model constructions. Since models are used in scenarios and should function sufficiently well in these scenarios we start with an exploration of specific nature of models in four scenarios. We are not presenting all details for a theory of models.

2 Case Study on some Scenarios for Model Utilisation

Models are used in various utilisation scenarios such as construction of systems, verification, optimization, explanation, and documentation. In these scenarios they function as instruments and thus satisfy a number of properties [7, 26–28].

**Models for Communication**

The model is used for exchange of meanings through a common understanding of notations, signs and symbols within an application area. It can also be used in a back-and-forth process in which interested parties with different interests find a way to reconcile or compromise to come up with an agreement.

The model has several functions in this scenario: (personal/public/group) recorder of settled or arranged issues, transmitter of information, dialogue service, and pre-binding. Users act in the speaker, hearer, or digest mode.

The communication act is composed of six sub-activities: derive for communication, transfer, receive, recognise and filter against knowledge and experience, understand, and integrate. We may distinguish two models at the speaker side and six models at the hearer side: speaker’s extracted model for transfer, transferred model for both, hearer’s received model, hearer’s understanding and recognition model, hearer’s filtered model, hearer’s understood model, and hearer’s integration model. These models form some kind of a model ensemble. Some are extensions or detailing ones; others are zooming ones. Communication is based on some common understanding or at least on transformation of one model to another one.

---

2 Collections of papers which are used as background for this paper is downloadable via Research Gate. Notions and definitions we used can be fetched there.
Models for Understanding

Models may be used for understanding the conceptions behind. For instance, conceptualisation is typically shuffled with discovery of phenomena of interest, analysis of main constructs and focus on relevant aspects within the application area. The specification incorporates concepts injected from the application domain.

The function of a model within these scenario is *semantification* or *meaning association* by means of concepts or conceptions. The model becomes enhanced what allows to regard the meaning in the concept.

Models tacitly integrate knowledge and culture of design, of well-forming and well-underpinning of such models and of experience gained so far, e.g. meta-artifacts, pattern and reference models. This experience and knowledge is continuously enhanced during development and after evaluation of constructs.

Models are functioning for elaboration, exploration, detection, and acquisition of tacit knowledge behind the origins which might be products, theories, or engineering activities. They allow to understand what is behind drawn curtain.

Models for Search

Users often face the problem that their mental model and their fact space are insufficient to answer more complex questions [12]. Therefore, they seek information in their environment, e.g. from systems that are available. Information is data that have been shaped into a form that is meaningful and useful for human beings. Information consists of data that are represented in form that is useful and significant for a group of humans. This information search is based on their on the *information need*, i.e. a perceived lack of some information that is desirable or useful. The information is used to derive the current *information demand*, i.e. information that is missing, unknown, necessary for task completion, and directly requested. Is is thus related to the task portfolio under consideration and to the intents.

Search is one of the most common facilities in daily life, engineering, and science. It requires to examine the data and information on hand and to carefully look at or through or into the data and the information.

There is a large variety of information search [5] such as:
1. querying data sets (by providing query expressions in the informed search approach),
2. seeking for information on data (by browsing, understanding and compiling),
3. questing data formally (by providing appropriate search terms during step-wise refinement),
4. ferreting out necessary data (by discovering the information requested by searching out or browsing through the data),
5. searching by associations and drilling down (by appropriate refinement of the search terms),
6. casting about and digging into the data (with a transformation of the query and the data to a common form), and
7. zapping through data sets (by jumping through provided data, e.g., by partially uninformed search).
Models for Analysis

Data analysis, data mining or general analysis combines engineering and (systematic) mathematical problem solving [20]. The model development process combines problem specification and setting with formulation of the analysis tasks by means of macro-models, integration of generic models, selection of the analysis strategy and tactics based on methodology models, models for preparation of the analysis space, and model combination approaches for development of the final model society as the analysis result [16, 14]. The typical process model that governs the analysis process is based on a layering approach, e.g. initial setting, strategy, tactics with generic (or general parameterised models), analysis initialisation, puzzling the analysis results, and final compilation. It is similar to experiment planning in Natural Sciences. The analysis puzzling may follow a number of specific scenarios such as pipe scenarios [19].

3 Model Conceptions for These Scenarios

It seems that these scenarios require completely different kinds of models. This is however often not the case. We can develop stereotypes which are going to be refined to pattern and later to templates as the basis for model development. We demonstrate for the four scenarios (communication, understanding, search, analysis) how models can be composed in a specific form and which kind of support we need for model-backed collaboration.

Deep Models

A typical model consists of a normal (or surface) sub-model and of deep (implicit, supplanted) sub-models which represent the disciplinary assumptions, the background, and the context. The deep models are the intrinsic components of the model. Conceptualisation might be four-dimensional: sign, social embedding, context, and meaning spaces. The deep model is relatively stable. In science and engineering it forms the disciplinary background. It is often assumed without mentioning it. For instance, database modelling uses the paradigms, postulates, assumptions, commonsense, restrictions, theories, culture, foundations, practices, and languages as carrier within the given thought community and thought style, methodology, pattern, and routines. This background is assumed as being unquestionable given. The normal model mainly represents those origins that are really of interest.

The deep model combines the unchangeable part of a model and is determined by (i) the grounding for modelling (paradigms, postulates, restrictions, theories, culture, foundations, conventions, authorities), (ii) the outer directives (context and community of practice), and (iii) the basis (assumptions, general concept space, practices, language as carrier, thought community and thought style, methodology, pattern, routines, commonsense) of modelling. The deep model can be dependent on mould principles such as the conceptualisation principle [9].
A typical set of deep models are (the models and) foundations behind the origins which are inherited by the models of those origins. Also modelling languages have there specific deep parts. As well as methodologies or more generally moulds of model utilisation stories.

**Model Capsules**

Model capsules follow a global-as-design approach (see Figure 2). A model has a number of sub-models that can be used for exchange in collaboration or communication scenarios. A model capsule consists of a main model and exchange sub-models. Model capsules are stored and managed by their owners. Exchange sub-models are either derived from the main model in dependence on the viewpoint, on foci and scales, on scope, on aspects and on purposes of partners or are sub-models provided by partners and transformed according to the main model. A sub-model might be used as an export sub-model (e.g. $A_{4,E}$) that is delivered to the partner on the basis of the import sub-model (e.g. $B_{4,I}$). The sub-models received are typically transformed. We thus use the E(xtract)T(ransform)L(oad) paradigm where extraction and loading is dependent on the language of the sending or receiving model and where transformation allows adaptation of the export sub-model to the import sub-model.

![Fig. 2. Exchange on the basis of model capsules with sub-models in model-based ETL-oriented communication scenarios](image)

**Model Suites**

Most disciplines simultaneously integrate a variety of models or a society of models, e.g. [3, 11]. The four aspects in Figure 1 are often given in a separate form as an integrated society of models. Models developed vary in their scopes, aspects and facets they represent and their abstraction.
A typical case are the four aspects that might coexist within a complex model. For instance, models in Egyptology [4] can be considered have four aspects where each of the aspects has its specific model. The entire model is an integrated combination of (1,2) signs in textual representation and an extending it hieroglyph form (both as representation), (3) interpretation pattern (as the foundation and integration into the thoughts), (4) social determination (as the social aspect), and (5) a context or realisation models into which the model is embedded. The co-design framework for information systems development (integrated design of structuring, functionality, interaction, and distribution) uses four different interrelated and interoperating modelling languages. These modelling languages are at the same level of abstraction and may be combined with additional orientation on usage (as a social component, e.g. represented by storyboards [21]). In this case, the foundational aspect is hidden within the modelling language and within the origins of the models, for instance in the conceptualisation. Following the four aspects in Figure 1, we derive now models that consider one, two, three, or all four aspects (Figure 3).

**Fig. 3.** The four aspect model suite and the corresponding planes for the layers within a model. Activities are governed at each plane by the WHAT I <actually_consider> as main activity.

A *model suite* consists of set of models \( \{M_1, \ldots, M_n\} \), of an association or collaboration schema among the models, of controllers that maintain consistency or coherence of the model suite, of application schemata for explicit maintenance.

---

3 The rich body of knowledge resulted in [22] or the encyclopedia with [17].
and evolution of the model suite, and of tracers for the establishment of the coherence.

Model suites typically follow a local-as-design paradigm of modelling, i.e. there must not exist a global model which combines all models. In some cases we might however construct the global model as a model that is derived from the models in a model suite. The two approaches to model-based exchange can be combined. A model capsule can be horizontally bound to another capsule within a horizontal model suite or vertically associated to other model capsules. Model capsules are handled locally by members in a team. For instance, model capsules are based on models A and B that use corresponding scientific disciplines and corresponding theories as a part of their background. The models have three derived exchange sub-models that are exported to the other capsule and that are integrated into the model in such a way that the imported sub-model can be reflected by the model of the capsule.

Model Scenes

Model scenes for the development process may be specified in a similar way as storyboarding [21]. A scene is used by members of the community of practice, follows a certain modelling mould, considers a typical ensemble of origin(al)s, inherits certain stereotypes and pattern, is embedded into a context and the tasks, and uses the deep model as the background for model development. These parameters govern and thus control the scene. The developer or modeller is involved into this scene. The input for the scene is the current model, the specific properties of the ensemble of origin(al)s, and especially the experience gained so far. This experience may be collected in a library or generalised to generic models. The output is an enhanced model. We notice that model utilisation scenes can be specified in a similar way.

Figure 4 displays the embedding of a model scene into the model mould or more specifically into the methodology as a macro-model for development.

A model scene is an element of a model story. We imagine that the story can be represented as a graph. A model scene considers an actual or normal model and at the same time the desired embedding into the deep model. The scene is relevant for the community of practice. The model should be accepted by this community. The model scene also embeds the deep model. The scene has its cargo [18], i.e. its mission, determination, meaning, and specific identity. The cargo allows to determine the utility that the model gained so far.

Model Stories

Model development and utilisation can be described as a graph of scenes. Let us consider the model development for search scenarios [12, 13] in Figure 5. This story can be used for derivation of a waterfall-like approach in Figure 6. We start with initialisation of the search landscape. The result is a search guideline (or search activity meta-model). The information demand is transferred to a search question. The search strategy is configured out of the seven kinds of search. The
result is a macro-search model. Selection of the search pattern depends on system information and on the data that is available. The result is a search meso-model, for instance, question-answer forms. Finally we may derive a model on the basis of the data. We might also reconsider the intermediate results and preview or prefetch the potential solutions.

This story is similar to data mining stories [13, 15]. Data mining uses macro-models as methodological foundation. Frameworks for data mining start with
problem specification and setting, continue with formulation of data mining tasks by means of macro-models, reuse generic models according to required adequacy and dependability, next then select appropriate algorithms according to the capacity and potential of algorithms, prepare furthermore the data mining as a process, and finally apply this process. The data mining mould can be supported by controllers and selectors.

**Spaces for Models**

Figures 1 and 3 use six planes for detailing models. Each of the planes has its specific quality requirements, support tools, and tasks. At the landscape layer we determine the orientation of the model that should be developed, its problem space, its focus and scope, its integration into the value chain of the application (domain), and its stakeholder from the community of practice with their specific interests and their responsibilities. We rely on mental and codified concepts which are often provided by the world of the origins that a model should properly reflect. The strategic layer adds to this the ‘normal way’ of development for utilisation of models as methodologies or mould, the embedding into the context and especially the infrastructure, the disciplinary school of thought or more generally the background of the model. The tactics plane embeds the foundations into the modelling process, for instance, by deep model incorporation. It also allows to sketch and to configure the model. The operational plane orients on the formation of the model and the adaptation to the relevant origin(al)s that are going to represented by the model(s). The main issue for the delivery plane is the design of the model(s). The last plane orientates utilisation of the model(s) that have been developed. This outer plane might also be structured according to the added value that a model has for the utilisation scenario. Each plane allows to evaluate the model according to quality characteristics used in the sufficiency portfolio.

The model planes have their own workspace and workplaces which are part of the infrastructure for modelling and utilisation.

**4 Model Development**

**Model Development Story**

The modelling story consists of the development story and of the utilisation story. The model development story integrates activities like

1. a selection and construction of an appropriate model according to the function of the model and depending on the task and on the properties we are targeting as well as depending on the context of the intended outcome and thus of the language appropriate for the outcome,
2. a workmanship on the model for detection of additional information about the original and of improved model,
3. an analogy conclusion or other derivations on the model and its relationship to the application world, and
4. a preparation of the model for its use in systems, for future evolution, and for change.

Model utilisation additionally uses assured elementary deployment that includes testing and model detailing and improvement. It may be extended to paradigmatic and systematic recapitulation due to deficiencies from rational and empirical perspectives by the way(s) incommensurability to be resolved. Model deployment also orients on the added value in dependence on the model function in given scenarios. A typical model mould is the mathematics approach to modelling based on (1) exploration of the problem situation, (2) development of an adequate and dependable model, (3) transformation of the first model to a mathematical one that is invariant for the problem formulation and is faithful for the solution inverse mapping to the problem domain, (4) mathematical problem solution, (5) mathematical verification of the solution and validation in the problem domain, and (6) evaluation of the solution in the problem domain [1].

Greenfield Development

Although development from scratch is rather seldom in practice and daily life we will start with the activities for model development. These activities can be organised in an explorative, iterative, or sequential order in the way depicted in Figure 3. We can separate activities into:

(1) Exploration of the origin(al)s what results in a well-understood domain-situation and perception models: The origin(al)s will be disassembled into a collection of units. We ensemble (or monstrate) and manifest the insight gained so far in a domain-situation model and develop nominal or perception models for the community of practice. It is based on a plausible model proposition, on a selection of appropriate language and of theories, on generic models, and on commonsense structuring.

(2) Model amalgamation and adduction is going to result in a plausible model proposition according to the selected aspects of the four aspects. Amalgamation and adduction are based on an appropriate empirical investigation on origin(al)s, on agreed consensus in the school of thought within the community of practice, on hypothetical reasoning, and on investigative design.

(3) Final model formulation results in an adequate and dependable model that will properly function in the given scenarios. We use appropriate depictions for a viable but incomplete model formulation, extend it by corroborated refinements and modifications, and rationally extrapolate the model in dependence on the given ensemble of origin(al)s. In order to guarantee sufficiency of the model, we assess by elementary and prototypical deployment for proper structuring and dependability, within the application domain, within the boundaries of the background, and within the meta-model or mould for model organisation.

---

4 As a generalisation, reconsideration of [10]
A number of moulds can be used for refinement of this development meta-model such as agile or experience-backed methodologies. Modelling experience knowledge development might be collected in a later rigor cycle (see design science, for instance, [29]). Model development is an engineering activity and thus tolerates insufficiencies and deficiencies outside the quality requirements. A model must not be true. It must only be sufficient and justified. It can be imperfect.

The result of development can also be a model suite or a model capsule. For instance, information system modelling results in a conceptual structure model, a conceptual functionality model, a logical structure and functionality model, and a physical structure and functionality model. It starts with a business data and process viewpoint model.

Model development can be based on a strictly layered approach in Figure 6 that follows the mould in Figure 5 based on planes in Figure 3.

![Fig. 6. Model suite development mould for some of the four aspects in Figure 1](image)

**Brownfield Development**

Modelling by starting from scratch (‘greenfield’) must be extended by methods for ‘brownfield’ development that reuses and re-engineers models for legacy systems and within modernisation, evolution, and migration strategies. The corresponding model already exists and must be revised. It may also need a revision of its deep sub-model, its basis and grounding, and its ensemble of original(s). All activities used for greenfield development might be reconsidered and revised.

## 5 Conclusion

Models are widely used and therefore many-faceted, many-functioning, many-dimensional in their deployment, and . Based on a notion of model developed
Models for Various Scenarios

at Kiel university in a group of more than 40 chairs from almost all faculties, we explore now the ingredients of models. The model-being has at least four dimensions which can be grouped into four aspects: representation of origins and their specific properties, providing essential foundations and thus sense-making of origins, relishing and glorifying models as things for interaction and social collaboration, and blueprint for realisation and constructions within a context. This four-aspect consideration directly governs us during introduction of model suites as a model or model capsules. The utilisation scenario and the function of a given model (suite) determine which of the four aspects are represented by a normal model and which aspects are entirely encapsulated in the deep model.

Models are embedded into their life, disciplinary, and technical environment, and their culture. They reuse intentionally or edified (or enlightened) existing sub-models, pre-model, reference model, or generic models. A model typically combines an intrinsic sub-model and an extrinsic extrinsic sub-model. The first sub-model forms the deep model. For instance, database modelling is based on a good number of hidden postulates, paradigms, and assumptions.

The model-being is thus dependent on the scenarios in which models should function properly. We considered here four central scenarios in which models are widely used: communication, understanding, search, and analysis. These four utilisation scenarios can be supported by specific stereotypes of models which model assembling and construction allows a layered mastering of models. The mastering studio has its workspace and its workplace, i.e. in general space for models.

References

Between Natural and Human Sciences: On the Role and Character of Theory in Socio-Environmental Archaeology

VPJ Arponen, Sonja Grimm, Lutz Käppel, Konrad Ott, Bernhard Thalheim, Yannic Kropp, Kim Kittig, Johanna Brinkmann, Artur Ribeiro

Abstract

Prominent voices in archaeology have expressed deep skepticism about the role of theory in archaeology while with new, exciting methods at its disposal, archaeological science is occasionally perceived as not needing theory at all. This paper reflects upon the debate about theory in archaeology to arrive at a robust but critical middle range concept of the role and character of theory in socio-environmental archaeology. It is argued that archaeology is a data-based science and, consequently, in order for theory to be meaningful in socio-environmental archaeology, theory ought explicitly aim to make its qualitative concepts quantitative to establish a clear relation to data and its interpretation. On the turn side, theory plays an important role critically reflecting upon the use of concepts in archaeological understanding and explanation, as well as their origins in particular paradigms, as examples of which certain debates in scientific archaeology are discussed (aDNA and migration, evolutionism). We argue that such model would serve archaeology far more than a naïve dismissal of theory on the one hand and the continued production of “high” theory in absence of operationalization on the other.

Introduction

Archaeology is a historical discipline between the natural sciences and the humanities. There is a more scientific and positivistic side to archaeology, and there is a more theoretical and speculative side (Sørensen, 2017; Killich 2015; Kristiansen 2014). The division echoes the classic “two cultures” argument made by C.P. Snow (1959). At least since the emergence of postprocessual archaeology, the relationship between archaeological science and archaeological theory has been tense. Our concern in this essay is to attempt to locate the crux of the tension and then attempt an ecumenical but critical account of the role of theory in archaeological science. We write in the context of a larger, interdisciplinary, socio-environmental archaeological research effort the fruits of which this special issue displays.

On the scientific (positivistic, empirical, material) side of archaeology, we find research and data mining at different sites, we find refined dating methods, restoration of fragmentary remnants, inventories, collections, and archives. Remnants are excavated, dated, physically analyzed, and stored. This side represents the actual and solid disciplinary work proceeding according to established as well as innovative scientific methods. It constitutes a growing database. On this what one might call positivist side, we see firm and impressive results and we register slow but steady progress over decades, from DNA and isotope analysis to digitalization. One can take this dimension as the hard core of archaeology (Kristiansen, 2014). The history of archaeology, then, can be written as the development “of techniques of recovery and material analysis” (Ion & Barrett, 2016: 133). In this sense, sequencing of ancient DNA, pollen analysis, and isotope analysis would be paradigm examples of scientific progress.

Given the above, it is not obviously wrong to define the epistemic self-understanding of archaeology in a prudent, modest, and enlightened, positivist, research-oriented way. This definition will entail some skepticism of “lofty” or “mere” theoretical speculations.
In works such as John Bintliff’s provocative “The Death of Archaeological Theory” (2011, see also Bintliff, 2015), the problematic nature of the relationship of scientific archaeology to theory was discussed. Bintliff argued that

Published papers increasingly begin with pages of scholastic citation to works of theory, followed by applications to archaeological data which rely more on repeated reference to the favoured approach than providing convincing matching of concepts to recovered material evidence. (Bintliff, 2011: 9)

In other words, theory appears to serve a lofty role detached from the archaeological practice while the real core of the archaeological practice is to be found in rigorous empirical work (see also Johnson, 2006). Bintliff’s views were echoed by Matthew H. Johnson who wrote that

> There is, to put it very simply, a disjuncture between what we say we do as ‘archaeological theorists’ and what we actually do as archaeologists […] The case studies offered in support of a particular theoretical position frequently do not match up to the claims made about them in the preceding theoretical excursus. (Johnson, 2006: 118, 119)

Arguably, the actual targets of the “death of theory” charge may not so much be theory in the broadest sense—for often, the critics themselves are prolific theorists themselves—but rather some particular instances of (postprocessual) theory.

Nonetheless, In general, still, well-known practitioners of scientifically but also social theoretically informed archaeology continue to be unimpressed by the fruits of the latest theoretical work citing a “lack of interpretive implementation and progress” (see e.g. Kristiansen, 2017). Elsewhere Kristiansen observed in the literature a wider “critical stance against a previously predominant postmodern/post-processual hegemony, and the reintroduction of a revised modern/processual approach” in archaeology (Kristiansen, 2014: 11). Similarly, in reference to the concept of agency as it came to archaeology from the works of Pierre Boudieu and Anthony Giddens some decades ago, Dobres and Robb (2000: 4) argued that “agency in archaeology is not a theoretically sophisticated paradigm, but rather a lingua franca—an ambiguous platitude meaning everything and nothing”.

A closely related critique of theory is that where the import of a theoretical framework with regard to archaeological interpretation is made explicit, the results regularly fail to convey anything substantially new about the research object at hand. Such a morale arises, for example, from John Barrett’s (2014: 68-71) discussion of the inanimate agency thesis in new materialist archaeology, a fairly fresh entrant to the archaeological theoretical scene (Harris & Cipolla 2017; not to be confused with the agency theory mentioned above). In closer scrutiny, says Barrett, either the inanimate agency thesis is vague in its statements as to wherein causality resides in an assemblage of human and non-human things; or it implausibly proposes that causal agent to be able to be a material thing; or, finally, that what the inanimate agency thesis really is proposing is the kind of holism that most would accept anyway: “Archaeologists have long characterized the conditions of the past as the operation of a complex system of relationships between different kinds of component” (Barrett, 2014: 65).

Where, then, does all this leave theory? How, if at all, can it be combined with scientific archaeology? What is the role of theory in archaeology? Despite these difficult questions, a longer standing view is “that everything archaeologists do is infused by theory (much of it, regrettably, still implicit)” (Schiffer, 1988: 461). Similar observations have been made at various junctures over the decades such as Alison Wylie (2002: 7) here:
observations are theory-laden and richly dependent on extended networks of theoretical claims and assumptions ... that include generalizations about observables as well as claims about unobservable dimensions of the reality under study ... These constitute a conceptual framework without which observations have no meaning or evidential import—indeed, without which they cannot be identified as observations.

It is thus not that there is no need for archaeological theory, but we seem to be at loss as to what it can be in the cross-fire of natural and human scientific conceptions.

John Bintliff argued that “[a] rehearsal of the old antagonisms between New Archaeology and the postprocessual programme is unproductive, if not tiresome, and it does not seem to me to be taking us anywhere” (Bintliff, 2000: 160). We believe there is nonetheless value in trying to make the fault lines of that antagonism as clear as possible and that is our aim in this paper.

In a nutshell, according to the analysis of the field put forward in this paper, against the foregoing background, this paper argues that archaeology is essentially a discipline orientated to the extraction and analysis of data, and that being the case, theory can be meaningful in archaeology only if theory connects theoretical concepts with data. That is to say, archaeological theory has to say what patterns in our data does a given piece of archaeological theory lead us to look for. To the detriment of not an insignificant portion of 20th and 21st century archaeological theory, the relationship of theoretical concepts to data and interpretation has unfortunately often been felt to be tenuous.

At the same time, we draw attention to a longer history of conceptual problems arising from the extension of emerging natural scientific techniques and ideas being applied to understand (pre)history that were later criticized for having ignored considerable socio-cultural depth to them. Issues from Social Darwinism via the selfish gene to the modern day archaeological debates about aDN and migration are cases in point. This is a history that any third and subsequent scientific revolutions in archaeology and elsewhere ought to keep in mind. With a view on this history, our paper is a plea for reflective Middle Range theory in archaeology.

That said, our paper also makes a case for theoretical reflection as an essential scientific skill. Reflection is characterized as the process of raising (self)awareness about the (implicit) role of paradigms and theoretical concepts in archaeological interpretation including the traversing of natural scientific concepts to human scientific explanation. We propose a modest concept of Middle Range Theory at the core of reflective, interdisciplinary socio-environmental archaeology.

Middle Range Theory

Any introduction to archaeology will characterize the discipline as essentially orientated to the excavation and analysis of the archaeological record. Therefore, if an archaeological concept means anything, it does so because the concept is somehow, even if indirectly, connected with the kinds of stuff the archaeologist find in the ground. In a phrase, in archaeology, quantitative concepts should be associated with qualitative concepts.

This simple observation allows us to pinpoint the nature of Bintliff's and others' critique of archaeological theory. Basically, in their view, ever since the emergence of postprocessualism, archaeological theorists have been struggling to connect theoretical concepts with archaeological data. As a result, postprocessual archaeological theory has been charged with making grand theoretical assertions but failing to connect these with particular data in a way that is unequivocal and/or substantially new.
We would like to suggest that the concepts of Middle Range Theory (MRT) on the one hand and that of High Theory on the other, can be used to conceptualize the situation. We do not wish to go in too much detail to the long debate about MRT in archaeology (Forslund, 2004) but few words on the notion are in order to make explicit the kinds of issues MRT has historically involved.

The term goes back to the American sociologist Robert K. Merton who in his Social Theory and Social Structure (1949, 1957, 1968) criticized the tendency in the American sociology of the time to work with grand theoretical systems (Merton, 1968: 39; Geels, 2007: 628). At the same time, Merton also expressed his criticism towards the opposite approach —small-scale empirical propositions informing day-to-day research. Merton (1968: 44) suggested MRT as a middle way between minor empirical hypotheses and grand theory:

[…] middle-range theory enables us to transcend the mock problem of a theoretical conflict between the nomothetic and the idiographic, between the general and the altogether particular, between generalizing sociological theory and historicism.

One hallmark of MRT for Merton was its ability to “guide empirical inquiry” (Merton, 1968: 39). Merton (1968: 39) wrote:

Middle-range theory involves abstractions, of course, but they are close enough to observed data to be incorporated in propositions that permit empirical testing.

The concept of MRT entered archaeology with Lewis Binford (1977) who used the term to refer to the theories of the formation processes of the archaeological record. Others followed with a critique introducing concepts of MRT that were taken as more Mertonian than Binfordian in spirit and letter (Schiffer, 1988; Raab & Goodyear, 1984).

As a result of the multifaceted history of the concept, the words “Middle-Range Theory” tend to evoke many associations in archaeology and its use is rarely precise. In our modest concept of MRT, it simply denotes archaeological theory that prioritizes the relation of theoretical concepts to empirical data.

Relatedly, MRT signals a certain desire to mediate “between undesirable extremes”, as Frank Geels observed in the context of Science and Technology Studies1 —the appearance of the concept of MRT is “an indication of discontent in a discipline” (Geels, 2007: 627). This arguably is the situation in archaeology in that there exists the perception that the latest breakthroughs in archaeological theory are failing to make an unequivocal interpretative difference. Its content as a concept aside, MRT is, therefore, quite specifically situated in a particular juncture in the history of archaeology.

High Range Theory and Reflectivity

Generally, theories can so to speak “fly at different altitudes”. In biology, a theory within population ecology flies at a different altitude than the general theory of evolution. This holds true also for archaeology. As we saw above, aversion against theory in archaeology often stems from the impression that here is too large a distance between the archaeological record and some “satellite” altitude of general theories stemming from the remote stratosphere of social theories.

---

1 Science and Technology Studies are a sociological and philosophical branch of history of science studying social and cultural formation processes of scientific knowledge and theories.
Merton, and several commentators on the MRT debate thereafter, distinguish between high and middle level theory, and sometimes low level theory (Raab & Goodyear, 1984; Smith, 2011). High level theory is by definition something that sets off from fairly abstract (philosophical, if you like) debates about the fundamental nature of something—say, of agency, of materiality, to pick some recent examples (Witmore, 2014; ANONYMIZED; Dobres & Robb 2000). The worry raised by Bintliff and others about “high” theory is that it threatens not to have an obvious empirical application for archaeological purposes. In our view, Bintliff and others’ worry is essentially justified.

That said, “high” theoretical and reflectivity ought to be considered a part of any scientist's toolkit. In the most general sense, the concept of reflection can be understood as referring to the awareness of research traditions or paradigms, theory, key “high” theoretical concepts and the influence these have on archaeological interpretation. In a more particular sense, reflectivity critically looks at how concepts are (implicitly) defined and used for explanatory purposes within a given paradigm (Kuhn, 1969; Lucas, 2017). A case in point is the ongoing debate about how the aDNA techniques are being used to (implicitly) define migration.

Gavin Lucas' (2017) recent discussion of the concept of a paradigm (Kuhn, 1996 [1962]) is very interesting pointing out some important nuances in the term—as well as the loose use the concept is regularly put to in archaeology and beyond. Lucas (2017: 265) notes that one can think of paradigms and research matrices in terms of, both, what they are and what they do. One of the central ways in which a paradigm does something, namely divide the field of science in schools or camps, is by conveying their orientation through classic publications, methods, techniques, instruments, and the like as these are disseminated through research training and communication within a field that shares a paradigm. Taking one approach or another will probably cancel out others (if not by meaning, then simply by time and energy used to pursue one or the other pathway), therefore, paradigms are not a trivial matter.

Every once in a while, archaeologists implicate themselves as guilty of naïve empiricism, a certain “fetishisation of ‘data’, ‘facts’ and quantitative methods” (Sørensen, 2017: 1), the way we look at archaeological features, findings or data and the question we ask are often implicitly directed by these theoretical approaches and certain “controlling models” (Clarke, 1972; Wylie, 2002). Therefore, it is of substantial importance to be aware of how theoretical and other (e.g. political, or ethical) conceptions implicitly may shape archaeological thought—a struggle that surely is difficult and never-ending, so to speak, a hermeneutic circle or spiral. The danger of unreflective and unnoticed migration of concepts from one domain to other is perhaps particularly present in socio-environmental archaeology understood as archaeology working with, both, approaches from the cultural and social sciences as well as environmental natural sciences.

An integral issue is the implicit transmissionimmigration of concepts and thought-models from one domain to another, as recently debated in the aDNA studies regarding the concept of migration (Heyd, 2017; Ion, 2017; Furholt, 2018). For decades in archaeology, Gustav Kossinna's concept of monolithic archaeological cultures identified on the basis of shared material culture, and the associated concept of migration as geographical movement of such a monolith, has been discussed critically to the point of rejection (cp. a similar paradigm in “New World archaeology”, see Clark, 1993). However, with the rise of aDNA, this concept of migrations has seemingly returned, this time migration being equated with the movement and appearance of certain aDNA in different

2 Reflectivity is perhaps related but still distinct, in particular in its methods, from Reflexive Archaeology as introduced in archaeological field practice by Ian Hodder (1997; Berggren, 2015).
areas. The debate is on-going, but arguably — alongside technical questions about the adequacy of sample sizes and the like — the danger here is the unreflective equation of the appearance "movement" of aDNA from one geographic area to the next with the concept of migration the latter of which arguably contains social, cultural, and political dimensions not visible in bare mere transmissions of aDNA. In Ion's words, aDNA "is just data in want of a narrative" (Ion, 2017: 186; see also Gramsch, 2015: 343). Surely, there may have been periods for which demic expansion is the appropriate model, yet the complex nature of the record may not have been sufficiently considered (Gramsch, 2015: 343). In other words, Explanatory power has been sought in a reduction to the supposed essentials seemingly allowing the archaeologists not to step into the bog of interpretative socio-cultural particularities in the first place.

An underlying issue we wish to draw attention here is that we have been here before. The history of science knows of cases of how a natural scientific discoveries have first seemed to reveal the nature of things only for significant doubts and reversals to surface later. Thus, in a parallel case of a reduction to the essentials, Richard Dawkins' view on the selfish gene were once, in not too distant past (Dawkins, 1976), used to provide a one stop shop accounts of such arguably quite complex and mixed phenomenon as altruism. Later critics would indeed argue that the reduction to the essentials was mistaken as, once again, what was once thought to be the essential causal core of a phenomenon was probably better thought of as at least partly socio-culturally shaped (Gould, 2002; Sterelny, 2007).

The immigration of natural scientific concepts into human scientific interpretation, of course, has a longer history. A classic example is social evolutionism, inspired by Darwin's evolutionary theory, and leading to the imposition of the concept of stages of development upon the variety of human social and cultural life. The concept of evolution extended to human social and cultural development had the analysts project the concept of evolution upon a domain that, however, worked by rather different principles, as prominent critics such as Tylor, Spencer, and Boas would argue.

In archaeology, Shanks and Tilley's classic critique of evolutionism in archaeology is complex, but one of the key statements was that, in this tradition, "societies were viewed as being involved in an endless series of technologically governed environmental adaptations" (Shanks & Tilley, 1988: 152). In other words, in evolutionistic research, "[t]he search is for universal processes underlying different empirical sequences of societal change, and the reason for this change is environmental adaptation" (Shanks & Tilley, 1988: 140). Shanks and Tilley trace this heritage to a range of literature from Childe (1936) to then-recent work in the 1970s and 1980s by Binford (1972), Flannery (1972), Renfrew (1972), Bintliff (1984), and others.

The alternative to evolutionism Shanks and Tilley proposed was put forward with apologies for the general, outline-like quality — one might say "high" theoretical character — of their remarks (Shanks & Tilley, 1988: 185). A central aspect of it was that

Societies, unlike individual organisms, do not have any clear-cut physical parameters or boundaries, nor do societies have conscious problems of self-maintenance or a need to adapt. Individuals may have these characteristics but they cannot be validly anthropomorphized in terms of entire social totalities … Societies construct their own social reality and the reproduction of societies entails far more than physical, biological reproduction. (Shanks & Tilley, 1988: 155)

The alternative proposed there essentially says that humans are subject to different, socially constructed determinations that can and do supplant and redefine the environmental constraints. The
In any case, the present point is, it is only “high” theory that produces this sort of reflection on the fundamentals of our scientific conduct. Reasoned and reflective archaeology is better than one conducted unawares of major paradigm alternatives (ANONYMIZED).—

Furthermore, reflection may help avoid implausible equations of scientifically detected phenomena (such as demic shifts of aDNA) with immigration. Interdisciplinary work environments—one in which representatives of both “cultures” regularly meet, present, and discuss on-going work—constitute an excellent forum for facilitating awareness and debate. In the end, however, as argued above, archaeology is a fundamentally data-based science, and “high” theory really ought to make the extra effort to descend from principled discussions upon the middle-range level, that is, to questions of operationalization of theoretical insights.

**Scales of Transformation**

Building on the foregoing metaphor of altitude, we see a continuum of theory formation in archaeology. This array can be organized as different layers or levels of theory formation (cp. Schiffer 1988:3). The layer model gives a static picture of theoretical altitudes, while a dynamic perspective would explain, why and how theories can “reach” specific different altitudes and how layers can come into contact. What is needed, then, would be a fleshed-out meta-theoretical hierarchical layer model of theories within archaeology. We argue that such model would serve archaeology far more than a naïve dismissal of theory on the one hand and the continued production of “high” theory in absence of operationalization on the other. Our ultimate interest is to improve our explanations through providing an explicit epistemological model of the scales of transformation.

To begin to conceptualize these scales, an analytic distinction between climatic and environmental spheres on the one hand, and social and cultural spheres on the other, may be made (Figure 1). In such a scheme, a reductivist explanation can be understood as one that seeks to collapse the two spheres together again by proposing that some one explanatory factor or factors from one sphere would alone account for the phenomenon under investigation. An example of a reductivist account could be the naïve Marxist view that changes in economic structure in the last instance drive other changes. A second example would be the environmental determinist view that the changing biological frame ultimately drives change in (pre)history.

---

3 We adopt Schiffer's (1988) classic model of the structure of archaeological theory in the recognition that there is in fact a plurality of theories used in archaeology. In Schiffer's words, this ranges from theories of Reconstruction of the “cultural and natural past” (Schiffer 1988: 469), via Methodological theory concerning “methods and techniques (of recovery, analysis, and inference)” (Schiffer 1988: 474), to Social Theory, the last of which has been our focus in this paper.
Figure 1. Scales of Transformation and Socio-Environmental Archaeology. The sphere of cultural and social processes is analytically distinguished from the sphere of climatic and environmental processes. Sub-processes and theories of them, within these spheres, are characterized by their differing hypothetical temporal and spatial scales (lengths of the arrows).

By contrast, the non-reductivist argument can be made that even if we assume triggers from one sphere—say, changing climate and environment—the logic by which these changes develop in the other sphere would be its own. Thus, for example, differences and changes in the form of governance will crucially affect the way communities deal with environmental challenges (Oliver-Smith 2012, Keyzer 2016).

A second point we wish to argue is, there are differing spatial and temporal scales at which a given (pre)historic phenomenon could plausibly be said to be occurring which bears some significance to explanation and understanding. To give a simple example, unless we talk about sudden catastrophes (Grattan 2006, Middleton 2017), most climatic and environmental changes occur over a lot longer time spans than most socio-cultural processes do. In modern contexts, this is referred to as the shifting base line: over human generations, changes in the environmental and climate may become imperceptible due to the relative comparative shortness of human memory as well as contextualism pertaining to human perception of the environment. As a rule, phenomena in the socio-cultural sphere can perhaps be said to be occurring in temporally and spatially shorter scales, yet, arguably, there is great variation there in how some economic processes may be global while cultural processes often are more local.

The point is, given the potential differences in the scales in which different phenomena can be plausibly said to be occurring, socio-environmental archaeology probably ought to be conscious of the differences in the scales. For example, it is not plausible that, say, the temporally long and geographically large scale phenomenon of the neolithization could be reductivistically understood and explained by appeal to long term climatic-environmental nor by short term ideological changes. In so far as the neolithization was a spatially and temporally dispersed phenomenon, it's explanation would seem to require as much. The ideal type of socio-environmental archaeology would be one that is conscious of, studies, and provides insights into the different scales of transformation.

References

Barrett *ABJC* (2014) *The material constitution of humanness*. Archaeological Dialogues 21(1) 65-


Conceptual Modelling and Humanities

Yannic Ole Kropp\textsuperscript{1}, Bernhard Thalheim\textsuperscript{2}

Abstract: Humanities are becoming a hyping field of intensive research for computer researchers. It seems that conceptual models may be the basis for development of appropriate solutions of digitalisation problems in social sciences. At the same time, humanities and social sciences can fertilise conceptual modelling. The notion of conceptual model becomes enriched. The approaches to modelling in social sciences thus result in a deeper understanding of modelling. The main aim of this paper is to learn from social sciences for conceptual modelling and to fertilise the field of conceptual modelling.

1 The Value of Conceptual Modelling

1.1 Computer science is IT system-oriented

Computer system development is a complex process and needs abstraction, separation of concern, approaches for handling complexity and mature support for communication within development teams. Models are one of the main artifacts for abstraction and complexity reduction. Computer science uses more than 50 different kinds of modelling languages and modelling approaches. Models have thus been a means for system construction for a long time. Models are widely used as a universal instrument whenever humans are involved and an understanding of computer properties is essential. They are enhanced by commonly accepted concepts and thus become conceptual models. The main deployment scenario for models and conceptual models is still system construction (with description, prescription, and coding sub-scenarios) although other scenarios became popular, e.g. documentation, communication, negotiation, conceptualisation, and learning.

1.2 Learning from Digital Humanities

Digital humanities become a hyping buzzword nowadays due to digitalisation and due to over-applying computer technology. We have been engaged in a number of projects, e.g. [1, 2, 4, 6, 9]. We step back now and reconsider the challenges to conceptual modelling in

\textsuperscript{1}Department of Computer Science, Christian-Albrechts University of Kiel, 24098 Kiel, Germany, yk@is.informatik.uni-kiel.de

\textsuperscript{2}Department of Computer Science, Christian-Albrechts University of Kiel, 24098 Kiel, Germany, thalheim@is.informatik.uni-kiel.de
these projects and generalize the experience we have gained in these projects. Let us first present a number of observations:

Observation (1): The concept spaces used in social sciences underpins the conceptual model. Conceptions are systems of concepts. The concept space is typically complex structured. It is used in a multi-viewpoint manifold.

Observation (2): Conceptualisation has to be co-considered at various abstraction levels at the same time, e.g. at the micro-, meso-, and macro-level.

Observation (3): The mould\(^3\) (and methodology) determines model handling and the utilisation scenarios in which a model functions by playing roles. Models incorporate their function.

Observation (4): The model consists of a surface (or normal) sub-model and of deep (implicit, supplanted) sub-models which represent the disciplinary assumptions, the background, and the context. The deep models are the intrinsic components of the model. Conceptualisation might be four-dimensional: sign, social embedding, context, and meaning spaces.

Observation (5): Models benefit and suffer from the art of omission. Social and cultural embeddings are considered to be obvious and can thus be omitted.

Observation (6): Models may be materialised and then they have a material obstinacy due to the chosen material.

Observation (7): Conceptual models have to carry at the same time a manifold of understandings and a manifold of domain-situation models.

1.3 The storyline

These observations and lessons are useful for conceptual modelling in our area. They are often not explicitly observed in computer science. They are however implicitly used. Think, for instance, on conceptual database models. We often use a conceptual schema that describes the structure of the entire database system and use additionally a number of conceptual views that describe the viewpoints of users of the database. Therefore, we explain now how conceptual modelling can learn from successful approaches in social sciences. The learning process will enhance the added value of conceptual modelling.

2 Learning from Humanities for Conceptual Modelling

According to [5, 10, 13] we define the model notion as follows:

\(^3\) The mould is a hollow form or matrix or simply frame for giving things (such as models) a particular shape. In production, moulds are used as a shaped cavity for forming fluid or plastic things.
“A model is a well-formed, adequate, and dependable instrument that represents origins and that functions in utilisation scenarios.”

“Its criteria of well-formedness, adequacy, and dependability must be commonly accepted by its community of practice (CoP) within some context and correspond to the functions that a model fulfills in utilisation scenarios.”

Well-formedness is often considered as a specific modelling language requirement. The criteria for adequacy are analogy (as a generalisation of the mapping property that forms a tight kind of analogy), being focused (as a generalisation of truncation or abstraction), and satisfying the purpose (as a generalisation of classical pragmatics properties). The model has another constituents that are often taken for granted. The model is based on a background, represents origins, is accepted by a community of practice, and follows the accepted context. The model thus becomes dependable, i.e. it is justified or viable and has a sufficient quality. Justification includes empirical corroboration, rational coherence, falsifiability (in our area often treated as validation or verification), and relative stability. The instrument is sufficient by its quality characterisation for internal quality, external quality and quality in use. Sufficiency is typically combined with some assurance evaluation (tolerance, modality, confidence, and restrictions).

2.1 The notion of conceptual model

A notion of conceptual model might be a slim, light, or concise one depending on the level of detail we need in model utilisation. We will use in the sequel one notion, i.e. the concise notion and refer for slim and light versions to [12, 14].

Concise version:
Conceptual Model \Subseteq (Model \oplus Concept(ion)s) \Rightarrow Enabler [7]:

A conceptual model is a model that is enhanced by concept(ion)s from a concept(ion) space, is formulated in a language that allows well-structured formulations, is based on mental/perception/situation models with their embedded concept(ion)s, and is oriented on a mould and on deep models that are commonly accepted.

The mould and the deep models form the matrix of a model [11]. We notice that a conceptual model typically consists of a model suite in social sciences. Each of the models in a model suite reflects some viewpoint or aspect.
2.2 The added value of conceptual modelling

Models do not have to be conceptual models. Conceptual models do not have to be based on an ontology. The main purpose of conception as a system of concept or of a concept(ion) space is the integration of interpretation pattern that ease the communication, understanding, delivery of a model in dependence on the model functions. The concept(ion) space, the mould of model utilisation, and the explicit knowledge of the social determination provide a means for the correct and sufficiently precise interpretation of the model elements.

2.3 The four dimensions of conceptual modelling

The consideration of the strategic, tactical, and operational sides of modelling and of conceptual modelling drives us to consider the four dimensions in Figure 1. These dimensions

Fig. 1: The representation, application context, foundation, and social dimension of conceptual modelling

cover the application areas in [15] and especially those in humanities. Information systems typically consider the representation dimension and only one of the branches of the foundation dimension. Computer engineering especially considers the application context dimension.

Prescriptive conceptual models that are used as the blueprint for system realisation also consider this dimension. The social embedding is typical for social sciences. The foundation dimension has additional aspects in social sciences since corroboration, comprehension and systematisation are far more complex. Conceptualisation is based on complex concept and conception spaces.
2.4 Handling forgetful mappings to IT and DBMS technology

In is often claimed that conceptual database or data models are mainly descriptive ones. Description is, however, only one of the functions that a conceptual data model has in a system development scenario. Other typical scenarios are documentation, prescription, communication, negotiation, and explanation. These scenarios are also observed for humanities.

In system construction we transform the conceptual data model to corresponding realisation models. This transformation also changes the semantics from rich semantics of conceptual models to lexical semantics which is based on the lexical interpretation of the words used in realisation models according to the meaning in the given application area. It is thus forgetful. The reestablishment of the conceptualisation must thus be handled by a reference to the conceptual model what also means to use a tight bundling of all models in the case of system maintenance (e.g. evolution and migration) and integration. The social dimension and the foundation dimension get also lost during transformation.

2.5 Sophisticated conceptual models are model suites

Based on the observations, we should consider a conceptual model as a model suite, i.e. a coherent collection of explicitly associated models. The associations are explicitly stated, enhanced by explicit maintenance schemata, and supported by tracers for the establishment of coherence [8]. Each model in the model suite has its orientation and its functions in utilisation scenarios. The association schema among the models allows to consider the model suite as a complex but holistic model.

Model suites in most sciences and engineering incorporate some conceptual models. This situation is not different for social sciences. For instance, the CRC 1266 [1, 2] uses as a complex model of transformation a model suite consisting of models for socio-economic formation (cluster B-E), for socio-environmental components of change (cluster F), and for natural science investigation (cluster G). The interplay of these models allows to suppose hypotheses and to draw conclusions. Most models are already conceptual ones. They use, however, different conception spaces. The association among these models is handled by interlinkage groups within the CRC.

2.6 Models as mediating instruments instead of middle-range theories

Middle-range theories [2] are essentially model suites. They are used for an integrating consideration of quantitative sources and theory conceptions. Quantitative sources are used for derivation of quantitative concepts. The theory offer underpins these concepts. Qualitative theory-oriented research uses theoretical concept(ion)s. These concepts are supported by supporting sources which are often generated before and might use the current
quantitative sources. A theory integrates these concepts. We use typically several theories, e.g. for plausibility check, for investigation, explanation, knowledge experience propagation, and discovery scenarios. In a proxy-based research we start with proxy sources that might be underpinned by proxy concepts. This research results in a theory request that can be satisfied by a theory offer.

This approach often results in a gap between qualitative and quantitative research. Models can be used to render the theory offer. At the same time, models may also render a qualitative theory. The rendering procedures are typically different. A model suite can now be constructed by models for theoretical concepts from one side and by models for quantitative concepts from the other side. In this case, we use models for the quantitative theory offers and for the qualitative theories. This approach is depicted in Figure 2.

This approach has already used for the investigation in the CRC 1266 [2]. In a similar form we can consider now conceptual models for other application cases.

3 Concluding: Conceptual Modelling Inspired by Humanities

Conceptual modelling is a widely used practice in many science and engineering disciplines. The current practice used for database conceptualisation can be enhanced by a number of insights that we observed in social science research.

- The concept(ion) space is often far more complex structured than finally represented and used for a singleton conceptual model. We should consider conceptual models that orient on different aspects and different levels.
- The context dimension should not be neglected for conceptual models.

Fig. 2: Models as integrating and mediating instrument for conceptualisation, investigation, explanation, knowledge experience propagation, and discovery
• The social dimension and the foundation dimension are equally important as the representation dimension.
• Model and especially conceptual models consist of a number of models and thus form a model suite.

We got now additionally a number of special necessities for conceptual modelling without which conceptual models are of low quality, not justified, and also not adequate.

Deep models: Models consist of normal sub-models and deep sub-models. The first ones are given in an extrinsic and explicit form. The later ones are often concealed.

Model mould: The second element of the matrix of modelling is the mould. We know a number of canonic approaches that guide the modelling process, the modelling outcome, and the capacity of the finally developed model.

Concept-biased modelling: Conceptual models are typically deeply biased by the concepts in a given domain. Concepts such as “village”, “settlement” and “center” are essentially representing the same understanding but are used in very different contexts. The same applies to database models, e.g. the concepts of “Person” or “Address” depend on geographic, law etc. assumptions.

Functions of models as the guiding principle: The utilisation scenarios determine the functions that a model has in such scenarios. The model is an instrument in these scenarios. Whether it is a proper and fit-to-use instrument depends on the function the model has (and thus on the purpose and the goal).

Bibliography


 Remark

We thank the reviewers for their remarks, suggestions, and critics. The paper is based on our previous papers on models, e.g. on the generalisation of approaches used in design science research [3]. The compendium [15] presents model notions and modelling used in agriculture, archeology, arts, biology, chemistry, computer science, economics, electrotechnics, environmental sciences, farming, geosciences, historical sciences, languages, mathematics, medicine, ocean sciences, pedagogical science, philosophy, physics, political sciences, sociology, and sports at Kiel university. It is based on a decade of Tuesday-evening-open-end discussions on models and modelling in sciences. We selected only four of our collaboration projects from humanities research and discussed some of the modelling lessons.

 Acknowledgement

This research was performed in the framework of the CRC 1266* ‘Scales of Transformation - Human-Environmental Interaction in Prehistoric and Archaic Societies’ which is funded by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation - Projektnummer 2901391021 - SFB 1266)*. We thank both institutions for enabling this work.

* [http://www.sfb1266.uni-kiel.de/en/](http://www.sfb1266.uni-kiel.de/en/)