Talks 2015 Given By Bernhard Thalheim

Selected and Compiled Version of All 2015 Talks
Conferences, Tutorials, Miscellaneous

2015

Bernhard Thalheim
Dr. rer.nat.habil.
Prof. @ Christian Albrechts University at Kiel, Germany
Department of Computer Science
Information Systems Engineering Group

(*) Kolmogorov Professor h.c. @ Lomonossov University Moscow, Russia
Overview

- Big Data - Big Challenges (invited keynote). 5th Int. Conf. Advances in Software Engineering, Prince-Sultan University, Riyadh, Saudi Arabia, 12.-12.05.2015
- Culture-Aware Web Information System Development. EJC'2015, Maribor, Slovenia, 08.06.2015
- Database Constraints - A Survey. Logics for Dependence and Independence, Dagstuhl, Germany, 22.06.2015
- Revisiting the Definition of the Relational Tuple Calculus. ADBIS 2015, Futuroscope, Poitiers, France, 09.09.2015
Not In This Collection

- Privatsphäre oder Ego-Striptease im Internet - Privacy-Enhanced Information Systems: Techniken und Möglichkeiten (in various variants: juristical, technological, theoretical, advisory, system development, youngsters). Schleswig-Holsteinische Universitäts-Gesellschaft (SHUG), Plön, Germany, 05.02.2015; Friedrichstadt, Germany, 04.03.2015; Sylt, Germany, 28.09.2015; Eckernförde, Germany, 12.11.2015.

- The model of the conceptual model. Modellierungsworkshop@CAU, Kiel, Germany, 10.02.2015

- Models in Computer Science, Symposium QFAM der GI. Hamburg, Germany, 12.03.2015

- Developing a Foundation of BPMN. Logics for Dependence and Independence, Dagstuhl, Germany, 23.06.2015

- Reasoning on Models of the Heart. IIIfTC workshop, Kiel, Germany, 29.-29.08.2015


- Big, Bigger, Biggest Data - unser Geldgewinnungsbluff?. Night of the pros, CAU Kiel, Kiel, Germany, 20.11.2015

Edited Volumes List for 2015


References

and a survey on results achieved are given in the 2015 Almanach of the Technical Faculty of Christian-Albrechts University at Kiel.

Papers are provided on demand.
The English and German talks which are not in this collection will be also send on demand.

Note: In 2015 our ISE group has published 27 papers and 4 editings.

Summary

Overall: 20 talks given by B. Thalheim in 2015.
Furthermore 1 talk by Frank Kramer,
1 talk by Hans-Joachim Klein,
and 2 talks Marina Tropmann-Frick.
This list does not include internal talks of the ISE group.
ISE Publication List for 2015


ISE Publication List for 2015


ISE Publication List for 2015


Big Data - Big Challenges

$5^{th}$ Conf. Advances in Software Engineering
Prince Sultan University, Saudi Arabia

12.5.2015

Bernhard Thalheim
Technologie der Informationssysteme
Institut für Informatik, Christian-Albrechts-Universität zu Kiel, BRD
Kolmogorow-Professor e.h. der Lomonossow-Universität Moskau
Instead of a Personal Profile

HERM 'bible'

Handbook of Conceptual Modeling

Dependency: TEUBNER-TEXTE zur Mathematik

B. Thalheim

Hype
Example
Solutions
Data Science
Privacy
Challenges

Content
Information
Concept
Topic

ADCIS
ASM
CM
EJC
e-Bus.
ER

FoIKS
MFDBS
NLDB
Semantics
WIS
WISE

Practical Database Design Methodologies.
Kuwait University Press, 1989

i.e., reaching the big triple "3"s (3, 30, 300)
(3 (habilitation students, profs, books, founded confs), 30 (PhD students, editorials, projects, co-chair confs),
300 (master/diploma/bachelor students, papers, presentations, member PC confs))

Big Data,
New Challenges
12.5.2015
B. Thalheim
Instead of a Personal Profile: General
Instead of a Personal Profile: Most Cited Papers ($h \geq 25$)
Overview

(1) Big data hype
   Yet another revision of what we know already.

(2) A big data example
   Think ... before doing anything.

(3) Big data solutions
   We are nowadays able to manage such data.

(4) Data Science
   We develop a new research methodology and new research approach.

(5) Data privacy
   Solutions against big brothers data. We need new rules, new restrictions, and a new culture.

(6) Challenges
   We need novel solutions and novel management.
# Data, Data, Data … like Sand in the Desert

**From mole-hill to mountains and high mountains!**

<table>
<thead>
<tr>
<th>Year</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1950ies</td>
<td>insurance data, UNIVAC (John Hancock Mutual Life Insurance): 2 million insured</td>
</tr>
<tr>
<td>1960ies</td>
<td>flight booking system American Airlines Sabre: 807 megabytes ($10^9$)</td>
</tr>
<tr>
<td>1970ies</td>
<td>Cosmos system (FedEx): 80 gigabytes ($10^{11}$)</td>
</tr>
<tr>
<td>1980ies</td>
<td>banks, e.g. CityCorb NAIB with 450 gigabytes ($5 \cdot 10^{11}$)</td>
</tr>
<tr>
<td>1990ies</td>
<td>supermarket analysis systems, e.g. WalMart with 180 terabytes ($2 \cdot 10^{14}$)</td>
</tr>
<tr>
<td>2000ies</td>
<td>internet engines, e.g. Google with 25 petabytes ($2.5 \cdot 10^{16}$)</td>
</tr>
<tr>
<td>2010ies</td>
<td>Web 2.0 services, e.g. Facebook 100 petabytes ($10^{17}$)</td>
</tr>
<tr>
<td></td>
<td>analysis of these data generates each day 500 terabytes ($5 \cdot 10^{14}$)</td>
</tr>
<tr>
<td>2015</td>
<td>earth exploration data reach 10 exabytes ($10^{19}$)</td>
</tr>
</tbody>
</table>

## What is the real challenge?

- memory: in-memory, secondary, tertiary
- algorithms complexity: polynomial, $n \cdot \log(n)$, $n$, $\log(n)$
- throughput for parallel transactions: 80,000 in parallel
What is Big Data?

There is no commonly accepted definition!

Either no definition at all or reference to some applications or some size definition.

Each author prefers another definition!

There is no guru, thousands of authors in hundreds of journals and conferences and in blogs. Quality assurance is typical for a hype: none at all.

The main definition is phenomenal: Triple V’s.

This is however not at all a definition.

My personal preference:
Big data is data with a size beyond the ability of commonly used software tools to capture, to store, to manage, and to process the data with acceptable time limits.
My Agenda Implied by the Definition: Data Literacy!!!

size beyond the ability of software tools to

- capture
- store
- manage
- process

with acceptable time limits

Resulting questions (applicable to any research issue and to big data too):

1. What does it mean?
2. Why does this matter? (Not: Does it matter?)
3. How then we should respond?
Agenda: Data Literacy!!!

Theodore Levitt: “What is needed is **discrimination** in the supply and use of data, not their abundance, regardless of relevance. Discrimination cannot be experienced in the vacuum. Magnitudes must be limited to what is **relevant** and **comfortable usable**. The effective use of information is governed by the **principle of parsimony**: limit it to the more-or-less precise purpose at hand. A good thing is not necessarily improved by its multiplication. The governing question is: **what is the question to be answered, the problem to be illuminated, the matter to be explored, the issue to be defined?** And it is precisely because these are not self-defining concepts that is essential to think them through in advance, because not amount of data tell you what information you’ll need to get at the right questions. The faster and more acrobatically the computer can perform, the greater the necessity that is presumed beneficiaries must first think about what it’s all for.”
Big Data

The V-‘Model’ for Big Data:
Volume - Velocity - Variety (- Veracity (Reliability))

Volume, velocity, and variety cannot be supported at the same time!

Better talk about Smart Data!
Applications that Need Big Data

- Energy management: sensors record consumption and support sophisticated management and control based on machine-generated, not really interpretable data

- Telecommunication: customer profiling, forecasting exodus and accession (web, social data, TA data, ...)

- Marketing: sentiment analysis using big octopuses (Twitter, web social data, ...)

- Customer services: monitoring, log analysis (also with human recorded data), location data (location-oriented advertisement)

- (Retail) Trade: personal messages in dependence on customer behaviour, face analysis, ... (web and social data, biometrics, ...), mobile data

- Data science: mining in data for next kind of science after empiric, theory-oriented, numeric/simulation-backed science

- Services: fault detection, risk analysis (all kinds of data (human generated, TA, machine-generated))
### Classification of Big Data

#### Analysis Type
- Real Time
- Batch

#### Processing Methodology
- Predictive Analysis
- Analytical
- Query and Reporting
- Miscellaneous
  - Social Network Analysis
  - Location based Analysis
  - Features recognition
  - Text Analytics
  - Statistical Algorithms
  - Transcription
  - Speech Analytics
  - 3D Reconstruction
  - Translation

#### Data Frequency
- On demand feeds
- Continuous feeds
- Real time feeds
- Time series

#### Data Type
- Meta Data
- Master Data
- Historical
- Transactional

#### Content Format
- Structured
  - Images
  - Text
- Unstructured
  - Videos
  - Documents
  - Audio
- Semi-Structured

#### Data Sources
- Web and Social Media
- Machine generated
- Human generated
- Internal Data Sources
- Transaction Data
- Biometric Data
- Via Data Providers
- Via Data Originator

#### Data Consumers
- Human
- Business Process
- Other Enterprise Applications
- Other Data Repositories

#### Hardware
- Commodity Hardware
- State of Art Hardware

---

Volume

Facebook: more than one billion of users
monthly over 600 million of mobile devises
per minute more than 650.000 content storage and delivery
with 35.000 “Likes”

200 million emails per minute
175 million short messages of tweets of more than 465 millions

Currently: databases management systems with
petabyte computation
however enormous investment
so know the value

† we need a novel algorithmics and other algorithms

Otherwise: we arrive in the blind zone
available data versus analysable data
Velocity

- enormous rate of data generation in various application fields
- that need a current computation in order react to changes depending on the application within minutes or nay seconds and in future milliseconds (stock exchange, i.e. gambling dan)

Currently: Google more that 8 million search requests per minute
Amazon: with more than 80.000 $ per minute
YouTube: 30 hours video material per minute for 1.3 million video downloads per day

First shock in 1996: Southern Hemisphere Observatory (300 GBit/sec in each of the 100 observation points that must be harmonised in appropriate time)
Variety

Technics: systems, hardware, communication

Software: versions, legacy systems, distribution, maintenance

Syntax: variety of representation and coding, granularity, abbreviations, scaling, exchange formats, schemata, transparency

Semantics: meaning of the whole data corpus or singleton data chunks, replication, homonymy, name space

Pragmatics: context, used and derived meaning, cultures

Quality: cleansing of data, formation, distribution, maintenance, precision, identification, access
Veracity

Data originate from very different courses are badly structured, are perhaps doubtful, imprecise, corrupted, inconsistent currently used analysis algorithms require however

    high quality data
that must be corrected just-in-time
incomplete, latent, approximative data,

resulting in uncertainty and imprecision.

Big data solutions must specifically optimised to unstructured and inconsistent data;

Dependability of processes, data, and models as the three essential requirements
What is collected? All or a cutout? How it is filtered and compacted? How it ages? What is going to be deleted?
NoSQL

(object-)relational database technology hits its limits

State-of-the-“art”

- data with delta changes but collection campaign
- most stupid data storage: value-key storage
- partially structured data
- often graph-structured databases
- browsing or nay zapping through document content instead of
  - querying by queries,
  - seeking for data by browsing, understanding and refining,
  - property-based questioning,
  - ferreting out data necessary by discovering,
  - searching by associations and drilling down,
  - casting about and digging into the data;
- MongoDB, Apache CouchDB, Neo4j, ArangoDB, Apache Cassandra, BigTable as the starting point
- Apache Hadoop with Map-Reduce-techniques
  not novel at all: combine-filter or structural recursion for DBMS
Big Data Solve Big Problems of Our Life

Examples: data for quality management: management of trucks, control of plane turbines

Disaster and catastrophe management (N.I.C.T.)
Data - The Oil of the Digital Age

Waste of oil: Patriot Act
2012: 2.7 zetabyte of collected data for crime detection !!! resulting in detection of about 40 potential crime cases since 2001 the money spent would be sufficient for complete health care and nourishment for everybody in the entire African continent

The new oil: current and target situation

<table>
<thead>
<tr>
<th>Big data sets</th>
<th>Classical data sets</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open world community (evolving and shrinking)</td>
<td>Closed individual world assumption</td>
</tr>
<tr>
<td>Partially inconsistent data set</td>
<td>Consistent and coherent data set</td>
</tr>
<tr>
<td>Individual access mechanisms</td>
<td>Global management</td>
</tr>
<tr>
<td>Schema-less work and thus chaotic access</td>
<td>Schemata for the data dictionary with well defined access</td>
</tr>
<tr>
<td>Redundant, often copied and partially corrected data</td>
<td>Controlled redundancy of data</td>
</tr>
</tbody>
</table>

Trading the oil as the future
Oil cartels as the nightmare
Big Data

Big: TA data, processing data, interaction data

Re-occurring problems: fragmentation; aging, duplicates, incomplete; infrastructure; reuse; provenance;

Problems with higher priority: master data management; integration, evolution; systematics for data exploration; query result presentation (KartOO!!);

Challenging old problems: increase business value of data (timely for streams, actionable by monitors, accessible by parsing, relevant in the cloud, holistic with virtualisation, secure with privacy, trustworthy by governance, authorized through timeline, ...); lower the cost of data (); realise the promise.

3-4 v’s: volume, variety, velocity, veracity
Is Big Data and Data Science: Hype or Reality?

Tushar Nawale (Saama Technologies): one of those heavily abused technology.

William Logsdon (Occidental Petroleum): Gartner Hype Cycle describes what is going on perfectly.

Maria Guinsburg (Eze Software Group): it is reality for some and hype for others. The problem existed long before and even had some proto-names, they not always has been solved or cost of solution was very high. The concept existed for quite a while. The technology started to catch up.
What are the Challenges?

Big data is data with a size beyond the ability of commonly used software tools to capture, to store, to manage, and to process the data with acceptable time limits.

**Volume:** development of intelligent stores, new devices, new management, new engines

such problems were on the research agenda all the time; and solved!!!! step by step; hardware research; transactional systems

vendors supersize me and hope that I become more stupid

**Variety:** intelligent integration, evolution, and migration of data massives

sit down and solve first all the old hometasks; think before developing; develop a culture

**Velocity:** speed combined with intelligence

do you need to travel with a plane to the nearest supermarket? develop appropriate computation means

**Veracity:** data with explicit description, with quality statements, trustworthy, information logistics

metadata; quality management; frankness for quality issues; downsize whenever lower quality is sufficient
12 Things You Must Never Do With Big Data

(1) Jump into Big Data without a corresponding business strategy.
(2) Start the project without understanding Value.
(3) Call any large-scale data project Big Data project.
(4) Ignore the Big Data cross-functional impact.
(5) Force fit traditional technologies with Big Data.
(6) Select a product suite without understanding the details.
(7) Ignore the importance of the support structure, e.g., data quality, governance, metadata, ....
(8) Approach it as a technology (only) project.
(9) Deploy a traditional “Tech plus Biz” team.
(10) Have a function focused rather than enterprise focused approach.
(11) Let your consulting firms do all the thinking for you.
(12) Limit your thinking.
(13)

Big Data is here. . . . Think big and plan clear.

© Al Naqvi, PlanClear.
And What is Thalheim’s Research Contribution to Big Data

- Current result: Revision and thoughtful application of existing technology to V-V challenges (not all four or three of them).

- Next results: Maturation of existing technology and adaptation to the new challenges, i.e. development of new theories and techniques for handling larger data, for real-time data processing, and for integration of data massives with continuous quality management.

- Later: data science.

- Why there is no paper by Thalheim about big data yet?
  I do not like to brand myself under hypes, sorry!

- Why you don’t follow the hype? And don’t publish and will thus perish?
  *My current statement:* Well, I am 63 years old ... and saw already five hypes in computer technology. None of the hypes have had the added value it promised. Some of them are now considered to be bad areas.
  *Super size me!* Big data, big gulp, Big Mac. Big data is a technological expression of gluttony. It is a recipe for indigestion and a path to obesity.
  *But I worked my way up.*
  The situation is different for my students: They have to join.

Think ... before doing anything!
A Big Data Application: Traffic Analysis
A Big Data Application

[Diagram of a railway space visualization system with modules such as Passenger Data Analyzer, Trip Scheduling Analyzer, and a multi-database system for logistic analysis.]
The Essence of the Solution

Microdata: origin-destination-(arrival)time data (GB)
Mesodata: purged and extended by potential train runs, corrections by train weight (hitting TB)
Analysis data: dynamic railway space analyser data (ranging to many TB per 10 minute intervals)
Presentation data: visual inspection data

So: infeasible, unmanageable, data cemetery, impossible.
## The Main Data Source

```
((check_in_) o(origin_station),

(check_out_) d(estination_station),

(check_out_) t(ime_at_destination))
```

<table>
<thead>
<tr>
<th>Passenger</th>
<th>Date</th>
<th>Origin</th>
<th>Destination</th>
<th>Ticket</th>
<th>-6</th>
<th>6-7</th>
<th>7-8</th>
<th>...</th>
<th>23-24</th>
<th>Sum per day</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Routing</th>
<th>Date</th>
<th>Origin</th>
<th>Destination</th>
<th>Set time</th>
<th>RouteID</th>
<th>Order Train</th>
<th>List Of Lines</th>
<th>Train Name</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>...</th>
<th>Direction</th>
<th>List Of Transfer Stations</th>
<th>Departure From Transfer Station</th>
<th>Station To</th>
<th>Arrival At Destination Station</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Weight data</th>
<th>Date</th>
<th>Station 1</th>
<th>Station 2</th>
<th>Line Name</th>
<th>Train Name</th>
<th>Estimated number of passengers</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Charts for Visual Inspection

The inspection chart for outgoing traffic from Shinjuku station
Passenger-flow (arrival-stations) departed from Shinjyuku Station (6-9am)
The inspection chart for incoming traffic to Shinjuku station
Passenger-flow (departure-stations) departed to Shinjyuku station (6-9am)
A Big Data Application: Problems we have had to Tackle for Sensor Data

Various levels of granularity at the same time

Complex filtering at real-time; high database performance

Complex windowed aggregation nested aggregates incrementally updated

Multiplexing and demultiplexing for streams of micro data

Data mining within the stream sophisticated data mining techniques

Extraordinary sizes of the meso and analysis databases

Uncomplete micro data input by data collection facilities

Meta-data injection difficult and requires high performance
Analysis Profiles, Demands and Portfolio

Profile support for the analysis
- Analysis profile
- Data profile
- Group profile

Information demand support depending on specifics
- Demand, need, resulting quality
- Focus and scope
- Context framework
- Formulation, language

Portfolio support for the analysis
- Tasks: characteristics, execution, result, control
- Data sets to be considered
- Background information and metadata
- Restrictions: subtasks, environment
Be however Careful with Aggregation of Data

Aims and assumptions

**Grouping** of data, records, objects based on selections

**Summarisability** depending on completeness and disjointness assumptions

Detection of misleading aggregations and prevention from misapplication

**Invariance properties** of aggregation operators

**Harmonisation of aggregation and abstraction** without forgetful handling

**Preservation or transformation of semantics** or explicit semantics transformation

**Derivation of repair functions** in the case of damages
The Data Exploration Cycle as Starting Point for the Data Analysis Advisor

Problem and goal formulation

Real-life system

Structure and function analysis
- System definition
  - (isolating abstraction)

Empirical data

Conceptional model

Data space
- formalisation / formalisation

Formal model

Formalisation/
- Algorithmisation
  - (generalising abstraction)

Programming

Computer model

Model checking

Applying results

Results

Model verification

Model validation
A Solution with High Potential: The Event-Oriented Observer System

Analysis space model with explicit specification of data need for analysis

Analysis-driven filtering and summarisation

Data collectors based on analysis portfolio

Inspection charts

Network data
Train run data
DAT data

Micro-data streamer and publisher
Collector management system
Event-Oriented Observer

Inspection chart management system

Station charts
... Track charts
Analysis Space

Analysis interest based on goals, intentions, targets

Analysis scenario as a small ‘workflow’ with candidate facts, measures, dimensions, and hierarchies; list of possible queries

Analysis data demand necessary for completing analysis tasks with the right granularity, level of detail

Analysis escort data from other resources for improvement

Analysis correction and calibration data for quality analysis

Analysis side data for explanation of the analysis results

Analysis context restricting or simplifying the analysis
Analysis Demand

Analysis situation through comparing initial situation with potential outcome

Analysis suppositions and necessary data for support or falsification

Analysis concept space supporting ‘semantification’ of the task and of results

Analysis phenomena

Analysis data models as model suite

Analysis agents for executing the analysis

Analysis termination and goal tests whether a given state or state set satisfies the goals
Analysis Portfolio

Analysis plan with steps/tasks, tools/instruments, analysis space exploration tools (e.g., informed search), adaptation/transformation, and quality control

Analysis task with subtasks (initial state, potential goal states, applicable state transformations, constraints), objectives, executables, and restrictions

Analysis hypotheses space

Analysis algorithmics applicable for analysis

Analysis data with input views $\mathcal{V}_C$ and functions for potential inspection charts $\mathcal{C}$ based on the meso databases (shuffling direct source, escort, side, calibration data)

Analysis potential outcome and potential failures
Data Collectors

Collector contract based on the policy, e.g., subscribe what is published

Collector definition as active media type $\mathcal{C} = (S_C, \Sigma_C)$

$S_C = \text{collector} \ (Structure, CollectorQueryForm(ViewImportingExportingFeatures))$

input views $(\mathcal{V}_{C,1}^I, q_{ExportDB_1}), \ldots, (\mathcal{V}_{C,n}^I, q_{ExportDB_n})$

queries mapping data from $\mathcal{V}_{C,i}^I$ to $S_C$

output views $(\mathcal{V}_{C,1}^O, q_1), \ldots, (\mathcal{V}_{C,m}^O, q_m)$

defined on $S_C$

functionality $F_C$

Collector activity based on activation time, redirection to other collectors after time points have been reached
Collectors in the Application

CREATE TRIGGER /* FOR INSERT EVENT INTO MICRO-DATABASE
/* POPULATE CORRESPONDING INBOUND IN COLLECTORS
/* POPULATE CORRESPONDING OUTBOUND IN COLLECTORS
ON INSERT (d,a,t) INTO MICRO-DB
INSERT INTO a-INBOUND (d,a,t)
INSERT INTO d-OUTBOUND (d,a,t) END;

Hype
Example
Solutions
Privacy
Challenges

Concept
Topic
Input data, output data and meta-data

Analysis programs for deriving data,

Inspection charts for presentation of (computed) data

Recharge functions for injection of modified data

Analysis quality agreement for coordination of the data between the collector management system and the inspection management system.

Realisation of inspection spaces through MVC

<table>
<thead>
<tr>
<th>MVC</th>
<th>model</th>
<th>view</th>
<th>controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>inspection MS</td>
<td>dataset supporter</td>
<td>inspection chart</td>
<td>transformation engine</td>
</tr>
</tbody>
</table>
A General Model of an Inspection Chart

Inspired by the Molnar/Lenz/β data cube model.

Data cell model with related data collections similar to the data cube model (cells with attributes that represent derived values) as a visualisation model.

Cell visualisations for visual representations of complex structured data (classical object-relational, hierarchies, geometric shapes, texts, links, ...)

Cell topology for interrelationship among the cells, redundancy and shared data sets

Cell traversal functions generalising classical cube operations

Constraints and restrictions for interrelated data among cells

Conceptualiser adds themes to the visualisation pipe (from micro, meso and analytical data to visualisation data)
Data Science - The Future

Data-intensive science became nowadays feasible although many research lacunas are waiting algorithmics currently with $n^3$ or $n^2$ but not yet $\log(n)$.

IO bandwidth increased by factor 10 over 10 years storage by factor $>100$ ☺

complex problems $\Rightarrow$ complex computation

The fourth generation of scientific research

<table>
<thead>
<tr>
<th>Data science: unifying theory, experiment, and simulation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Computational science: simulating complex phenomena</td>
</tr>
<tr>
<td>Theory &amp; science: models, generalisations</td>
</tr>
<tr>
<td>Empirical science: describing natural phenomena, experimenting</td>
</tr>
</tbody>
</table>

Essentials

 sede data captured by instruments or generated by simulator
 sede processed by software
 sede information/knowledge stored in computer
 sede scientist analyses database / files using data management and statistics
Digital Science (e-Science or Science 2.0)

(1) collaboration (or crowd) stories for development of science within a community (how),

(2) profiles, level of engagement and interest of collaborators (who),

(3) content that is either private or shared based a number of sharing pattern (what),

(4) coherent presentation of content depending on the profile of the user and on progress of work within a community,

(5) rights, roles and plays of contributors within the story assigned (which rights portfolio), and

(6) constraints for the participation and contribution (which conditions).
Services for Digital Science

- *services for collaboration* within a group or community of scientists,

- *data, information or knowledge services* for collection, creating, delivering, maintenance and cleansing of content within a scientific community,

- *computational, exchange and control services* for the portfolio of tasks a scientist has been assigned,

- *network services* for hooking into the network at any time independently of the current location and environment of the scientist, and

- *protection services* for privacy and security of scientific communities.
Specification of Digital Science Services

*End* (wherefore) of the services;

*Sources* (whereof) used for the service;

*Supporting means* (wherewith) the service is relying on;

*Surplus value* (worthiness) the service might give to the users;

*Purpose* (why, whereto, when, for-which-reason) of the service;

*Activities* (how) supported by the service for collaboration stories based on data consumption (what-in) and resulting in data production (what-out);

*Parties* such as suppliers (by-whom), consumers (to-whom), and producers (whichever);

*Application domain* describing the application area (wherein), application cases (wherefrom), the problems (for-what), the organizational unit (where), triggering events (whence), and IT data, control, computation (what, how)

*Context* for the service such as the system context (whereat), the story context (where-about), the coexistence context (whither), and the time context (when).
Issues Alarming People on Privacy

Orwell 2006

- Unnoticed readout due to ‘invisibility’ of escorting technology and permanent observation
  (“we are currently not completely storing car highway data”)
  calm computing of profiles, portfolio, targeted marketing

- Traceability, composition of mobility profiles, RFID in travelling tickets

- Responsibility for history of objects if those are tracked

- Paternalism based on technology, systematic observation and automatic compilation of all minor lapses

- Personalisation and categorization with all negative consequences

- Criminal abuse

Many countries with protection of proprietorship of data but without data protection
“[P]rivacy will be to the information economy of the next century what consumer protection and environmental concerns have been to the industrial society of the 20th century.”

**Media privacy:** supported by laws, constitutional rights and other legal frameworks

**Territorial privacy:** supported by laws, constitutional rights and other legal frameworks

**Communication privacy:** supported by laws, constitutional rights and other legal frameworks

**Bodily privacy:** supported by laws, constitutional rights and other legal frameworks

**Information privacy:** not well supported, tools for the “glass box customer”
Information Privacy

**Information privacy:** not well supported, tools for the “glass box customer”

**Openness and transparency:** no secret record keeping

**Individual participation:** ability of change by the subject of the record

**Data quality:** relevant to the purposes and up-to-date

**Collection limitation:** collection proportional to its purpose

**Use limitation:** used for their specific purpose by authorized personnel

**Reasonable security:** adequate security safeguard

**Accountability:** accountable for the compliance with the other principles
Privacy Principles

Ethics: respecting rights and legitimate interests of others

Proportionality: controls and costs should be commensurate with value and criticality of information

Timeliness: actions should be timely and coordinated to prevent and respond to privacy breaches

Reassessment: privacy should periodically be reassessed and upgraded accordingly

Accountability: responsibility must be explicit

Awareness: principles, standards, conventions, mechanisms should be known

Democracy: weighted against relevant rights of users and other affected individuals
Trends Against Privacy Enhancing Technologies

Ego striptease - I will show you who you are
Data crakes flickr, del.icio.us, upcoming.org and blog: social Web 2.0 applications

Powerpoint Karaoke
The linguistic fingerprint: Silver bullet or mere myth?
In the absence of trust
Transparency and privacy: The 7 laws of identity and the identity metasystem
Know your citizens: State authorities’ access to sensitive information

Lingering incapacitation by profile-driven data delivery
www.dictionary.com uses 223 tracking cookies
reorganisation of our life
The 7 laws of identity

(1) User Control and Consent - Technical identity systems must only reveal information identifying a user with the user’s consent.

(2) Minimal Disclosure for a Constrained Use - The solution that discloses the least amount of identifying information and best limits its use is the most stable long-term solution.

(3) Justifiable Parties - Digital identity systems must be designed so the disclosure of identifying information is limited to parties having a necessary and justifiable place in a given identity relationship.

(4) Directed Identity - A universal identity system must support both “omni-directional” identifiers for use by public entities and “unidirectional” identifiers for use by private entities, thus facilitating discovery while preventing unnecessary release of correlation handles.

(5) Pluralism of Operators and Technologies - A universal identity metasystem system must channel and enable the inter-working of multiple identity technologies run by multiple identity providers.

(6) Human Integration - The universal identity metasystem must define the human user to be a component of the distributed system integrated through unambiguous human-machine communication mechanisms offering protection against identity attacks.

(7) Consistent Experience Across Contexts - The unifying identity metasystem must guarantee its users a simple, consistent experience while enabling separation of contexts through multiple operators and technologies.

Microsoft has proposed architectural principles (“7 Laws of Identity”) to support convergence towards an inter-operable, secure, and privacy-enhancing plurality of identity systems - an “Identity Metasystem”.
Research Issues on Privacy Enhancing Technologies

Decentralized architectures and query methodologies for weakly structured and heavily distributed data

Automatic acquisition and integration of context for information supply on real demand (context-sensitive information logistics)

Dynamic orchestration of services, conditioning, optimal service granularity, information asymmetry, payment

Data protection and security preferences of users and automatic alignment with characteristics of services

Adjustment between inspection of technology and minimization of transaction costs

Ubiquitous and calm availability of all relevant data with redesign of business processes, including logistics

Novel cooperation and coordination models based on policies, contracts, arbiters

Support for economy of attention of human users with limited time
Our Contribution

Explicit association of pieces of information to proprietors, possessors, ..., enterprises

Systematic methodology: “private information”, privacy-based decomposition and constraints

Framework: informational privacy assertions
- definition of private information and categories
- classification of relationships of “private information pieces” to its possessors and proprietors

Model: restrictions and conditions of possession
- possession is according to consent/no-consent of the proprietor, legal/illegal possession of private information, and awareness/no-awareness

Proof of concept: Framework for closed systems
The Information Privacy Model (IPM)

Towards sound theory and feasibility of management

Infons as the basic unit of chunks of data to be considered

Information: true infon(s)

Possession of infons by agents

Proprietorship of infons by individuals

Logical and procedural treatment of possession, proprietorship and their relations

Constraints limiting the usage of infons

Architectures for information privacy enhanced information systems

Management of possession and proprietorship
Infon: The Concept

Generalizing Kauppi, Devlin, Seligman, Wille

Infon definition:

- discrete item of information
- parametric: objects, anchors

Action results in changing many infons

Basic infon set \( I \): (temporary, epistemic) subset of the set of all infons

- all dual infons of the basic infons
- expansion of all anchored infons into the basic infons

all others: secondary infon

e.g., predicates \( Patient(Bernhard, Thalheim,\ldots,100352422728) \), \( \sim Village(Dresden,Saxony) \)
Possession and Proprietorship of Infons

**Information:** infon $A$ that is true

- zero information, atomic information, compound information
- private information: atomic (compound) information: that refers to identifiable individual(s).

**Possession of single piece of atomic private information**

by many possessors and one proprietor

![Diagram](image)
State of Possession of Infons

Consent of possession

State of possession

Legality of possession

Awareness of possession

Constraints to the state of possession

Awareness of the individual

Consent of the individual

Greedy deletion of information

Restrictions on visibility of information impact
Data like Sand in the Desert - Novel Approaches are Needed

New system intelligence alike life in difficult environments

Novel data capture techniques, novel data storage, novel data management, novel data processing.
Summarising and Concluding

- Applications might require their specific architecture that is more effective than the classical architecture.

- Data analysis is seldom digging or browsing or uninformed search; it is more often hypothesis/suggestion-based. Inspection charts can thus be specified in advance.

- Event-based data collection is an alternative to data hill building open field surface mining (with data hills generated) versus underground mining.

- DBMS support this approach as well.

- Analysis portfolio as a systematic way of exploration.

- Collectors for eager/lazy data collection within the profile of interest.

- Inspection charts as a generalisation of spreadsheets.
Thank you!

thalheim@is.informatik.uni-kiel.de
Culture-Aware Web Information Systems Development

25th International Conference on Information Modelling and Knowledge Bases, EJC'2015, Maribor, Slovenia

June 10, 2015

Hannu Jaakkola(1), Bernhard Thalheim (2),(*)

(1) Tampere University of Technology, Pori, Finland
(2) Kiel University, Germany
(* Kolmogorov Professor h.c. of Lomonossow University Moscov
Outline of the Talk

Information systems development considers

- Applications: become global, multi-cultural, multi-user
- Dimensions of WIS: separation of concern
- Background: Hofstede ... Lewis
- Problems: WIS adaptation to culture
- Solutions: parametric stories and content

Our concerns:

1. Web information systems must support users within their culture
2. Culture analysis and support for various communication styles and pattern
3. User must be characterised and appropriately supported
4. Towards “industry 4.0” at the global level
Sophisticated Web Information System Support in Dependence on the Real Demand of a Given User

Users with their life cases
within their community/society
with their knowledge
within the context, e.g. culture
with their tasks and portfolio
The Six Concerns for Web Information Systems and the Conceptual Facilities

- User and intention
- Context
- Content
- Usage and story space
- Functionality
- Presentation

Overview

Dimensions

Background

Challenges

Observations

Solutions

Finally

Culture, infrastructure, organisation, system context

Goal, application area, user profile and portfolio, personal culture, information demand

Stories, tasks, choices of task flow

Navigation, search, guidance and help, work

Interfaces depending on the environment

Data, objects, knowledge

Web information system

Concept Topic

Content

Information
From Web 1.0 to Web 2.0

Web 1.0: author driven, publish/provide_story/support or advertise/wait/attrac/react/retain for users: inform/subscribe/obtain/answer/come_back

Web 2.0: user driven, content centered, GoogleAdSense, Flickr, Wikipedia, blogs, optimised search engines, pay per click, web services, participate instead be attracted, tagging, syndication common usage of bookmarks, clicks; communities; tracking goals of usage; data ownership, portability, economics, transparency; architectures of participation Aal principle: andere arbeiten lassen
Next Generation Web

Knowledge dimension of Web Work

Overview
Dimensions
Background
Challenges
Observations
Solutions
Finally

Web 2.8 collaboration variety multiple identity

Next Generation Web

micro-, meso-, macro-knowledge
contextual knowledge
behavioural knowledge
content knowledge
factual knowledge
visualisation

Content
Information
Concept
Topic
The Three-Model Perspective for Culture

User model in a social sphere depending on
- typical human culture, limitations of users,
- life cases, situations, demands, needs,
- general behavioural pattern, habits learned;

Collaboration model: collaboration within
- companies, culture in a corporation, organisations,
- project, tackle a job,
- team work, groups;

Technology model depending on expectations on
- data and content,
- functionality, typical features,
- presentation, typical way of interface.
Our EJC 2015 Research

Preliminary research based on Hofstede/Reinecke/Trompenaars and Lewis and EJC 2014: development of adaptation features (views, view towers) in dependence on user profiles and user portfolio

Cultures of user and intention EJC 2014

Culture-aware usage and story space

Web information system

Culture-aware content

Culture-aware presentation EJC 2014

Culture-aware functionality (Forthcoming paper)

Cultural context

Culture-aware stories

Culture-aware content

Next paper: Culture-aware functionality
Conceptual Modelling of Collaboration: Separation of Concern Into 3C-C

Collaboration Triangle Relating Communication, Coordination, and Cooperation

**Communication act view** based on sending and receiving

**Concurrency view** based on commonly used data, functions, and tools

**Cooperation context view** combines the context of cooperation, i.e. portfolio to be fulfilled, the cooperation story and the resources that are used
Diversity and Systems

different story lines with communication culture mismatches

communication content

communication logistics (just in time, just the simplest, on demand, in the right format, in the right amount, at the right moment, supported by the right content and the right kind of data to users delivery in the right profile, delivery in the right granularity and precision as well as in the right size, unabridged, and within the frame agreed upon in advance)

services must be adaptable to

- user cultures
- local culture
- organisation culture
National Cultures in Lewis’ Model

Overview
Dimensions
Background
Challenges
Observations
Solutions
Finally

Content
Information
Concept
Topic
5.6 Hofstede’s Dimensions

- **Power Distance** (PDI) describes the extent to which a society accepts that power is distributed unequally.
- **Individualism / Collectivism** (IND) describes the extent to which a society emphasizes the individual or the group.
- **Masculinity / Femininity** (MAS) refers to the values to be held in a society.
- **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 a culture programs its members to accept delayed gratification of their needs.
- **Well-being versus Survival** (WVS) indicates the acceptance of indulgence connected to enjoyable life and happiness. The measures of this dimension do not yet have wide coverage.
### Characteristics of Cultural Stereotypes

<table>
<thead>
<tr>
<th>Linear-active</th>
<th>Multi-Active</th>
<th>Reactive</th>
</tr>
</thead>
<tbody>
<tr>
<td>introvert</td>
<td>extrovert</td>
<td>introvert</td>
</tr>
<tr>
<td>patient</td>
<td>impatient</td>
<td>patient</td>
</tr>
<tr>
<td>plans ahead</td>
<td>plans grand outline</td>
<td>looks at general principles</td>
</tr>
<tr>
<td>does one thing at a time</td>
<td>does several things at once</td>
<td>reacts</td>
</tr>
<tr>
<td>punctual</td>
<td>not punctual</td>
<td>punctual</td>
</tr>
<tr>
<td>compartmentalises activities</td>
<td>one activity influences another</td>
<td>sees whole picture</td>
</tr>
<tr>
<td>sticks to plans</td>
<td>changes plans</td>
<td>makes slight changes</td>
</tr>
<tr>
<td>sticks to facts</td>
<td>juggles facts</td>
<td>statements are promises</td>
</tr>
<tr>
<td>gets information from official sources</td>
<td>prefers oral information</td>
<td>information from official and oral sources</td>
</tr>
<tr>
<td>follows correct procedures</td>
<td>pulls strings</td>
<td>networks</td>
</tr>
<tr>
<td>completes action changes</td>
<td>completes human transactions</td>
<td>react to partners</td>
</tr>
<tr>
<td>likes fixed agendas</td>
<td>interrelates everything</td>
<td>thoughtful</td>
</tr>
<tr>
<td>uses memoranda</td>
<td>rarely writes memos</td>
<td>plans slowly</td>
</tr>
<tr>
<td>dislikes loosing face</td>
<td>has ready excuses</td>
<td>must not lose face</td>
</tr>
</tbody>
</table>
## Relationships between Hofstede Dimensions and UI Design

<table>
<thead>
<tr>
<th>Index</th>
<th>Low</th>
<th>High</th>
</tr>
</thead>
</table>
| **PDI** | access variations, non-linear navigation  
            · non-structured data allowed  
            · most information at interface level, deep hierarchy not accepted  
            · friendly messages  
            · need for support is low | linear navigation, few links, minimal amount of alternatives  
            · structured data expected  
            · data in hierarchy, little data on top level |}
| **IDV** | traditional colors and images  
            · image to text direction  
            · high multimodality  
            · colorful expression | colors used to encode information  
            · text to image direction  
            · low multimodality  
            · monotonous color map |
## Relationships between Hofstede Dimensions and UI Design

<table>
<thead>
<tr>
<th>Index</th>
<th>Low</th>
<th>High</th>
</tr>
</thead>
</table>
| **MAS** | · little saturation, pastel colors  
· exploration and different navigation paths allowed  
· personal content presentation and friendly communication | · high contrast, bright colors  
· restricted navigation possibilities  
· encouraging words in communication |
| **UAI** | · most information at interface level, complex interfaces  
· nonlinear navigation accepted  
· colors, typography and sound to maximize information | · organize information hierarchically  
· linear navigation paths, guided  
· redundancy and cues to avoid ambiguity |
| **LTO** | · reduced information density  
· content structured into small units | · most information at interface level  
· content can be arranged around a focal area |
# Cultural Factors in WIS

<table>
<thead>
<tr>
<th>IS Property</th>
<th>Culture analysis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complexity of UI</td>
<td>More accepted in high PDI than low PDI cultures. Multi-active cultures are used in multi-tasking and are more familiar in handling tasks in non-linear manner.</td>
</tr>
<tr>
<td>Long response times</td>
<td>More accepted in high PDI than in low PDI cultures.</td>
</tr>
<tr>
<td>UI colours</td>
<td>Accepted and expected in multi-active cultures. Color map and meaning of colours differs between cultures. In some cultures colors have also emotional connections.</td>
</tr>
<tr>
<td>Symbols and logos</td>
<td>In principle symbols provide a common language. Symbols in different cultures may have different meaning. There are also symbols that are not proper to be used in some cultures.</td>
</tr>
<tr>
<td>Support for uncertain situations</td>
<td>High UAI cultures respect features that guarantee the correctness of operations - e.g. repeating questions and extra confirmation operations. In linear-active those may be disturbing.</td>
</tr>
<tr>
<td>Decision making</td>
<td>In low PDI cultures the users are more ready to make fast decisions that have wide influence. In high PDI cultures the users would need confirmation from colleagues in higher positions, which leads to circulation of the activity step by step. This kind of circulation is felt inconvenient in low PDI cultures.</td>
</tr>
<tr>
<td>Privacy issues</td>
<td>Individualistic cultures are more aware of privacy issues than collective cultures.</td>
</tr>
<tr>
<td>Feedback</td>
<td>Fast feedback is appreciated in cultures that have low LTO level; high LTO index indicates readiness to wait for the feedback.</td>
</tr>
<tr>
<td>Clarity</td>
<td>Low UAI cultures are tended to accept confused situations in IS usage; high UAI cultures appreciate clarity.</td>
</tr>
<tr>
<td>Predictability</td>
<td>Predictability of information system’s behaviour is expected in high UAI cultures.</td>
</tr>
</tbody>
</table>
Typical Mini-Stories in e-Business

- advertise and quote; represent, variations, collections
- request and response; communicate, procurement
- select and collect; selection with versions
- bargain and contract; versioning, conditioning
- requisition and order; process, assemble
  - ...
- checkout; complete, pay
- deliver and invoice; logistics
- pay or return; process, inventory, use
Checkout in Germany

Promp start
review past history
statement of context
examination of facts
frank proposal
absorb counter-argument
offer new counter-proposal
cautious but firm agreement

"Follow the agenda"

The Truth is the Truth

Registrierung
Angaben zur Lieferung
Angaben zur Bezahlung
Kundenkonto
Übersicht
Bestell-bestätigung

Registrieren
Lieferadresse eingeben
Rechnungsadresse angeben
Wahl der Zahlungsweise
Kundenpasswort wählen
Bestellung prüfen
Bestellung abschicken

Gast-Checkout
Kunden-Login

Content
Information
Concept
Topic
Checkout in Germany

Cultural-Aware WIS Development
June 10, 2015
Jaakkola/Thalheim

Overview
Dimensions
Background
Challenges
Observations
Solutions
Finally
Checkout in Spain
Checkout in Spain
Checkout in China
Checkout in US
Checkout in US

Apple Store

Overview
Dimensions
Background
Challenges
Observations
Solutions
Finally

Content
Information
Concept
Topic
Checkout in Japan
Checkout in Japan
Checkout in Finland
Checkout in Finland

Overview
Dimensions
Background
Challenges
Observations
Solutions
Finally

Content
Information
Concept
Topic
Generation of the Cultural View

Culture integration e.g. regional and local in dependency on the bundle of cultures for a (migrating) user

Policy culture according to the rules of business

Content logistics supporting the communication logistics

Additional material provided at the right moment in the best-fitted granularity

Workflow recording against the ‘lost-in-the-hyperspace’ problem

Documentation for later recovery and proof of activity

Company culture ...
Mini-Story for Checkout in Finland

Choose checkout options → Normal registration

Register as known customer → Register as guest → Provide personal details

Selection of delivery → Register as user (password, ...)

Survey of orders → Payment in variants → Outside payment

Provide address details → Confirm in variants

Register as known customer → Provide address details

Register as guest → Provide address details

Survey of orders → Payment in variants

Confirm in variants → Provide address details
Mini-Story for Checkout in Japan

Content
Information
Concept Topic

Overview
Dimensions
Background
Challenges
Observations
Solutions
Finally


**Stereotypes for Database Schemata**

(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.

Parametric database structure pattern

name of the pattern $N$

schema of the pattern $S$

deployment conditions $\Psi$,  
*parametric* integrity constraints $\Sigma$, 
parameters $p_i$ of the pattern with their pre- and post-conditions $\gamma_i, \delta_i$, 
controls $C$ for all functions that can be applied to the pattern, and 
supports $U$ for all functions that can be applied to the pattern

$$SP = (N, S, \Psi, \Sigma, \Psi, C, U)$$
## Stereotypes for Database Schemata

<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 schemata 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) (e)</td>
</tr>
<tr>
<td>Femininity</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
## Stereotypes for Database Schemata

<table>
<thead>
<tr>
<th>Cultural stereotype</th>
<th>Preferences</th>
<th>Schema</th>
</tr>
</thead>
<tbody>
<tr>
<td>Uncertainty avoidance</td>
<td>complete schema with all elements, hierarchical structuring, more linear, well-scoped sub-schemata with simple reference to main schema</td>
<td>(a),(d)</td>
</tr>
<tr>
<td>Uncertainty tolerance</td>
<td>extensible schema, flexible schema style, web-like schemata</td>
<td>(c),(e)</td>
</tr>
<tr>
<td>Long-term culture</td>
<td>all potential elements are reflected as well as all viewpoints, focused (oil stain) schemata</td>
<td>(a), (b)</td>
</tr>
<tr>
<td>Short term culture</td>
<td>handy schemata depending on current use and smooth integration of them, decomposable schemata</td>
<td>(e)</td>
</tr>
<tr>
<td>Indulgence</td>
<td>schema with a central part containing all necessary elements and further elements that might of use in future</td>
<td>(e),(c)</td>
</tr>
<tr>
<td>Restraint</td>
<td>puritanical schemata without any non-essential elements</td>
<td>(a)</td>
</tr>
<tr>
<td>Linear-active culture</td>
<td>schemata with step-wise exploration of all its aspects</td>
<td>(b)</td>
</tr>
<tr>
<td>Multi-active culture</td>
<td>different variants of the global schema for parallel integrated work</td>
<td>(d),(c)</td>
</tr>
<tr>
<td>Reactive culture</td>
<td>completely fledged schemata with all details and views for later work</td>
<td>(d)</td>
</tr>
</tbody>
</table>
Parametric Views Supporting Adaptation

An abstract example of a view in the higher-order entity-relationship model with five types $T^i_V$ that are defined five queries $q^i_V$ which use base types $T^1_1, ..., T^1_{k_1}, T^2_1, ..., T^2_{k_2}, T^3_1, ..., T^3_{k_3}$ where the two relationship types are defined through joins of their component types.
Achievements Reached So Far

IS usage and development is communication among Human(s) and Computer(s)

Layered and networking structure of cultures

User modelling

Story and content adaptation

- **Cultures of user and intention**
  - EJC 2014

- **Culture and other context**
  - parametric database schemata and views adaptable to culture

- **Culture-aware content**

- **Web information system**

- **Culture-aware usage and story space**
  - parametric mini-stories adaptable to user culture

- **Culture-aware functionality**
  - (Forthcoming paper)

- **Culture-aware presentation**
  - EJC 2014
Thank you!
Database Constraints - A Survey

Logics for Dependence and Independence
Dagstuhl
22.6.2015

Bernhard Thalheim
Technologie der Informationssysteme
Institut für Informatik, Christian-Albrechts-Universität zu Kiel, BRD
Kolmogorow-Professor e.h. der Lomonossow-Universität Moskau
Observations

- Right logic in the best-fitted structure, in the best complexity and micro-structure, fitted to its purpose and to the user
  - Counterexamples
    - Tuple calculus in relational database theory with the nonsense of safe formula
    - Null value logics without consideration of nil semantics and pragmatics
- State of the art:
  - Relational databases: poor structuring and superficial, incomprehensible classification of constraints
    - with many non-axiomatisation or non-decidability results
  - ER schemata: implicit structuring semantics
- Constraints with insufficient expressive power despite the high variety, e.g., cardinality constraints cannot express finiteness of cycles
Distinguishability and Identifiability

- Objects with their way of identification.
  Value-representability is a must.
- Objects with their properties (intext).
- Objects with their context
  e.g. relationships to other objects, concerns, viewpoints
- General management of worlds, e.g., relationships with properties also many-dimensional, multi-layered
- Detecting cycles is a nightmare. *Cyclic structuring* leads to non-first-order logics. Structures with abstract linking are potentially cyclic.
- The OID has already been implemented by relational databases: tuple identifier. The OID is too expressive.

Object-oriented approaches have improved and damaged culture at the same time.

Next after OO: components!!!!
**Pearls of Structuring**

Lessons Learned for the Good

Hierarchical structuring of types leads to a generalized first-order predicate logics.

**Optionality is difficult** and is handled by

- **weak semantics** for object sets $\mathcal{O}$ with $\text{compon}^{opt}(\mathcal{O}) \neq \emptyset$

  not true: $\mathcal{O} \models \{\alpha_1, \ldots, \alpha_m\}$ iff $\mathcal{O} \models \{\alpha_i\}$

- **strong semantics** for object sets $\mathcal{O}$ with $\text{compon}^{opt}(\mathcal{O}) \neq \emptyset$

  in general: $\mathcal{O} \not\models \alpha \rightarrow \alpha$

  $\mathcal{O} \models \alpha \rightarrow \beta$ and $\mathcal{O} \models \beta \rightarrow \gamma$ do not imply $\mathcal{O} \models \alpha \rightarrow \gamma$.

**Axiomatizability:** Implication operator $\Phi^*\models$ reflexive, monotone, closed and compact $\iff$ deductive system $\Gamma$ $\Phi_{\Gamma} \equiv \Phi_{\models}$

$\Phi_{\models}$ inference, deduction properties and generalization invariant:

$\Phi_{\Gamma^*(\emptyset)} = \Phi_{\models}(\emptyset)$
Constraints

Insufficient Expressivity of Database Models

- Functional, key, inclusion and exclusion dependencies are constraints that are natural in the relational model.
- Key-based inclusion dependencies are handicapped.
- Multi-valued dependencies are far better expressed in the ER model.
- Cardinality constraints are overloaded.
  - better treat maximal and minimal cardinality in separate systems.
- Integrity enforcement is treated separately for types.
- Join dependencies are representational constraints.
# Static Integrity Constraints

<table>
<thead>
<tr>
<th>Layer</th>
<th>Conceptual Layer</th>
<th>Implementation Layer</th>
<th>User Layer</th>
<th>Conceptual Layer</th>
<th>Implementation Layer</th>
<th>User Layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Partial identification</td>
<td>functional, equality-generating</td>
<td>key, uniqueness, trigger, check</td>
<td>identification</td>
<td>relative independence</td>
<td>decomposition, stored procedures, trigger</td>
<td>structure !</td>
</tr>
<tr>
<td>relative independence</td>
<td>multivalued, hierarchical, join, exclusion, tuple-generating constraint, horizontal decomposition</td>
<td>no null, stored procedures, trigger</td>
<td></td>
<td>existence constraint</td>
<td>null-value-free, union constraint, numerical, cardinality</td>
<td>no null</td>
</tr>
<tr>
<td>existence constraint</td>
<td></td>
<td></td>
<td></td>
<td>redundancy constraint</td>
<td></td>
<td></td>
</tr>
<tr>
<td>redundancy constraint</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Kinds of semantics: lexical semantics, grammatical semantics, statistical semantics, logical semantics, prototype semantics, program semantics, dynamic semantics, flexible semantics: which kind of interpretation, with(out) context abstraction, states considered, configuration by association of state with context, interpretation function, evaluation functions, elaboration function, resulting in a large variety ranging from everyday semantics to canonical mathematical semantics, from open world to closed world, from finiteness restriction to infiniteness tolerance, computer science specifics: operational, denotational, axiomatic, transformational, algebraic, macro-expansion, and grammar semantics

see Schewe/Thalheim SDKB 2010 and 2011

NULL markers as the invention of the devil

Wrong applications in database theory

Pseudo-semantic web without semantification

Special logics e.g. concept logics, knowledge, information

Transaction logics
Restrictions of This Talk 2/3

Not considered but ready for further discussion

Dynamic constraint sets (classically only static constraint sets)

The HERM schema of the Conference Database after paper assignment
Restrictions of This Talk 3/3

First part: Should be a talk given by Miika and Juha!

Second-order logics

Dependence logic that axiomatises
sets of functional and inclusion constraints

Monadic logics

Faithfulness of constraint sets versus

closed-world specification (all valid constraints)
error-proneness of constraint sets
robust sets of constraints

Natural constraint sets despite classes of specific constraints
Optimisation of Behaviour Through Normalisation of Database Structuring

Normalisation as a vehicle for performance improvement

(A) Redundancy problematic whenever additional actions are required;
(B) Blocking of management due to the information capacity of the schema (e.g., insertion anomaly);
(C) Information loss after database modification whenever data are eagerly deleted (e.g., deletion anomaly);
(D) Evolution sensitivity and instability of the database due to changes;
(E) Different abstractions at the same time (e.g., views, derived attributes, logs as part of structures);
(F) Performance problems caused by integrity constraint maintenance (e.g., update anomalies);
(G) Expensive maintenance, operating and modification due to consistency maintenance.
The Constraints Handling Framework

(1) **Specification level**: description of the constraints property, validation, policies for evaluation, specific policies, transformations of constraint properties to others;

(2) **Control or technical level**: application of the constraint constraint property portfolio, techniques and methods (e.g., cascade, restrict, no action, defaultify nullify);

(3) **Application or technology level**: management of constraint handling

(4) **Establishment or organisational level**: based on a methodology and constraint maintenance system.

Level *five*: facilities for handling satisfaction of constraints and for predicting changes of satisfaction

Level *six* optimises the constraint management;

Level *seven* uses experiences for innovation and adaptation.
Constraints: Forgotten Properties

- Functional dependencies: \( \forall x_1, x_2 (\psi(x_1, x_2) \rightarrow \phi(x_1, x_2)) \), expressible through equality sets, invariance properties for subsets, sensitive against supersets, pair algebra, two-tuple-implication

- Hierarchical dependencies: reasoning through the tableau calculus, not invariant for most operations, semantical gap between FD’s and MVD’s, variety of definitions

- Implicit model assumptions: component inclusion condition, identifiability whenever set semantics, cycles and finiteness, superficial structural representation

- Normalization as structural instead of behavioral optimization

- Schema transformation and equivalence
Constraints: Supplanted Properties

- Finiteness properties: finite implication, finite calculi, finite representation of potentially infinite sets, finite computation (‘safety’)
- Parallel execution: transaction based concurrent semantics, causal semantics of processes, predicative semantics for engines
- Unsafety of data: null values (overloaded, logics, computation, identification), fuzzy data with error model, aggregated macro-data versus micro-data
- Completeness of specification: infeasible for humans, normal forms based on completeness, order dependence of (normalization) algorithms, time-dependence of constraints validity, real-life constraints
- Weak constraints: deontic maintenance, temporal maintenance, default values
Normal Forms

Maintenance through keys, domain and referential constraints

Forbidden substructures used for the definition of normal forms, e.g.,

3NF: $Z \rightarrow \{A\} \in \Sigma^+, \ A \not\in Z$, $A$ non-key attribute $Z \rightarrow U \not\in \Sigma^+$

Boyce-Codd-Normalform: $Z \rightarrow \{A\} \in \Sigma^+, \ A \not\in Z$, $Z \rightarrow U_i \not\in \Sigma^+$

4NF: $Z \rightarrow X \in \Sigma^+, \ X \not\subseteq Z$, $Z \rightarrow U \not\in \Sigma^+$

5NF: $(Y_1, ..., Y_k) \in \Sigma^+, \ \exists j : Y_j \rightarrow U_i \not\in \Sigma^+$

Project-Join-NF: $(Y_1, ..., Y_k) \in \Sigma^+, \ \{X \rightarrow U_i \in \Sigma^+\} \neq (Y_1, ..., Y_k)$

Inklusion-NF: $R_i[X] \subseteq R_j[Y] \in \Sigma^+, \ Y \rightarrow U_j \not\in \Sigma^+$

Domain-Key NF (DKNF): $\alpha \in \Sigma^+$, non-trivial, $\Sigma_K \not\models \alpha$

more than 35 other useful normal forms have been introduced

Synthesis and decomposition algorithms
# Solved and Open Problems

<table>
<thead>
<tr>
<th>class</th>
<th>axiomatiz.</th>
<th>complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Hilbert/Gentzen</td>
<td>worst</td>
</tr>
<tr>
<td>keys</td>
<td>(\sqrt{\text{Hilbert}})</td>
<td>(\checkmark)</td>
</tr>
<tr>
<td>keys (bounded domains)</td>
<td>(\sqrt{\text{Hilbert}})</td>
<td>(\checkmark)</td>
</tr>
<tr>
<td>keys (with nulls)</td>
<td>(\sqrt{\text{Hilbert}})</td>
<td>(\checkmark)</td>
</tr>
<tr>
<td>keys NF(^2)</td>
<td>? (\sqrt{\text{LogChar}}) (hierarch.)</td>
<td>????</td>
</tr>
<tr>
<td>fd's</td>
<td>(\sqrt{\text{Hilbert}}, \sqrt{\text{Graph}})</td>
<td>(\checkmark)</td>
</tr>
<tr>
<td>Hungarian deps</td>
<td>(\sqrt{\text{LogChar}})</td>
<td>??</td>
</tr>
<tr>
<td>multivalued deps</td>
<td>(\sqrt{\text{Hilbert}}, \sqrt{\text{Graph}})</td>
<td>????</td>
</tr>
<tr>
<td>hierarchical deps.</td>
<td>(\sqrt{\text{Hilbert}}, \sqrt{\text{Graph}})</td>
<td>??</td>
</tr>
<tr>
<td>join deps.</td>
<td>(\sqrt{\text{Gentzen}})</td>
<td>??</td>
</tr>
<tr>
<td>cardinality constr.</td>
<td>(\sqrt{\text{Charact.}})</td>
<td>??</td>
</tr>
<tr>
<td>inclusion/exclusion</td>
<td>(\sqrt{\text{Hilbert}})</td>
<td>????</td>
</tr>
<tr>
<td>transition constr.</td>
<td>????</td>
<td>???</td>
</tr>
</tbody>
</table>

Warning: Hilbert-/Gentzen-kind calculi are only one option!
Problems: Implicit Model-Inherent Integrity Constraints

Typical implicit model-inherent integrity constraints:

*Component-construction constraints* existence, cardinality and inclusion of components;

*Identification constraints* implicitly used for the set constructor value-representability or weak-value representability;

*Acyclicity and finiteness* of structuring cardinality constraints may be based on potentially infinite cycles;

*Implicit meaning of constraints*, e.g., value construction, optional/mandatory existence, derived value/object, identification, union, combination/association;

*Superficial structuring* leads to representation of constraints through structures.
**However: Enrichment**

*Lessons of implicit definition*

Component-construction constraints: existence, cardinality, inclusion impact on translation and implication

Identification constraints: set, list constructors

Acyclicity and finiteness of structuring supports axiomatization and definition of the algebra

Superficial structuring leads to representation of constraints through structures

Implicit model-inherent constraints belong to the performance and maintenance traps.
Problems: Modelling Problems with Constraints

**Representable by structuring** e.g. join dependencies

\[ X \rightarrow Y | Z \]

\[ Y \cap Z = \emptyset, \quad X \cup Y \cup Z = R \]

another relational representation by decomposition:

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Humans** Do you reason abstractly?

Can you understand logics?

Do you understand your colleagues semantics?

**Systems** Are there systems which use integrity constraints beyond keys, referential integrity and domain constraints?
Problems: Computational Problems with Constraints

Combinatorial complexity is exponential
necessary complete set of constraints for normalization, optimization
either we need to know all valid constraints (CWA) or to know constraints which are valid and invalid

Too simple properties simple axiomatisation, ...
“too simple” is misunderstood and taken for granted, also intentionally extended to all other cases

Maintenance of consistency full of mis-conceptions
triggers are only and only applicable to strict hierarchical constraint sets
but active database people believe in triggering
solution: greatest consistent specialization approach to refinement of operations

Object-identification through identifiers has to be supported by value identification
thus XML will never have a computational query algebra
Combinatorial Problems with Constraints 1

Maximal number of minimal keys (sharp)

\[ \binom{n}{\lfloor \frac{n}{2} \rfloor} \]

(J. Demetrovics (1979))

Restricted domains

\[ \left| \text{dom}(B_i) \right| \leq k \quad (1 \leq i \leq n) \]

\[ k^4 < 2n + 1 \]

\[ \left( \binom{n}{\lfloor \frac{n}{2} \rfloor} - \lfloor \frac{n}{2} \rfloor \right) \]

(\(\beta\) (1984))

same behavior on non-uniform domains

similar behavior in presence of null values

(Selesnev, \(\beta\) (1985))
Combinatorial Problems with Constraints 2

Functional dependencies
maximum number $N(n)$ of basic functional functional dependencies for schemata on $n$ attributes

$$2^n \left(1 - \frac{4 \log_2 \log_2 n}{\log_2 e \log_2 n}\right) (1+o(1)) \leq N(n) \leq 2^n \left(1 - \frac{\log_2^3 n}{150\sqrt{n}}\right)$$

minimal generating sets of fd's $(n$ odd; $n$ even (replace $\frac{1}{n^2}$ by $\frac{1}{n^2 2n^8}$))

$$\left(\frac{n}{\lfloor \frac{n}{2} \rfloor}\right) + \frac{1}{n^2} \left(\frac{n}{(n+3)\frac{2}{2}}\right) \leq M(n) \leq 2^n \left(1 - \frac{\log_2^3 n}{150\sqrt{n}}\right)$$

closed families of fd's

$$2^{\left\lfloor \frac{n}{2} \right\rfloor} \leq Cl(\mathcal{F}, n) \leq 2^{\left\lfloor \frac{n}{2} \right\rfloor}(1+o(1))$$

(J. Demetrovics, G.O.H. Katona, D. Miklos, $\beta$ (1982-2006))
Combinatorial Problems with Constraints 3

rather than describing the entire set of basic functional dependencies use a relation $R^C$ which allows to reason on the set of constraints

Armstrong relations - size

$$\frac{1}{n^2} \binom{n}{\lfloor \frac{n}{2} \rfloor} \leq L_{\{key\}}(n) \leq \binom{n}{\lfloor \frac{n}{2} \rfloor} + 1$$

$$c_1 n^{\frac{k-1}{2}} \leq L_{\{key(k)\}}(n) \leq c_2 n^{\frac{k-1}{2}}$$

$$\frac{1}{n^2} \binom{n}{\lfloor \frac{n}{2} \rfloor} \leq L_F(n) \leq \left( \binom{n}{\lfloor \frac{n}{2} \rfloor} (1 + \frac{c_3}{\sqrt{n}}) \right)$$

(J. Demetrovics, G.O.H. Katona & Hungarians, $\beta$ (1982-2005))
Good Case however: Average Complexity of Constraints 1

conjecture: average relation has a key system of polynomial size

average relation: almost all minimal keys are of same length

\[ 2 \log_{||D||}(R^C) \]

Poisson distribution

deviation is constant

if \( X \) is of length \( 2 \log_{||D||}(R^C) \) then \( X \) is minimal

deviation decreases with increasing domain sizes

same behaviour for the non-Bernoulli case

small fraction of relations with large key systems

similar results for functional dependencies

(J. Demetrovics, G.O.H. Katona, D. Miklos, O. Selesnev, \( \beta \), 1995-2005)
Why Constraints

Visuality

Triggering

Identity

Open Publications

Good Case however: Average Complexity of Constraints 2

Dependence on domain size

Behavior of the Key Probability (Frequency Polygon)
Good Case however: Average Complexity of Constraints 3

average length \( av_n(l, 2) \) of minimal keys in relations with \( m \) tuples

\[
\left\lfloor \log_2 m \right\rfloor \leq av_n(m, 2) \leq 2\left\lfloor \log_2 m \right\rfloor .
\]

random relation

\[
P(X \rightarrow \{A_j\}, l) = \begin{cases} 
0, & \quad \text{if} \ 2\log l - \\
\sum_{A_i \in X} H_2(\kappa_i) \rightarrow +\infty , \\
e^{-2^{a-1}(2^{-H_2(\kappa_j)}-1)}, & \quad \text{if} \ 2\log l \\
- \sum_{A_i \in X} H_2(\kappa_i) \rightarrow a , \\
1, & \quad \text{if} \ 2\log l - \\
\sum_{A_i \in X} H_2(\kappa_i) \rightarrow -\infty .
\end{cases}
\]

If \( X \) is a set of attributes of size definitely larger than \( \frac{2\log l}{H_2} \) then \( X \rightarrow \{A_j\} \) holds with high probability for any \( A_j \).
Problems: Logical Problems Restricting Reasoning based on Constraints

(un-)decidable implication

$P, NP, PSpace, ExpSpace, ...$

$\models = \models fin$ or not

Axiomatization or not

Hilbert type

Gentzen type

Properties of constraint sets

- decidable for tuple-generating and equality-generating dependencies
- undecidable for embedded JD’s
- undecidable for embedded MVD’s
- undecidable for FD’s and ID’s

Axiomatizability of constraint classes

- FD’s and ID’s are not axiomatizable
  
  *but weak FD’s and ID’s axiomatizable*

- JD’s not axiomatizable

  but tuple-generating and equality-generating dependencies are axiomatizable

28
“Tuple-generating” constraints are in reality existence constraints
value must (not) exist; also in domain, referenced type (inclusion/exclusion/component),
potentially also in same class (redundancy constraint)
multivalued and hierarchical dependencies
hidden properties, e.g., subset property

“Equality-generating” constraints are in reality binding constraints
value must/might (not) be equal, is potentially derivable,
generalised to \((p,q)\) constraints, used for cardinality constraints

Cycle constraints are not yet developed
cycle must/might (not) exist, cycles in values (‘marriage’, ‘friend’), cycle in types, cycles in value sets

Logical gap between existence and binding constraints

Derived constraints for derived structures
Kinds of constraints, e.g., explicit declaration, coupling, semantic association, semantic unit, structural representability
The Power of Functional Dependencies

<table>
<thead>
<tr>
<th>Functional dependencies</th>
<th>Schema after decomposition</th>
</tr>
</thead>
<tbody>
<tr>
<td>{A} \rightarrow {B}</td>
<td>Schema 1</td>
</tr>
<tr>
<td>{A} \rightarrow {B} \rightarrow {C}</td>
<td>Schema 1'</td>
</tr>
<tr>
<td>{A} \rightarrow {B, C}</td>
<td>Schema 2</td>
</tr>
<tr>
<td>{A} \leftrightarrow {B} \rightarrow {C}</td>
<td>Schema 1, Schema 1'</td>
</tr>
<tr>
<td>{A} \leftrightarrow {B, C}</td>
<td>Schema 3</td>
</tr>
<tr>
<td>{A} \leftrightarrow {B} \leftrightarrow {C}</td>
<td>Schema 3</td>
</tr>
<tr>
<td>{A} \rightarrow {B} \leftrightarrow {C}</td>
<td>no new schema</td>
</tr>
<tr>
<td>{A, B} \rightarrow {C}</td>
<td>Schema 1, Schema 1'</td>
</tr>
<tr>
<td>{A} \leftrightarrow {B}</td>
<td></td>
</tr>
</tbody>
</table>

Why Constraints
Visuality
Triggering
Identity
Open
Publications

Content
Information
Concept
Topic
FD-Axiomatisation for Tuple and Set Constructors

Given \(X, Y, Z \subseteq \text{Sub}(T)\)

**reflexivity axiom:** \(\frac{X \rightarrow Y}{Y \subseteq X}\)

**sub-attribute axiom:** \(\frac{\{X\} \rightarrow \{Y\}}{Y \leq X}\)

**extension rule:** \(\frac{X \rightarrow Y}{X \rightarrow X \cup Y}\)

**restricted meet rule:** \(\frac{X \rightarrow Y}{X \rightarrow X \cap T Y}\)

\[\exists X', Y' \geq X, Y : X' \cdot Y' = X \cap Y = X' \cap Y'\]

**transitivity rule:** \(\frac{X \rightarrow Y, Y \rightarrow Z}{X \rightarrow Z}\)

**Join (generalised \(\cap\)):**
- \(Z\) sub-attribute of \(Y\) for \(Y, Z \in \text{sub}(X)\): \(Y \cap_X Z = Z\)
- \(X = A\{B\}, Y = A\{C\}, Z = A\{D\}, C, D \leq B : Y \cap_X Z = A\{C \cap_B D\}\)
- \(X = A(A_1, ... A_n), Y = A(B_1, ..., B_n), Z = A(C_1, ..., C_n), B_i, C_i \leq A_i (\forall i) : Y \cap_X Z = A(B_1 \cap_{A_1} C_1, ..., B_n \cap_{A_n} C_n)\)

**Meet (generalised \(\cup\)):**
- \(Z\) sub-attribute of \(Y\) for \(Y, Z \in \text{sub}(X)\): \(\bullet_X Z = Y\)
- \(X = A\{B\}, Y = A\{C\}, Z = A\{D\}, C, D \leq B : Y \bullet_X Z = A\{C \bullet_B D\}\)
- \(X = A(A_1, ... A_n), Y = A(B_1, ..., B_n), Z = A(C_1, ..., C_n), B_i, C_i \leq A_i (\forall i) : Y \bullet_X Z = A(B_1 \bullet_{A_1} C_1, ..., B_n \bullet_{A_n} C_n)\)

Example: \(T = \text{Library}\{\text{Lending}(\text{Student, Book})\}\)

\[T^C = \{t_1, t_2\}, \quad t_1 = \{\text{Alf, Math}, \text{Betty, DB}\}, \quad t_2 = \{\text{Betty, Math}, \text{Alf, DB}\}\]

\(\models_{T^C} \{\text{Library}\{\text{Lending}(\text{Student})\} \rightarrow \{\text{Library}\{\text{Lending}(\text{Book})\}\}\}

\(\not\models_{T^C} \{\text{Library}\{\text{Lending}(\text{Student})\} \rightarrow \{\text{Library}\{\text{Lending}(\text{Student, Book})\}\}\} \)
FD-Axiomatisation in Extended HERM

Classical: tuple and list constructors

\[
\frac{X \rightarrow Y}{Y \subseteq X}, \quad \frac{X \rightarrow Y}{X \rightarrow X \sqcup T Y}, \quad \frac{X \rightarrow Y, Y \rightarrow Z}{X \rightarrow Z}
\]

Extended HERM: tuple, list, set, and multiset constructors

\[
\mathcal{X}, \mathcal{Y}, \mathcal{Z} \subseteq \text{Sub}(T)
\]

Axioms

\[
\frac{\mathcal{X} \rightarrow \mathcal{Y}}{\mathcal{Y} \subseteq \mathcal{X}}, \quad \frac{\{X\} \rightarrow \{Y\}}{Y \leq X}, \quad \frac{\{X, Y\} \rightarrow \{X \sqcup N Y\}}{X, Y \text{ reconcilable}}
\]

Rules

Extension \[
\frac{\mathcal{X} \rightarrow \mathcal{Y}}{\mathcal{X} \rightarrow \mathcal{X} \sqcup \mathcal{Y}}
\]

Transitivity \[
\frac{\mathcal{X} \rightarrow \mathcal{Y}, \mathcal{Y} \rightarrow \mathcal{Z}}{\mathcal{X} \rightarrow \mathcal{Z}}
\]

reconcilable: have inductively the same structuring for tuple and multiset constructors or \( Y \leq X \) or \( X \leq Y \)

\( Y \leq X \): there exist a projection from \( X \) to \( Y \)

\( X \sqcup_T Y \): lattice operation on \( T \)-subtypes

©Hartmann/Link DEXA 2005
Functional Dependencies

*Not as well understood as it seems!*

Explicit declaration of partial identification: explicitly declaring a functional association $X \xrightarrow{\text{Ident}} Y$.

Tight functional coupling: numerical constraints i.e., $X \xrightarrow{\text{Num}} Y$.

Another denotation is based on cardinality constraints.

Semantic constraint specific for the given application: application has a limited scope and allows us to strengthen the constraint $X \xrightarrow{\text{Sem}} Y$.

Semantical unit with functional coupling: Semantical units are those reducts of a type that are essential in the given application $X \xrightarrow{\text{Unit}} Y$.

Structural association among units: Semantical units may allow a separation of concerns for certain elements $X \xrightarrow{\text{Struct}} Y$. 
### Functional Dependencies

**Not as well understood as it seems!**

\[
\begin{array}{c}
T_A \\
\text{A} \\
\text{C}
\end{array} \xleftarrow{(1,1)} \xrightarrow{T_A\text{.to.}T_B} \begin{array}{c}
T_B \\
\text{B} \\
\text{D}
\end{array}
\]

\{A\} \rightarrow \{C\}: explicit and direct dependency.

\{A\} \rightarrow \{B\} is an inter-type constraint and leaves the scope of type \( T_A \).

<table>
<thead>
<tr>
<th>Constraint Type</th>
<th>Instantiation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explicit declaration</td>
<td>( T_B = \text{StudyProgram, } B = \text{ProgramCode, } D = \text{ProgramName} )</td>
</tr>
<tr>
<td>Tight coupling</td>
<td>( T_A = \text{Student, } T_A\text{.to.}T_B = \text{MajorProgram, } T_B = \text{StudyProgram} )</td>
</tr>
<tr>
<td>Semantic constraint</td>
<td>( T_B = \text{StudyProgram, } B = \text{ProgramCode, } D = \text{ResponsibleProfessor} )</td>
</tr>
<tr>
<td>Semantical unit</td>
<td>( T_B = \text{StudyProgram, } B = \text{ProgramCode, } D = \text{ProgramDegree} )</td>
</tr>
<tr>
<td>Structural association</td>
<td>( T_A = \text{Student, } T_B = \text{RegisteredStudent, } A = \text{PersonID, } B = \text{StudentNr} )</td>
</tr>
</tbody>
</table>
Axiomatisation follows the Classical Approach

**Augmentation** e.g.,

\[
\begin{align*}
R : X \xrightarrow{\text{Ident}} & Y \sqcup_R Y' \\
R : X \sqcup_R X' \xrightarrow{\text{Ident}} & Y
\end{align*}
\]

**Trivialisation of identification**

\[
\begin{align*}
R : X \xrightarrow{\text{Sem}} & Y \sqcup_R Y' \\
R : X \sqcup_R X' \xrightarrow{\text{Sem}} & Y
\end{align*}
\]

**Adaptation of semantics scope**

**Implication** e.g.:

\[
\begin{align*}
R : X \xrightarrow{\text{Ident}} & Y, R : Y \xrightarrow{\text{Sem}} Z \\
R : X \xrightarrow{\text{Ident}} & Z
\end{align*}
\]

\[
\begin{align*}
R : X \xrightarrow{\text{Sem}} & Y, R : Y \xrightarrow{\text{Ident}} Z \\
R : X \xrightarrow{\text{Sem}} & Z
\end{align*}
\]

Axiomatisation as pair algebra and by lifting among types.

Basis: subset, substructuring, implication properties of FD’s.
Key Dependencies

Descriptive meanings of the key concept depending on the levels

- **Language definition level**: uniqueness
  - expressed via identification
  - existence property: secondary role
- **External level**: uniqueness inherited
  - integrity and accessibility concepts not under consideration
- **Conceptual level**: identification and integrity
  - surrogate concept, key dependency
- **Internal level**: uniqueness covered by identification
  - additionally of interest: restrictions like invariant values in keys, accessibility
Explicit and Excluded Functional Dependencies

\( X \not\rightarrow Y \): the functional dependency \( X \rightarrow Y \) is not valid.

**Axioms**

\[ X \cup Y \rightarrow Y \]

**Rules**

\( X \rightarrow Y \)

\[ X \cup V \cup W \rightarrow Y \cup V \]

\( X \rightarrow Y, \ X \rightarrow Z \)

\[ Y \rightarrow Z \]

\( X \rightarrow Y \)

\[ X \rightarrow Y \cup Z \]

\( X \rightarrow Y \)

\[ X \rightarrow Y \cup Z \]

\( X \rightarrow Y \)

\[ X \rightarrow Y \cup Z \]

\( X \rightarrow Z \)

\[ X \rightarrow Y \cup Z \]

\( Y \rightarrow Z \)

\[ X \rightarrow Y \]

\( X \rightarrow Z \)

\[ X \rightarrow Y \]

\( X \rightarrow Z \)

\[ X \rightarrow Y \]
**Multivalued Dependency** \( X \rightarrow\rightarrow Y|Z \)**

Given a relational schema \( R = (U_R, \Sigma_R) \), some relation \( R^C \) on \( R \) subsets \( X, Y \subseteq U_R \) and \( Z = U_R \setminus (Y \cup X) \)

1. **Classical definition**: \( R^C \models X \rightarrow\rightarrow Y|Z \) if for all \( t, t' \in R^C \) with \( t =_X t' \) an object \( t'' \in R^C \) exists with \( t'' =_{X \cup Y} t \) and \( t'' =_{X \cup Y} t' \)

2. **Decomposition** definition: \( R^C \models X \rightarrow\rightarrow Y|Z \) if \( R^C = R^C[X \cup Y] \bowtie R^C[X \cup Z] \)

   We may further conclude: If \( R^C \models X \rightarrow Y \) then \( R^C \models X \rightarrow\rightarrow Y|Z \)

3. **Independence** definition: \( R^C \models X \rightarrow\rightarrow Y|Z \) if \( (\sigma_{X=x}(R^C))[Y] = (\sigma_{(X=x) \land (Z=z)}(R^C))[Y] \) for all \( X \)-values \( x \in R^C[X] \) and all \( Z \)-values \( z \in R^C[Z] \)

**Proof of equivalence**: (1) \( \Rightarrow \) (2) \( \Rightarrow \) (3) \( \Rightarrow \) (1)

E.g., for (1) \( \Rightarrow \) (2) we need to prove: \( R^C \supseteq R^C[X \cup Y] \bowtie R^C[X \cup Z] \)

Given \( t_1 \in R^C[X \cup Y] \) and \( t_2 \in R^C[X \cup Z] \) mit \( t_1 =_X t_2 \)

\( \Rightarrow \) \( \{t_1\} \bowtie \{t_2\} \subseteq R^C \) because \( R^C \models X \rightarrow\rightarrow Y|Z \)
Multivalued Dependency $X \rightarrow\rightarrow Y|Z$

Given a relational schema $R = (U_R, \Sigma_R)$, some relation $R^C$ on $R$ subsets $X, Y \subseteq U_R$ and $Z = U_R \setminus (Y \cup X)$

(4) **Constructor** definition: $R^C \models X \rightarrow\rightarrow Y|Z$

if for all $x \in R^C[X]$ with $(\sigma_{X=x}(R^C))[Y \cup Z] = (\sigma_{X=x}(R^C))[Y] \times (\sigma_{X=x}(R^C))[Z]$  

i.e. $\nu_Z(\nu_{Y \setminus X}(\nu_X(R^C))) = \nu_{Y \setminus X}(\nu_Z(\nu_X(R^C)))$

(5) **Structuring** definition: $R^C \models X \rightarrow\rightarrow Y|Z$

<table>
<thead>
<tr>
<th>$R^C$</th>
<th>${X}$</th>
<th>${Y}$</th>
<th>${Z}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$A_1$</td>
<td>$A_k$</td>
<td>$A_{k+1}$</td>
</tr>
<tr>
<td></td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
</tr>
<tr>
<td></td>
<td>$\ldots$</td>
<td>$\ldots$</td>
<td>$\ldots$</td>
</tr>
</tbody>
</table>

$X = \{A_1, \ldots, A_k\}$, $Y = \{A_{k+1}, \ldots, A_m\}$, $Z = \{A_{m+1}, \ldots, A_n\}$

by a nested relation
Points of View through Separation

Example: Employee, Dependent, Project, Supplier, Product

\{ Employee \} →→ \{ Department, Dependent \}|{ Project, Product, Supplier \} \\
\{ Employee \} →→ \{ Dependent \}|{ Department, Project, Product, Supplier \} \\
\{ Project \} →→ \{ Employee, Department, Dependent \}|{ Product, Supplier \} \\
\{ Product \} →→ \{ Department, Employee, Dependent, Project \}|{ Supplier \}

Definition (6): MVD defined by two (Y,Z) relationship types can be generalised to hierarchical dependencies and n-ary relationship types

\text{DepBasis}(Employee,\Sigma) \setminus \{ \text{Employee} \} = \\
\{}\{} \text{Department }, \{}\{} \text{Dependent } , \{}\{} \text{Project, Product, Supplier } \}\\
\text{DepBasis}(Project,\Sigma) \setminus \{ \text{Project} \} = \\
\{}\{} \text{Product }, \{}\{} \text{Supplier } , \{}\{} \text{Employee, Department, Dependent } \}
Derivation Rules for MVD and FD

\[ X, X', X'', Y, Y', Z, Z', V, W \subseteq U \]

All sets within an MVD constitute a cover of \( U \)

**Axiom:**

\[(1_F) \quad X \cup Y \rightarrow Y \]  \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad (1_M) \quad X \rightarrow \emptyset|Z\]

**Rules:**

\[(21_F) \quad \frac{X}{X \rightarrow Y} \] (FD – MVD reduction)

\[(22_F) \quad \frac{X \rightarrow Y, Y \rightarrow Z}{X \cup V \cup W \rightarrow V \cup Z} \] (FD transitivity)

\[(21_M) \quad \frac{X \rightarrow X' \cup Y|X'' \cup Z}{X \cup X' \cup X'' \rightarrow Z|Y} \] (MVD weakening)

\[(23_M) \quad \frac{X \cup V \rightarrow Y|Z, X \rightarrow Y \cup Z|V}{X \rightarrow Y|Z \cup V} \] (MVD root reduction)

\[(3_{F,M}) \quad \frac{X \cap Y \rightarrow Y \setminus X|X \setminus Y, X \rightarrow Z}{X \cap Y \rightarrow Y \cap Z} \] (FD pushback)

\((1_F), (1_M), (21_F), (22_F), (23_M), (3_{F,M})\) are complete and correct

for functional and multivalued dependencies
Derivation Rules (structurally)

Axiom

\[ X \cup Z \rightarrow X \]

Root reduction

\[ Y_{(X)} \rightarrow X \cup V \quad (X)Z \]

\[ Y \cup Z_{(X)} \rightarrow X \quad (X)V \]

\[ Y_{(X)} \rightarrow X \quad (X)Z \cup V \]

Weakening

\[ X \cup Y_{(X)} \rightarrow X \quad (X)X' \cup Z \]

\[ X \cup X' \cup X'' \rightarrow X \quad X \]

\[ X \cup X' \cup X'' \rightarrow X \quad X \cup X' \cup X'' \]

Tree restructuring

\[ Y_{(X)} \rightarrow X \quad (X)Z \cup Z' \]

\[ Y_{(X)} \rightarrow X \quad (X)Z \cup Z' \cup Y' \]
Referential Constraints

Generalising foreign key constraints

Relational understanding \( \pi_{\text{list}(A)}(\alpha) \subseteq \pi_{\text{list}(C)}(\beta) \)

1. table / relation type \( \alpha \) with primary key \( A \)
   - either artificial ID or surrogate key or value key
2. \( B \) is a key in \( \beta \)
3. \( C \) can be property based
   generalise to \( \text{view}_{1,\text{list}(A)}(\alpha) \subseteq \text{view}_{2,\text{list}(C)}(\beta) \)

meaning:
- reference as a pointer to already existing data
- reference as a binding means
- reference with master/slave and other protocols

Differentiate introduction and reference
Cardinality Constraints are Global Constraints

Diagram:

- **Trip**
  - Cardinality: \{1\}
  - Visits: \{3,4,7\}
  - Starts in: \{2,3,5,6\}
  - Corrected: \{3\}

- **visits**
  - Cardinality: \{1,2,3,6\}
  - Corrected: \{6\}

- **starts_in**
  - Cardinality: \{2,3,5,6\}
  - Corrected: \{2\}

- **City**
  - Cardinality: \{1,2,3,6\}
  - Corrected: \{6\}
Open Issues

A maximally reduced list.

Coherence constraints, e.g., Zip

Define/use constraints resulting in sophisticated treatment of object identity

similar to mirror typing in hierarchical DBMS

Redundancy maintaining constraints

Explicit handling of incompleteness of constraint set in specifications

Separation of concern for constraints

Explicit specification of enforcement

SPA framework (sensing, processing, actuation)

see also the SDKB 2008 list, list in the HERM book
Questions for the Next Decade

- Complexity of key systems in nested relations
- Complexity of other constraint classes:
  - Multi-valued, hierarchical, join dependencies
  - Inclusion and exclusion constraints
  - Satisfiability of constraint sets
- **Graphical reasoning systems**
- What is known on combinatorial properties of weak constraints?
- Average complexity of Armstrong relations?
- Average complexity of functional dependencies?
- Approaches based on combinatorics or probability theory
“Remembering the Future”

- Semantics with explicit consideration of structuring and type system variants
  (e.g., multivalued dependencies representing structuring and decomposition)
- Combinatorial complexity for holistic consideration
  (i.e., exponential ('worst case'))
- Ability for expressing constraints by users
  (Too natural and not state constraints; difficult to express constraints; abstraction; semantics by different users; everyday life logics, different interpretations (strong, ‘normal’ (logics), weak))
- Computational facilities in systems
  (keys, referential integrity constraints, domain constraints, implementation (physical and/or logical) constraints, views, CASE tool biases)
- Too simple properties (also for axiomatisation) are misleading
  anything in the world has its price we have to pay at the minimum; complex things will remain to be complex
- Logical systems are in general complex
  (not decidable or axiomatisable, deriviation complexity, ...)

Why
Constraints
Visuality
Triggering
Identity
Open
Publications

Content
Information
Concept
Topic
The Simplicity of Graphical Reasoning

Darwen FD rule

\[ X \rightarrow Y_0 Y_1, \ Y_1 Y_2 \rightarrow W \]
\[ XY_2 \rightarrow Y_0 Y_1 W \]
Graphical FD Reasoning

\[(S) \quad Y \rightarrow B \quad YC \rightarrow B\]

\[(Q) \quad Y \rightarrow A, Y \not\rightarrow B \quad YA \not\rightarrow B\]

\[(T) \quad Y \rightarrow A, YA \rightarrow B \quad Y \rightarrow B\]

\[(R) \quad YA \rightarrow B, Y \not\rightarrow B \quad Y \not\rightarrow A\]

\[(P) \quad YC \not\rightarrow B \quad Y \not\rightarrow B\]

\[\Box \quad \neg(Y \rightarrow B, Y \not\rightarrow B)\]

Resulting rules for graphical reasoning
The Simplicity of Graphical Reasoning

$$U_R = \{A, B, D, F, G, I\}$$
$$\sigma_R = \{A \rightarrow IG, D \rightarrow FG, IAB \rightarrow D, IF \rightarrow AG\}$$

Classical synthesis algorithms:

$$R_1 = (\{A, G, I\}, \{A \rightarrow GI\})$$
$$R_2 = (\{A, F, I\}, \{A \rightarrow I, FI \rightarrow A\})$$
$$R_3 = (\{A, B, D\}, \{AB \rightarrow D\})$$
$$R_4 = (\{D, F, G\}, \{D \rightarrow FG\})$$

This normalisation not minimal

Instead of $$R_1$$ take $$R'_1 = (\{A, G\}, \{A \rightarrow G\})$$.

$$R_2$$ is not in BCNF. It cannot be split into two relation schemata.
The Simplicity of Graphical Reasoning

Is D in any minimal key?
What are the minimal keys?
AB, BC, BE, CF?
Is Date correct?

Difficult but possible

What are alternative reductions?
Which dependencies are central?
Another FD Example
Another FD Example

```plaintext
BE → C → CE
  ↓     ↓    ↓
  B     ACD  CF
  ↓     ↓    ↓
  BC     E    D
  ↓     ↓    ↓
  E       D    F

BE → C → CE → A
  ↓     ↓    ↓
  B     ACD  CF
  ↓     ↓    ↓
  BC     E    D
  ↓     ↓    ↓
  E       D    F
```

(1) derived

(2) derived

(3) derived
An another FD Example

A directly reduced set

Another irreducible set

Derivation of \( \{A, C, D\} \rightarrow \{B\} \) as two step procedure
Graphical FD !!Set!! Reasoning

|        | |Y|=1 | |Y|=2 |
|--------|-----------------|-----------------|
| (S)    | ![Diagram](#)   | ![Diagram](#)   |
| (T)    | ![Diagram](#)   | ![Diagram](#)   |

Why Constraints Visuality Triggering Identity Open Publications

Content Information Concept Topic
Additional cardinalities:
\[
\text{card}(\text{Flies}[\text{Gate}, \text{Time}, \text{Flight}], (\text{Gate}, \text{Time})) = [0,1]
\]
\[
\text{card}(\text{Flies}[\text{Gate}, \text{Time}, \text{Flight}], (\text{Flight}, \text{Time})) = [0,1]
\]
potentially \[
\text{card}(\text{Flies}[\text{Time}, \text{Flight}], (\text{Flight})) = [0,1]
\]
As FD’s:
\[
\{ \text{Flight Time} \rightarrow \text{Pilot}, \text{Time Pilot} \rightarrow \text{Flight}, \text{Flight} \rightarrow \text{Time}, \text{Gate Time} \rightarrow \text{Flight}, \text{Flight Time} \rightarrow \text{Gate} \} \]
Is this set complete or do we need other constraints?
The Triangular Representation

Trivial or redundant FD’s not represented, eg.:

- \( \text{Flight} \rightarrow \text{Flight}, \text{Flight} \rightarrow \text{Flight Time} \)
- \( \text{Flight} \rightarrow \text{Time Pilot} \iff \{\text{Flight} \rightarrow \text{Time}, \text{Flight} \rightarrow \text{Pilot}\} \)

Example ternary case:

Initial set of FD’s: filled circles

- \( \text{Flight Time} \rightarrow \text{Pilot} \) 2D
- \( \text{Time Pilot} \rightarrow \text{Flight} \) 2D
- \( \text{Flight} \rightarrow \text{Time} \) 1D

There is a sound and complete axiomatisation!
Graphical Reasoning for Ternary Cases

Sound and complete implication system: \((A \neq B; \ A, B \notin Y)\)

\((S)\) Extension: \(\frac{Y \rightarrow A}{YB \rightarrow A}\)

\(Y \rightarrow A\) is extended by \(B\)

\((T)\) Rotation: \(\frac{Y \rightarrow A, YA \rightarrow B}{Y \rightarrow B}\)

\(Y \rightarrow A\) is rotated around \(Y\)

\(\Rightarrow\) supported by \(YA \rightarrow B\)

Example ternary case:

Implied set of FD’s: empty circles

\(\vdash_{(S)} \circ Flight\ Pilot \rightarrow Time\)

\(\vdash_{(T)} \circ Flight \rightarrow Pilot\)
Incorporating Negated Dependencies

Sound and complete implication system: \((S), (T) + (P)\)

\[
\begin{align*}
\frac{YB \rightarrow A}{Y \rightarrow A} & \quad (P) \\
\frac{Y \rightarrow B, Y \rightarrow A}{YA \rightarrow B} & \quad (Q) \\
\frac{Y \rightarrow B, YA \rightarrow B}{Y \rightarrow A} & \quad (R)
\end{align*}
\]

Reduction | Extension | Rotation

Example ternary case:

- \(Flight \rightarrow Time\)
- \(Time \rightarrow Pilot\)
- \(Flight \rightarrow Time\)
- \(Time \rightarrow Pilot\)

\[]((S)) \circ Flight \rightarrow Time\)
\[]((T)) \circ Flight \rightarrow Pilot\)
\[]((R)) \circ Time \rightarrow Flight\)
Graphical Reasoning for Ternary Cases

Example ternary case:

```
Flight
   /\   /
 /     \
Pilot   Time
```

The closed set:

- Flight Time → Pilot
- Time Pilot → Flight
- Flight → Time
  - Flight Pilot → Time
  - Flight → Pilot

The result is one of the 14 different ternary cases (schema / relationship types).

What about raising the number of attributes?
The Tetrahedral Representation

A quaternary case has \(\binom{4}{3} = 4\) nested ternary cases forming the surface of a *tetrahedron*.

Edges correspond to the \(\binom{4}{2} = 6\) nested binary cases, each shared between two triangles.

The initial set:

- *Flight Time* → *Pilot*
- *Time Pilot* → *Flight*
- *Flight* → *Time*
- *Gate Time* → *Flight*
- *Flight Time* → *Gate*
The Quadratic Representation

Ternary case - triangle 2D
Quaternary case - square 2D

Example complete case:

The initial set:
- Flight Time → Pilot
- Time Pilot → Flight
- Flight → Time
- Gate Time → Flight
- Flight Time → Gate

Key features: 'points towards', 'parallel'
Graphical Reasoning by the ST Algorithm

Example (full):

FD’s with minimal left sides:
- Flight → Time
- Flight → Pilot
- Time Gate → Pilot
- Flight → Gate
- Time Pilot → Gate
- Time Pilot → Flight
- Gate Time → Flight

The result is one of the 165 different quaternary cases (schema / relationship types).
The Pentagonal Representation

A quinary case has \( \binom{5}{4} = 4 \) nested quaternary, \( \binom{5}{3} = 10 \) nested ternary and \( \binom{5}{2} = 10 \) nested binary cases.

Derivation by extension:

- \( C \to A \)
- \( BC \to A \)
- \( CD \to A \)
- \( CE \to A \)
- \( BCD \to A \)
- \( CDE \to A \)
- \( BCE \to A \)
- \( BCDE \to A \)

Key features: 'points towards', 'parallel'
Hexagonal Representation
### Feasibility ???

**The number of closed sets**

<table>
<thead>
<tr>
<th>#Attrs</th>
<th>#Sets</th>
<th>#Cases</th>
<th>#Sets*</th>
<th>#Cases*</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>3</td>
<td>7</td>
<td>5</td>
</tr>
<tr>
<td>3</td>
<td>45</td>
<td>14</td>
<td>61</td>
<td>19</td>
</tr>
<tr>
<td>4</td>
<td>2 271</td>
<td>165</td>
<td>2 480</td>
<td>184</td>
</tr>
<tr>
<td>5</td>
<td>1 373 701</td>
<td>14 480</td>
<td>1 385 552</td>
<td>14 664</td>
</tr>
<tr>
<td>6</td>
<td>75 965 474 236</td>
<td>?</td>
<td>75 973 751 474</td>
<td>?</td>
</tr>
<tr>
<td>...</td>
<td>?</td>
<td>?</td>
<td>?</td>
<td>?</td>
</tr>
</tbody>
</table>

- Two sets belong to the same case (relationship/schema type) if they differ only in the order of attributes.
- * means zero-dimensional FD’s are allowed (eg. $\emptyset \rightarrow \text{Pilot}$).
Triggers: One of the Hereditary Diseases

Issues That Are Still Unsolved

Trigger defined through term rewriting systems

- confluence
- termination
- effect preservation

unfortunately only for strongly hierarchical systems (KDS/β).

Execution semantics for trigger set needed

Be careful: changes with DBMS

\[ C_1 \subseteq C_2, \ C_1 \subseteq C_3, \ C_2 \parallel C_3 \]

\[ \text{insert}_{C_1} \rightarrow^* \text{delete}_{C_3} \text{delete}_{C_2} \text{delete}_{C_1} \]

\[ \text{insert}_{C_1} \rightarrow^* \text{insert}_{C_3} \text{delete}_{C_2} \text{delete}_{C_1} \]

\[ \text{insert}_{C_1} \rightarrow^* \text{insert}_{C_2} \text{delete}_{C_3} \text{delete}_{C_1} \]

Greatest consistent specialisation (KDS/β)
Rule Triggering May Fail

Rule triggering is nice but

- termination problem
- confluence property (Church-Rosser)

and it may fail:

Consider a simplistic example: three classes $A, B, C$

integrity constraints: $A \subseteq B$, $A \subseteq C$ (inclusion constraints)

$B \parallel C$ (exclusion constraint)

rule triggering system:

$$
\begin{align*}
& i_A(x) \leadsto i_A(x); i_B(x), & i_A(x) \leadsto i_A(x); i_C(x), \\
& d_B(x) \leadsto d_B(x); d_A(x), & d_C(x) \leadsto d_C(x); d_A(x) \\
& i_B(x) \leadsto i_B(x); d_C(x), & i_C(x) \leadsto i_C(x); d_B(x)
\end{align*}
$$

general compression rules:

$$
\begin{align*}
i_T(x); d_T(x) & \leadsto d_T(x), & d_T(x); i_T(x) & \leadsto i_T(x)
\end{align*}
$$

applying this ECA set to the operation $i_A(x)$ allows to derive either

$$
\begin{align*}
i_C(x); d_B(x); d_A(x) \\
\text{or } i_B(x); d_C(x); d_A(x) \text{ or } d_B(x); d_C(x); d_A(x)
\end{align*}
$$

Therefore: be careful during design of constraints and structures!

GCS result: FAIL
Can The Designer Repair ECA Faults?

\[ R_1 = (A :: INT, C :: INT) \]
\[ R_2 = (B :: INT, D :: INT) \]

\[ I_1 \equiv R_1[A] \subseteq R_2[B] \quad \text{(inclusion dependency)} \]
\[ I_2 \equiv R_2 : D \rightarrow B \quad \text{(functional dependency)} \]
\[ I_3 \equiv R_1[C] \parallel R_2[D] \quad \text{(exclusion dependency)} \]

\[ r_1 = \text{ON insert}_{R_1}(a, c) \]
\[ \text{IF } a \notin R_2[B] \text{ THEN insert}_{R_2}(a,?) \]
\[ r_2 = \text{ON delete}_{R_2}(b, d) \]
\[ \text{IF } b \in R_1[A] \land b \notin R_2[B] \text{ THEN delete}_{R_1}(b,?) \]
\[ r_3 = \text{ON insert}_{R_2}(b, d) \]
\[ \text{IF } (b', d) \in R_2 \land b' \neq b \text{ THEN fail} \]
\[ r_4 = \text{ON insert}_{R_1}(a, c) \]
\[ \text{IF } c \in R_2[D] \text{ THEN delete}_{R_2} (?, c) \]
\[ r_5 = \text{ON insert}_{R_2}(b, d) \]
\[ \text{IF } d \in R_1[C] \text{ THEN delete}_{R_1} (?, d) \]

insert\textsubscript{R\textsubscript{1}}(a, c) \quad \text{AFTER INSERT} \quad a \notin R_1[A] \]
\[ \text{AFTER INSERT} \quad c \notin R_2[D] \]
Loving or Hating ECA?

Size of triggers: Triggers tend to become large, incomprehensible, resist maintenance and cause the *trigger crisis*.

Rule triggering is safe iff the structuring (structure *and* static integrity constraints) is strictly hierarchical (stratifiable). *This property is undecidable.*

Effect preservation of rule triggering systems is undecidable.
- Insert operation: $\text{Ins}(db, set) \sqsubseteq db$
- Delete operation: $\text{Del}(db, set) \sqsubseteq db$

Practical advice of DBMS: Whenever a problem occurs, disable rule triggering.
- *Sybase:* at most one trigger per event and per relation
- *The sad example:* LAUBAG lost their data in 1996

All execution models fail: For each execution model and a correctness condition a set of types and a set of constraints exists such that the ECA enforcement does not satisfy the correctness condition.

Therefore: either use GCS or be careful during design of constraints and structures!
Integrity Enforcement via GCS instead of ECA

Basic idea: Enhance (specialise) [basic] operations of the DBMS in such a way that the application of the operations to a consistent database will lead to a consistent database again

Restriction: Use the smallest enhancement

Greatest consistent specialisation allows to derive integrity enforcing operations for the generic database functions

reflection + predicate transformer ⇒ greatest consistent specialisation

state variable
state constraints

Extending GCS by weakening the order, by considering specific workflows instead of all, by distributed enforcement
The GCS Approach

Linguistic reflection:

reflection types \(SCHEMA_{rep},\ CLASS_{rep},\ TYPE_{rep},\ METHOD_{rep},\ COMMAND_{rep}\),

\[
\begin{align*}
\text{insert} & : \ S :: SCHEMA_{rep} \times C :: CLASS_{rep} \rightarrow METHOD_{rep} \\
\text{delete} & : \ S :: SCHEMA_{rep} \times C :: CLASS_{rep} \rightarrow METHOD_{rep} \\
\text{update} & : \ S :: SCHEMA_{rep} \times C :: CLASS_{rep} \rightarrow METHOD_{rep} .
\end{align*}
\]

one macro \text{generic} with signature \(SCHEMA_{rep} \rightarrow SCHEMA_{rep}\)

operation on state space \(X\) - characterized by two predicate transformers \(wp(S)\) and \(wlp(S)\)

assign to some postcondition \(\mathcal{R}\)

\textit{weakest (liberal) precondition} of \(S\) to establish \(\mathcal{R}\)

\(wlp(S)(\mathcal{R})\) - initial states with all terminating executions of \(S\) reach final state characterized by \(\mathcal{R}\) (provided \(S\) defined)

\(wp(S)(\mathcal{R})\) - initial states with all executions of \(S\) terminate, reach a final state characterized by \(\mathcal{R}\) (provided \(S\) defined)
GCS Approach (2)

Y-operation $S$ on $X$  \hspace{1cm} X-constraint $\mathcal{I}$

**Greatest Consistent Specialization (GCS)** of $S$ with respect to $\mathcal{I}$:

$S_{\mathcal{I}}$ specialises $S$ ($S_{\mathcal{I}} \sqsubseteq S$),

$S_{\mathcal{I}}$ is consistent with respect to $\mathcal{I}$

$S_{\mathcal{I}}$ is maximal for each $X$-operation $T$ satisfying the properties: $T \sqsubseteq S_{\mathcal{I}}$

*specialisation order* $T \sqsubseteq S$

$$wp(S)(\text{true}) \Rightarrow wp(T)(\text{true}) \quad \text{and}$$

$$wlp(S)(\mathcal{R}) \Rightarrow wlp(T)(\mathcal{R})$$

GCSs always exist

compatible with conjunctions - universal conjunctivity

subsumption freeness

usually non-deterministic

quasi-deterministic branches of a GCS

**maximal quasi-deterministic consistent specialisations** (MQCSs)

*quasi-determinism* - determinism up to the selection of some values
The Computation Of GCS

\[ R_1 = (A :: INT, C :: INT) \]
\[ R_2 = (B :: INT, D :: INT) \]

\( I_1 \equiv R_1[A] \subseteq R_2[B] \) (inclusion dependency)

\( I_2 \equiv R_2 : D \rightarrow B \) (functional dependency)

\( I_3 \equiv R_1[C] \parallel R_2[D] \) (exclusion dependency)

ON

\[ \text{insert}_{R_1}(a, c) \]

IF \( c \in R_2[D] \) THEN Fail
ELSE \( \text{insert}_{R_1}(a, c) \) IF \( a \not\in R_1[A] \)
THEN \( \text{insert}_{R_2}(a, d) \) WHERE \( d \not\in R_1[C] \cup R_2[D] \)
Identifiability, Identifier, Accessibility

Three kinds of identification statement

- **Identifiability** as the existence of a potential identification
- **Identification** as the declaration of a specific identification
- **Identifier** as the special implementation
- **Key access** as a special identifier

Diagram:

- **Language definition level**
  - **Uniqueness**
  - **Identification**
  - **Existence**

- **External level**
  - Identification

- **Conceptual level**
  - Identification (surrogate)
  - Integrity constraint (key dependency)

- **Internal level**
  - Identification accessibility
  - Invariance
Equality Concepts

Functions of an object

Equality concepts

Functions in value-oriented ... ... in value-based databases with keys
Object identifier

The devil and the beast

- Value-representability is a must.
- Detecting cycles is a nightmare.
- Cyclic structuring leads to non-first-order logics. Structures with abstract linking are potentially cyclic.
- The OID is too expressive.
- The OID has already been implemented by relational databases: tuple identifier.

Object-oriented approaches have improved and damaged culture.

Next after OO: components
Varieties of representations

Implementation alternatives

Object-oriented and object-relational approaches: objects are decomposed into a set of related objects

XML-based approaches: classes are hidden

Storage alternatives

Class-separated snowflake representation: an object is stored in several classes

storage engine

object-relational viewpoint

Full-object representation: all data associated with the object are compiled into one object

input engine, output engine

XML viewpoint
Identification is not for free

values - identified by themselves
objects - context-dependent identification
1-1 association of real world things and objects
real world things have an identification
most general: history
this approach is not applicable: ⇒ OID
objects identifiable
existence generic operations
support for consistency

Value representability

- Object identifiers do not have a meaning to the user.
- In general, a unique identification of an object is impossible if all values and references are equal.
- It is a design decision whether each object should be identifiable.

an object in \( \mathcal{D} \) is identifiable iff its orbit (for the automorphism group) is trivial
Value Identifiability

$C$ a class with representation type $T_C$

$C$ value-identifiable : $Unique(C, I_C)$

$C$ value-representable : $Unique(C, V_C)$

all objects in each instance $D_S$ are identifiable iff all classes in $S$ are weak value-representable

Value Representability

$C$ value-representable : $Unique(C, V_C)$

$\exists$ proper value type $V_C$ $\forall D$ of $S$ $\exists c : T_C \rightarrow V_C$

(1) uniqueness constraint on $C$ defined by $c$ holds

(2) $\forall$ uniqueness constraint on $C$ defined by $c' : T_C \rightarrow V'_C$ with value type $V'_C$ exists a function $c'' : V_C \rightarrow V'_C$ that is unique on $c(codom(D(C)))$ with $c' = c'' \circ c$
Values versus Objects

Values:
7, “Hugo”, (Seminar 9434, Keycode 1535)

Objects:
Cicero - as author, historical person
Tully - as friend

⇒

values - identified by themselves
objects - context-dependent identification

1-1 association of real world things and objects
real world things have an identification
most general: history
this approach is not applicable:          ⇒        OID
Values versus Objects

⇒

(1) OID invisible for users
(2) separation value/object
(3) objects have values
(4) operations (better events) are associated with objects
(5) requirements

(a) objects identifiable
(b) existence generic operations
(c) support for consistency

general object:

- identifier
- ‘set’ of values
- ‘set’ of references
- ‘set’ of methods

no definition

formal approach:

define set of objects

schema definition, instance
Distinguishability Problem

Why Constraints
Visuality
Triggering
Identity
Open
Publications

Content
Information
Concept
Topic
Distinguishability Problem

Why
Constraints
Visuality
Triggering
Identity
Open
Publications

Content
Information
Concept
Topic
Reference graphs

reference graph of $C$ in $S$

1. $\exists v_C \in V : l(v_C) = \{t, C\}$
   
   - $t$ - top-level type in the structure expression $S$ of $C$

2. for each proper occurrence of a type $t \neq ID$ in $T_C$ there exists a unique vertex $v_t \in V$ with $l(v_t) = \{t\}$

3. for each reference $r_i : C_i$ in $S$ the reference graph $G^i_{ref}$ is a subgraph of $G_{ref}$

4. for each vertex $v_t$ or $v_C$ corresponding to $t(x_1, \ldots, x_n)$ in $S$ there exist unique edges $e^{(i)}_t$ from $v_t$ or $v_C$ respectively to $v_{t_i}$ in case $x_i$ is the type $t_i$ or to $v_{C_i}$ in case $x_i$ is the reference $r_i : C_i$
   
   - in the first case $l(e^{(i)}_t) = \{S_i\}$ for the corresponding selector name $S_i$
   - in the latter case the label is $\{S_i, r_i\}$

$S = \{C_1, \ldots, C_n\}$ , $S' = \{C'_1, \ldots, C'_n\}$ another schema such that for all $i$ there exists a uniqueness constraint on $C_i$ defined by some $c_i : T_{C_i} \rightarrow T_{C'_i}$

**identification graph** $G_{id}$ of the class $C_i$ is obtained from the reference graph of $C'_i$ by changing each label $C'_j$ to $C_j$

**Algorithm**

$$F(C_i) = \begin{cases} T_i & \text{for uniqueness constraint } c_i : T_{C_i} \rightarrow T_i \\ \text{undefined} & \text{else} \end{cases}$$

iterate as long as possible:

1. If $F(C_j)$ is a proper value type and $ID$ occurs in some $F(C_i)$ corresponding to a reference to $C_j$ ($i \neq j$), then replace this $ID$ in $F(C_i)$ by $F(C_j)$.

2. If $ID$ occurs in some $F(C_i)$, then let $F(C_i)$ be recursively defined by $F(C_i) == S_i$, where $S_i$ is the result of replacing $ID_i$ in $F(C_i)$ by the type name $F(C_i)$. 

smallest labelled graph $G_{rep} = (V, E, l)$
Acyclic and cyclic schemata

- If in the reference graph $G_{ref}$ there exist uniqueness constraints for $C$ and each $C_i$ such that $C_i$ occurs as a label in $G_{ref}$, then $C$ is value-representable.
- If there exist an identification graph $G_{id}$ and uniqueness constraints for $C$ and each $C_i$ occurring as a label in $G_{id}$, then $C$ is value-identifiable.
- Value-representability is decidable for acyclic graphs.
- If there exists an acyclic identification graph the value-identifiability is decidable.
- If all explicit constraints are uniqueness constraints then value-identifiability and value-representability are decidable.
Weak value-representability

considering all references to and from a class for identification

weak value-representable classes

\( C \) - weak value-representable class in \( S \)

\[ \Rightarrow \quad \exists \text{ value type } V^w_C : Unique(C, V^w_C) \]

\( S \) : all objects in each instance \( D_S \) are identifiable iff all classes in \( S \) are weak value-representable
Open Problems: THE MFDBS List

(MFDBS1) Satisfiability of specification

(MFDBS2) Integration of static and operational constraints

(MFDBS3) Strong and soft interpretation of constraints

(MFDBS4) Global normalisation

(MFDBS5) Continuous engineering and consistent extensions

(MFDBS6) Integration of quality requirements into specification

(MFDBS7) Complexity of constraint sets only for simple classes

(MFDBS8) Enforcement of integrity constraint sets

(MFDBS9) Implication problems on the level of classes of constraints

(MFDBS10) Treatment of semantics by views

(MFDBS11) Distributed integrity management

(MFDBS12) Integration of vertical, horizontal and deductive normalisation

(MFDBS13) Treatment of incomplete specifications
Open Problems: Constraints

(TIC1) Complexity for real life constraint sets (average complexity)

(TIC2) Incomplete constraint sets and normalisation

(TIC3) Denormalisation: definition, treatment, algorithmics

(TIC4) Global normalisation: beyond classical local vertical normalisation

(TIC5) Partial axiomatisation: incomplete constraint sets

(TIC6) Graphical reasoning: for other constraint sets

(TIC7) Real-life constraint sets outside the orientation to classes
Open Problems: General Theory

(TT1) Partial identifiability

(TT2) Query optimisation with aggregation, grouping, ordering

(TT3) Inheritance: design, code, decision inheritance

(TT4) Maintenance optimisation: time/mode/strictness of integrity maintenance

(TT5) View towers: with partial local enforcement

(TT6) Component systems

(TT7) Quality management for distributed information systems
Thank you!

thalheim@is.informatik.uni-kiel.de
Publications on Database Theory

Publications on Genericity

Publications on Model Suites, Evolution, Migration

Publications on Business Process Modelling & Notation


Publications on Object Identification

Publications on Enforcement and GCS


Publications on Concepts, Content, Topics

Publications on Keys


Publications on Science and Art of Conceptual Modelling

Hungarian Publications


Editing: LNCS 305, 364, 495, 2582
Models
Towards a Theory of Models

Foundational Aspects of Conceptual Modelling, CBI 2015, Lisbon
www.is.informatik.uni-kiel.de/~thalheim/models.htm

13.7.2015

Prof. Dr. Bernhard Thalheim
Information Systems Engineering Group
Computer Science Institute
Kiel University, Germany
The starting point in 2009/2010:
There is no common understanding of a model, of to model, and of modelling.
Thus it seems that there is no notion of a model.

However:
http://www.degruyter.com/view/product/448425
http://www.is.informatik.uni-kiel.de/en/is/miscellaneous/mmm/
Many Kinds of Models

**Situation model:** reflection of a given state of affairs

**Perception model:** reflection of my current understanding of world

**Formal model:** based some formalism, within a well-based formal language

**Conceptual model:** based on formal concepts, conceptions

**Experimentation model:** guideline for experiment

**Mathematical model:** in the language of mathematics

**Computational model:** based on some (semi-)algorithm

**Physical model:** physical instrument

**Visualisation model:** for representation using some visualisation

**Representation model:** for representation of some other notion

**Diagrammatic model:** using a specific language, e.g. UML, electrotechnics

**Exploration model:** for property discovery

**Heuristic model:** based on some Fussy, probability, plausibility correlation

**Prediction model:** based on continuation assumptions, steadiness

... model:
In Which Cases is an (e)ER Schema a Model

Description scenario:

Prescription scenario:

Explanation scenario: what can be found in the DB and what not

Documentation scenario:

Communication scenario: understanding the database

Negotiation scenario:

Inspiration scenario:

Exploration scenario:

Definition scenario:

Prognosis scenario:

Reporting scenario: schema as a result of design
Program and Problems to be Tackled for a General Theory of Conceptual Models

(1) Is there any general theory of conceptual models?
   Far too many application areas; many approaches, viewpoints!!

(2) Restricting consideration to language-based models
   Which universality we need?

(3) Models as artifact, image and prescription
   Are we able to handle this variety?

(4) Purpose orientation of models
   Can we derive model properties from the purpose?

(5) Semiotics as a language background
   Which kinds of associations we might derive?

(6) Principles and theory
   Is there any theory of models?

(7) Modelling acts and modelling workflows
   Are we able to manage modelling at a SPICE level 3?
Towards a General Modelling Theory

What is a Model? adequate and dependable artifact, functional, effective

Generalising the variety of model notions

Language matters e.g., enables or disables

Managing complexity of models and methods

Model development methods

Model utilisation methods

Model profile and portfolio

Model usage stories

The model as embodied perception and understanding.
Properties of a Model

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

(1') **Analogy** property: the model is analogous to the origins based on some analogy criterion.

(2) **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.

(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.

(9') **Function** property: models and conceptual models are governed by the utilisation scenario. The model suffices in its function.
Conception Frame of the Notion of the Model (Preliminary)

- **“what, whereof”: artifacts** for be represented by the instrument
- **“who”: community** of practice
  - interest, portfolio, profile, roles, specific plays, rights, obligations, current practice
  - functions in the scenario
  - utilisation pattern
  - implied purpose and goal
  - utilisation methods
- **“in what way, how”: usage spectrum and scenarios**
  - utilisation scenario
  - capacity, potential
  - utilisation pattern
- **“wherefore”: adequate**
  - analogous
  - focused
  - purposeful
  - empirical corroboration
  - coherent conform
  - validatable falsifiable
  - stability plasticity
  - quality
  - internal
  - external
  - in use
  - assurance evaluation
  - basement, paradigms, postulates, restrictions,
  - theories, culture, foundations, conventions, commonsense
  - concepts, foundations, language as carrier,
  - assumptions, thought community, practice
- **“why, worthiness”: dependable**
  - basis
  - application domain or discipline, disciplinary thought
  - assumptions, thought community, practice
  - time, space, granularity, scope
- **“by what means”: background**
  - grounding
  - application domain or discipline, disciplinary thought
  - assumptions, thought community, practice
Explicit Treatment of Purposes

explanation, blueprint, reconstruction, illustration

system construction

communication, fault prevention

easy explanation
The Usefulness, Usability and Usage of the Model

A model is an instrument that is adequate and dependable that is going to be used depending on its purpose.

Therefore, we need to enhance the model to become used.

**Functional models** are models for which methods $M_D$ exists for utilisation in dependence on the end/purpose/function $D$. (The capability of a model.)

**Effective models** are functional models that thoroughly function in the application scenario (‘usage games’). (The model as a tool and its value.)

**Conceptual models** are models that incorporate concepts or conceptions.

The model is an INSTRUMENT!!!

---

**Information**

- **Content**
- **Concept**
- **Topic**
The Model is an Instrument

based on utilisation scenarios and use spectra

**Judgement** 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.

**Instrument as such** with some appreciation (Wertschätzung, Auszeichnung)  
**well-shaped instrument** on the basis  
- of some criteria in dependence on intended utilisation and criteria for  
- that it is accepted in a CoP, and  
- that it is syntactically, semantically, pragmatically well-shaped,  
**fits** to the intended use, and  
**is appropriate** for the use spectra.
An Instrument that is Used as a Model

Given artefacts $\mathcal{A}$ (origins).

Any (real or virtual) instrument can be used as a model if it is adequate for $\mathcal{A}$, i.e., well-defined, $\mathcal{A}$-analogous, $\mathcal{A}$-reduced (or $\mathcal{A}$-focused), and $\mathcal{D}$-purposeful,

and

dependable (or workmanlike, professional):

justified (corroborated, rational coherent and conform, falsifiable, stable and plastic) and

sufficient (of firm quality and evaluated).

The instrument has its 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).
Model Adequacy

**Analogous** to the origins (artefacts) to be represented according to some analogy criterion;

**More focused** (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled

Sufficient to satisfy its **purpose**.

plausible reasoning on the basis of abduction, induction analog / autoepistemic / default / defeasible / non-monotonic ...
reasoning, also reasoning by (counter-)examples

Assumption: An instrument is **well-formed**, i.e. it satisfies a well-formedness criterion.
Dependability: Justification

(Empirical) explanatory statement: based on the profile and background, corroboration, viability typically given by an argument calculus (e.g. observations, reflections)

Rational coherence: inner model coherence; reflects norms and standard accepted by CoP (e.g. controlled redundancy); reflects common practices

Validatable (falsifiable): against the artifacts it represents (e.g. things or other models) based on validation models abductive calculus or model checking

Stable and plastic: represents a family of artifacts; not overfitted to a small set of artifacts (or singleton) - no need to constant adaptation; general to certain extent; based on analogy; used as a masterpiece (‘beau ideal’); may be reflected by another model (e.g. visualisation model)
Dependability: Sufficiency based on Quality Safeguard

**Quality in use:** performance while using, parsimony, understandability, aptitude or eligibility, attractiveness, usability, effectivity, integratability, functionality, stability, added values in utilisation scenario, freeness or restrictedness from side conditions, ...

**External quality:** extensibility, adaptability, replacebility, efficiency, stability, validatable, testable, analysable, exactness, error-proneness, spacial and temporal behaviour, dependability, maintainable, safe, ...

**Internal quality:** value of the instrument, e.g. inner coherence, toleratable redundancy, simplicity, evolution-proneness, constructivity, modularisation, coherence, efficiency, ...

**Evaluation procedure:** typically using a number of specific evaluation methods
tolerated discrimination or dissimilarity or non-adequacy; modality (necessity, contingency or possibility, relativity) within the context; confidence of the evaluation
The Background of an Instrument to be used as a Model

Each artefact has its background consisting of the

(1) **basis** $B$ (basement, paradigms, postulates, restrictions, theories, culture, foundations, conventions, commonsense);

(2) **grounding** $G$ (concepts, foundations, language as carrier, assumptions, thought community, practice);

filtered against utilisation scenario and in dependence on

((i) **context** $C$) (application domain or discipline, school of thought, time, space, granularity, scope);

((ii) **community of practice** $P$) (interest, portfolio, profile, roles, specific plays, rights, obligations, current practice).
The Directives for an Instrument to become a Model

**Context** $C$, e.g. application domain or discipline, school of thought, time, space, granularity, scope, infrastructure, ... .

**Community of practice** $P$ (author, developer, implementer, documenter, user, ... (with their interest, portfolios, profiles, roles, specific plays, rights, obligations, current practice) governs the viewpoints, orientation and background of users involved

**Profile** goal, purpose, function of the instrument (current versus intended state, methods, usage scenarios envisioned)

**Origins/artefacts** to be represented by the model e.g. artefacts, other models, with purpose invariance
The Ladenness in Modelling

"Interest-ladeness" based on viewpoints on origins and usage spectra

Part of reality

Elements of interest

Origins of interest in the reality

Utilisation scenario, usage spectra

Function, goal and purpose bundle

"Background-ladeness"

Intensional elements of background

Usage of background

Confidence

Modality

Exactness

"Development process ladenness"

Modell as combined result and partial point of view in a modelling process

"Pragmatism-ladenness"

CoP, e.g. applicator, modeler, ...

Focused activity

Previous modelling experience

accumulating experience

Content

Information

Concept

Topic
The Kiel Notion of the MODEL

Background of an instrument used as a model: (1) basis $B$ (basement, paradigms, postulates, restrictions, theories, culture, foundations, conventions, commonsense); (2) grounding $G$ (concepts, foundations, language as carrier, assumptions, thought community, practice); (3) context $C$ (application domain or discipline, school of thought, time, space, granularity, scope); (4) community of practice $P$ (interest, portfolio, profile, roles, specific plays, rights, obligations, current practice).

A MODEL is an instrument that satisfies for artefacts $A$ it represents:

**Adequate** for $A$, i.e., well-defined, $A$-analogous, $A$-reduced (or $A$-focused), and $D$-purposeful.

**Dependable:** (or workmanlike, professional), i.e., justified (corroborated, rational coherent and conform, falsifiable, stable and plastic) and sufficient (of firm quality and evaluated).

**Functional models** are models for which methods $M_D$ exists for utilisation in dependence on the end/purpose/function $D$.

**Effective models** are functional models that thoroughly function in the application scenario ('usage games').

**Conceptual models** are models that incorporate concepts or conceptions.
Models
Theory
13.7.2015
B. Thalheim

Overview
Instruments
Adequate
Dependable
Background
The MODEL
Agenda

Portfolio: Utilisation scenarios and use spectra

Model as an instrument

Context
- time/space/
discipline/school/
application

Develop, construct,
configure, orchestrate,
choreography

Artefacts represented
by model

Community of practice
- role, right,
obligation, play

Usage
- business cases
installation, consolidate

Use for solving/
plan/construct/
skilled application

Prognose/simulate/
estimate/
forecast/appreciate/
treasure

Goal/ purpose/
function, profile

Goal/
purpose/
function

Artefacts represented
by model

Development methods
- Describe/characterise/
picture/deport/represent/
relate/specify
- Create/produce/
form/shape/design/
arrange/
organise/model
- Evolve/migrate/
redevelop
- Corroboration/viability/
justifly/support/
found/establish/
prove/argue/
substantiate/motivate/
constitute/show/why
- Evaluate/quality/
argument/assess/
internal quality/
external quality/
quality of use
modality/confidence

Content
- Paradigms,
culture, background,
foundations, theories,

Concepts, languages, routine, training, infrastructure, assumptions,
though community, thought style, pattern, methodology, guidelines, practice

Basis
- postulates, restrictions,
authorities, conventions, commonsense

Grounding

Information
- Concept

Topic
Modelling: Towards Art

...the art of programming (D.E. Knuth).

The expression or application of human creative skill and imagination, often in a visual form, producing instruments to be appreciated primarily for their easy and simple way of usage in some scenarios.

The conscious use of the understanding in the production of well-formed instruments intended to be contemplated or appreciated as adequate and dependable.

Marriam-Webster:

1: skill acquired by experience, study, or observation

2a: a branch of learning:

2a(1): one of the humanities

4b(2): plural: liberal arts

2b: archaic: learning, scholarship

3: an occupation requiring knowledge or skill

4a: the conscious use of skill and creative imagination especially in the production of aesthetic objects; also: works so produced

4b(1): fine arts

4b(2): one of the fine arts

4b(3): a graphic art
Modelling: Towards Science

is any system of knowledge that is concerned with the physical world and its phenomena and that entails unbiased observations and systematic experimentation. In general, a science involves a pursuit of knowledge covering general truths or the operations of fundamental laws. *Encyclopedia Britannica*

Science is treated in a number of articles. For the history of Western and Eastern science, see science, history of. For the conceptualization of science and its interrelationships with culture, see science, philosophy of. For the basic aspects of the scientific approach, see physical science, principles of. For the historical development of the different sciences and their scope, component disciplines, methods, and principal problems, see physical science; Earth sciences; biology; medicine, history of; engineering; social science.

1. the state of knowing: knowledge as distinguished from ignorance or misunderstanding
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: natural science
4. a system or method reconciling practical ends with scientific laws
5. ...
Modelling: Towards Culture

Real goal: culture of models, model activities, and modelling!

A collective phenomenon, which is shared with people who live or lived within the same social environment, which is where it was learned;
Culture consists of the unwritten rules of the social game;
It is the collective programming of the mind that separates the member of one group or category of people from others. (Hofstede)

A coherent whole of principles, methods, and models that are used in the design and realisation of an enterprise organisational structure, business processes, information systems, and infrastructure. (Lankhorst)

TOGAF (The Open Group): 1. A formal description of a system, or a detailed plan of the system at component level to guide its implementation.
2. The structure of components, their inter-relationships, and the principles and guidelines governing the design and evolution over time.
Thank you!

thalheim@is.informatik.uni-kiel.de
Revisiting the Definition of the Relational Tuple Calculus

ADGIS’2015
Futoroscope, France
Sept. 9, 2015

Bader AlBdaiwi and Bernhard Thalheim

Computer Science Department
Kuwait University, Kuwait

Dep. of Computer Science
Kiel University, Germany
Overview

- Database theory has been developed at some technology stage but did not really evolve to the current stage.
- Old database problems are still open and will not be solved due to some kind of new technologies.
- Many assumptions have been valid in the past but are not valid anymore.
- Many theories have been developed but need a revision nowadays.
- Technologies and theories have been partially based in the past on really wrong approaches but were so successful that nobody touches them.

Our concerns:

1. Tuple calculus is an example of the last point.
2. We can repair it if we use logics in a better way.
3. We do not loose anything but avoid such pitfalls and flaws.
Tuple Calculus Defined by Brute-Force Application of First-Order Predicate Logics

relational data scheme \( \mathcal{D} = (U, D, \text{dom}) \)
tuple (or object) as a function \( t : X \rightarrow D_{DSD} \) with \( t(A) \in \text{dom}(A) \) for \( A \in X \).
relation type \( R : (\text{attr}(R), \text{id}(R), \Sigma(R)) \)
relation \( R^C \) over a relation type \( R \)
relational database schema \( S \) over the relational data scheme \( \mathcal{D} \)
relational database \( D_S^C \) over \( S \) and \( \mathcal{D} \)

formulas of the relational tuple calculus:
(1) atomic formulas: \( P_R(x_1, \ldots x_m), \text{true}, \text{false}, x_i \theta x_j \) and \( x_i \theta d \);
(2) formulas for formulas \( \alpha, \beta \):
\( \neg \alpha, \alpha \land \beta, \alpha \lor \beta, \alpha \Rightarrow \beta \)
\( \forall x(P_R) \alpha \) and \( \exists x(P_R) \alpha \) for free variables \( x \) in \( \alpha \);
(3) nothing else is a formula.

tuple relational expression:
\( \{(x_1, \ldots, x_p) \mid \alpha \} \) for \( \alpha \) with free variables \( x_1, \ldots, x_p \), i.e. \( \alpha(x_1, \ldots, x_p) \)
Research Issues for the Tuple Calculus

Finiteness of answers for finite databases that are defined on potentially infinite relational data schemes (i.e. with potential infinity of domains) guaranteed for the relational algebra and DBMS infinite views as an input to tuple relational queries infinite data warehouses, ...

Multi-set semantics of SQL queries instead of the set semantics provided by the the First-order PL

Optimisation for tuple query expression sets

User-friendly specification pattern especially for complex questions with easy-to-go transformations to query features of modern DBMS
Finiteness it only one of the Open Research Problems

**Codesign** of information systems based on co-evolving structures and functions

**Structuring of modern information systems** with appropriate languages, for large schemata, for object-relational databases, for big data, for analysis data, based on agile and sustaining approaches, using pattern, with controlled redundancy

**Functionality of information systems** within a holistic setting, for higher SQL etc. languages, within IT infrastructures, supporting needs of users such as privacy

**View management** for a large variety of uses beyond the query expansion features, integrating also data massives and marts

**Workflow management** for flexible and adaptable workflows, with robust behaviour

**Storyboarding** for information systems according to the needs, demands and the profile of users within different cultural setting

**Database theory** for advanced systems, for complex constraints, advanced logics, with incomplete constraint knowledge, reasoning with constraints

**Model theory and science**

see BIR’2013 (LNBIB 158, 2013, Springer) for a detailed list
Nightmares with the Tuple Calculus

small database schema with
attr(Movie) = \{ title, year \}, attr(Performance) = \{ cinema, movie, date, time \}
for dom(cinema) = STRING, dom(time) = TIME and dom(date) = DATE

‘Which movies are performed in UCI?’
\{(m) | \exists d \ \exists t \ \exists y \ Performance(UCI, m, d, t) \land Movie(m, y)\}.

‘In which cinemas has not been played the movie Casablanca?’
\{(c) | \forall d \forall t \neg (Performance(c, Casablanca, d, t))\}

\{(c, d) | \exists t \ Performance(c, Albatros, June 17, 2015, t)
\lor Performance(UCI, Casablanca, d, t)\}

Casablanca on July 1, 2015 (any time), then (c, July1, 2015)
Albatros in Cinemax cinema (any time)
then also all pairs (Albatros, d) with arbitrary d

\{(c) | \forall t : Performance(c, Rambo, d, t)\}
results in \emptyset if dom(date) is infinite
Half-Way Proposals

Safe formulas: domain independent and well-defined usages of variables

The problem in determining whether a tuple relational expression is domain independent is undecidable.

Whether a given domain independent formula is equivalent to a safe formula is undecidable.

Relationally restricted formulas: $\forall x (R)(\alpha \implies \beta)$ is relationally restricted if $x$ is the only variable in $\alpha$ and $x$ occurs positive in $\beta$ where both $\alpha$ and $\beta$ are quantifier free formulas.

same problem

Tuple relational calculus with domain restrictions:

$\{ T \mid L \mid \alpha \}$

The tuple relational calculus with domain restrictions has not the same expressive power as the relational algebra.
HERM Query Algebra

Operations are defined on the basis of structural recursion \( op = src[e,g,\sqcup] \)
with a value \( e \) of type \( t' \), a function \( g : t \rightarrow t' \) and a function \( \sqcup : t' \times t' \rightarrow t' \):

\[
\begin{align*}
src[e,g,\sqcup](\emptyset) &= e \\
src[e,g,\sqcup](\{x\}) &= g(x) \quad \text{for } x \text{ of type } t, \\
src[e,g,\sqcup](X \cup Y) &= src[e,g,\sqcup](X) \cup src[e,g,\sqcup](Y) \quad \text{for } X, Y \text{ of type } \{t\}, \\
\end{align*}
\]
e.g., \( \text{filter}(\phi) = src[\emptyset, \text{if} \_ \text{then} \_ \text{else} \ast (\phi \times \text{single} \times (\text{empty} \ast \text{triv})), \sqcup] \),
\( \text{sum} = src[0, \text{id}, +] \)

Operations for tuple types are defined by structural recursion on basic operations
projection \( \pi_i : t_1 \times \ldots \times t_n \rightarrow t_i \), (Cartesian) product \( \times : t \rightarrow t_1 \times \ldots \times t_n \),
reordering \( \rho \), renaming \( \kappa \)

Operations for set types are defined by structural recursion on basic operations union \( \sqcup \),
difference \( \setminus \), constant, singleton element,
e.g., join operation

Operations on function types are defined by structural recursion on basic operations composition \( \ast : (t_2 \rightarrow t_3) \times (t_1 \rightarrow t_2) \rightarrow (t_1 \rightarrow t_3) \), evaluation \( \text{ev} : (t_1 \rightarrow t_2) \times t_1 \rightarrow t_2 \) and abstraction \( \text{abstr} : (t_1 \times t_2 \rightarrow t_3) \rightarrow (t_1 \rightarrow (t_2 \rightarrow t_3)) \)

Conversion operations from tuple to set types, from function types to collections, etc.

Operations for URL extension using labels \( l \) extending the type system to
\[
t = b \mid l \mid t_1 \times \ldots \times t_n \mid \{t\} \mid [t] \mid l : t
\]
that is restricted to rational trees (number of different subtrees is finite)
HERM-Algebra via Structural Recursion

- **Selection**: \( \sigma_\alpha = \text{srec}_\emptyset, \iota_\alpha, \cup \)

  \[ \iota_\alpha(\{o\}) = \begin{cases} \{o\} & \text{if } \{o\} \models \alpha \\ \emptyset & \text{otherwise} \end{cases} \]

- **Projection**: \( \pi_X = T[X] = \text{srec}_\emptyset, \pi_X, \cup \), \( X \subseteq T \)

  \[ \pi_X(\{o\}) = \{o|_X\} \]

- **(Natural) Join**: \( \Join = \text{srec}_\emptyset, \Join_T, \cup \)

  \[ \Join_T(\{(o_1, o_2)\}) = \{o \in \text{Dom}(T_1 \cup T_2) | o|_{T_1} = o_1 \land o|_{T_2} = o_2\} \]

  Cartesian product and intersection as special cases

- **Renaming**: \( \rho_{X,Y} = \text{srec}_\emptyset, \rho_{X,Y}, \cup \)

  \[ \rho_{X,Y}(\{(o)\}) = \{o' \in \text{Dom}((T \setminus X) \circ Y) | o|_{T \setminus X} = o'|_{T \setminus X} \land o|_X = o'|_Y\} \]

- **Nesting**: \( \nu_X = \text{srec}_\emptyset, \nu_{X,\{X\}}, h_2 \) for \( X \subseteq T = R \)

  \[ T' = C(R \setminus X) \cup_R \{X\}, \quad o'|_X \in \pi_X(T'^C), \]

  \[ h_2(\{o'\}, T'^C) = \{o' \cup T'^C \} \text{ if } o'|_X \not\in \pi_X(T'^C) \]

  \[ h_2(\{o'\}, T'^C) = \{o \in \text{Dom}(T') | \exists o' \in T'^C : o|_{R \setminus X} = o'|_{R \setminus X} \land o(X) = \{o''[X] | o'' \in T'^C \land o'|_{R \setminus X} = o''|_{R \setminus X}\}\} \]

- **Unnesting**: \( \mu_X = \text{srec}_\emptyset, \mu_{X,\{X\}}, h_2 \), \( \{X\} \subseteq T = R \)

  \[ T'' = C(R \setminus \{X\}) \circ X, \quad h_2(\{o'\}, T'^C) = \{o' \cup \}

  \{o \in \text{Dom}(T') | \exists o'' \in R^C : o[R \setminus \{X\}] = o''[R \setminus \{X\}] \land o|_X \in o''|_X\} \]
Algebraic Tuple Calculus

\[ \{ \text{target structure} \mid \text{context} \mid \text{conditions} \} \]

identical copy expression: \( \{t \mid T \mid T^C\} \)

selection expression: \( \begin{array}{c}
\{ t \mid T \mid \beta \}
\{ t \mid T \mid \beta \land \alpha \}
\end{array} \)

join expression: \( \begin{array}{c}
\{ t_1 \mid T_1 \mid \alpha_1 \}, \{ t_2 \mid T_2 \mid \alpha_2 \}
\{ t_1 \land T_1 \cup T_2 \mid T_1 \cup T_2 \mid \alpha_1 \land \alpha_2 \}
\end{array} \)

\( \land T_1 \cup T_2 = \text{src} \left[ \emptyset \cup T_1 \cup T_2, \land T_1 \cup T_2, \cup T_1 \cup T_2 \right] \)

union expression: \( \begin{array}{c}
\{ t \mid T \mid \alpha_1 \}, \{ t \mid T \mid \alpha_2 \}
\{ t \mid T \mid \alpha_1 \lor \alpha_2 \}
\end{array} \)

difference expression: \( \begin{array}{c}
\{ t \mid T \mid \alpha_1 \}, \{ t \mid T \mid \alpha_2 \}
\{ t \mid T \mid \alpha_1 \land \lnot \alpha_2 \}
\end{array} \)

renaming expression: \( \begin{array}{c}
\{ t \mid T \mid \alpha \}
\{ \rho(t) \mid \rho(T) \mid \rho(\alpha) \}
\end{array} \)
Equivalence to the Query Algebra

The algebraic tuple calculus is correct, i.e. it does not result in undefinable expressions. Any expression is evaluated in a finite relational database to a finite set of objects within the context of the expression.

The algebraic operations selection, join, union, difference, renaming, and projections are directly expressible within the algebraic tuple calculus.

Expressions in the algebraic tuple calculus can directly be expressed through algebraic operations selection, join, union, difference, renaming, and projections.

The algebraic tuple calculus has the same expressive power as the relational algebra.

The following three query languages for the relational database model are equivalent: The relational algebra; the algebraic tuple calculus; the safe formula calculus.
The General Tuple Calculus

including also aggregation

• For ordered sets \( X \subset U, Y \subset U \cup U^* \), a \emph{recursion frame} \((e, g, h)\) on \( X, Y \), a relation type \( R \) from \( S \), and a relation \( R^C \).

  The expression
  \[
  \{ t \mid Y \mid \text{src}[e, g, h](R^C) \}
  \]

  is called (atomic) general tuple expression on \( Y \) over \( S \).

• Given an ordered set \( Y \subset U \cup U^* \), a \emph{recursion frame} \((e, g, h)\) on \( X, Y \), and a general tuple expression over \( S \)

  \[
  X^C = \{ t \mid X \mid \alpha \}
  \]

  Then the expression
  \[
  \{ t \mid Y \mid \text{src}[e, g, h](X^C) \}
  \]

  is a general tuple expression.

• Nothing else is a general tuple expression on \( Y \) over \( S \).

  \[
  \begin{align*}
  \{ t \mid X \mid \alpha \} \\
  \{ t \mid Y \mid \beta \}
  \end{align*}
  \]

  for \((e, g, h)\) and \( X, Y \subset U \cup U^* \).

Given a relational database schema \( S \) and a finite database \( \mathcal{D}_S^C \) over a schema \( S \). Any general tuple expression over \( S \) yields a finite set for \( \mathcal{D}_S^C \).
Finally

Lessons

• The need of **purpose-oriented languages**.
• The **triple definition frame** for well-formed expressions.
• Simple **extension** to eER or object-relational or XML languages.
• Treatment of **bag expressions** for SQL and NoSQL.
• Maintaining **optimisation features** of the algebra.
• Supporting **view extensions and generic operations**.
• **Don’t use languages in a convenience-oriented manner** if they are not providing the right features in the form that you need.
Thank you!

thalheim@is.informatik.uni-kiel.de
A Conceptual Model for Services
Towards a Theory of Models

Models
Towards a Theory of Models
CMS 2015 @ ER 2015
www.is.informatik.uni-kiel.de/~thalheim/models.htm

21.10.2015

Bernhard Thalheim ¹, (Ajantha Dahanayake ²)

¹ Christian-Albrechts-University Kiel, Germany & Lomonosov University Moscow, Russia
² Prince Sultan University, Dept. of CIS, Riyadh, Kingdom of Saudi Arabia
thalheim@is.informatik.uni-kiel.de  adahanayake@pscw.psu.edu.sa
My Background

ER - main guide

Handbook

Dependencies

Modelling

ADBIS

ASM

CM

EJC

e-Bus.

ER

FoIKS

MFDBS

NLDB

Semantics

WIS

WISE

i.e., reaching the big "3"s (3, 30, 300)
(3 (habilitation students, profs, books), 30 (PhD students, editorials, projects),
300 (master/diploma students, papers, presentations))
The starting point in 2009/2010: There is no common understanding of a model, of to model, and of modelling. Thus it seems that there is no notion of a model. However: http://www.degruyter.com/view/product/448425
Overview

Visions, results, projects

(1) Necessities of real service applications
    provide a model that covers all aspects

(2) Application domain and requirements elicitation
    integrate the deployment and user perspective

(3) IT systems provide services
    Mathematics, engineering, application domains

(4) Service modelling
    including application domain

(5) Mapping services to Information systems
    providing a validatable and verifiable system specification
Program and Problems to be Tackled for a General Theory of Service Models

(1) Is there any general theory of service models?
   Far too many application areas; many approaches, viewpoints !!

(2) Restricting consideration to language-based models
   Which universality we need?

(3) Models as instrument, image and prescription
   Are we able to handle this variety?

(4) Purpose orientation of models
   Can we derive model properties from the purpose?

(5) Semiotics as a language background
   Which kinds of associations we might derive?

(6) Principles and theory
   Is there any theory of models?

(7) Modelling acts and modelling workflows
   Are we able to manage modelling at a SPICE level 3?
Towards a General Modelling Theory and a Conceptual Model for Services

What is a Model? adequate and dependable instrument, functional, effective

Generalising the variety of model notions

Language matters e.g., enables or disables

Managing complexity of models and methods

Model development methods

Model utilisation methods

Model profile and portfolio

Model usage stories

The model as embodied perception and understanding.
Application Example: Ophthalmology

**Application:** develop a system that supports similarity matching between patient’s eye image and images available in a repository

**Main Issue:** representation of an ophthalmologist’s way of information processing during matching images to identify the disease

**Research issue:** integration of computational, cognitive, and socio-technical aspects
Application Example: Intensive Breath Care by Evita 4

IPPV (Intermittent Positive Pressure Ventilation);
SIMV (Synchronized Intermittent Mandatory Ventilation);
MMV (Mandatory Minute Volume Ventilation);
SB (Spontaneous Breathing);
CPAP (Continuous Positive Airway Pressure);
ASB (Assisted Spontaneous Breathing);
BIPAP (Biphasic Positive Airway Pressure);
BIPAPAssist (Biphasic Positive Airway Pressure Assisted);
APRV (Airway Pressure Release Ventilation);
PPS – Proportional Pressure Support;
ILV (Independent Lung Ventilation);
Automatic Tube Compensation ATC;
Apnoea Ventilation;
NIV mask ventilation; ... ... ...
Application Systems are User-Oriented: Services Provided by a System that Satisfy the Information Demand

**Services** deliver and accept the data based on content to the user

**Information** as requested by the user

**Features** of the information system for the support of the user
Application Systems are User-Oriented: Interaction Objects for Information-Intensive Systems

Containers deliver and accept the data based on content to the user.
Container are media types, i.e., provide data with functions for their utilisation.

Information as requested by the user through media objects.

Media objects for holistic information logistics.
Cross-Disciplinary Approach to Development

(1) Task- and usage-driven development of supporting means for real-life applications

   computer engineers neglect user requirements

(2) Systems must follow rules, culture, understanding of users, e.g., privacy, security, dependability

   computer engineers are not educated to take this into account

(3) Systems become far more complex, handle huge data, must be integrated, used with skills

   holistic understanding of applications with flexible integration

(4) Strengths, weaknesses, opportunities, and threat of systems must be understood by users before used with full benefit

   users must not be adapted to the systems but should continue to live

(5) Systems are going to be used by many different people with many different background

   cultures matter, cross-disciplinary, cross-cultural
Requirements impacted from the Application Domain to Social-Technical Systems
Conceptual Model for IT Service System

- Fundamental elements for developing applications;
- Organising 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 synchronise 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.
Many Kinds of Models

**Situation model:** reflection of a given state of affairs

**Perception model:** reflection of my current understanding of world

**Formal model:** based some formalism, within a well-based formal language

**Conceptual model:** based on formal concepts, conceptions

**Experimentation model:** guideline for experiment

**Mathematical model:** in the language of mathematics

**Computational model:** based on some (semi-)algorithm

**Physical model:** physical instrument

**Visualisation model:** for representation using some visualisation

**Representation model:** for representation of some other notion

**Diagrammatic model:** using a specific language, e.g. UML, electrotechnics

**Exploration model:** for property discovery

**Heuristic model:** based on some Fussy, probability, plausibility correlation

**Prediction model:** based on continuation assumptions, steadiness

... model:
Properties of a Model

(1) **Mapping** property: the model has an origin and can be based on a mapping from the origin to the instrument.

(1') **Analogy** property: the model is analogous to the origins based on some analogy criterion.

(2) **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.

(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.

(9') **Function** property: models and conceptual models are governed by the utilisation scenario. The model suffices in its function.
Explicit Treatment of Purposes

- explanation, blueprint, reconstruction, illustration
- system construction
- communication, fault prevention
- easy explanation

Service Model
21.10.2015
© β/A.Da.
From Goal and Purpose to Function

**Goal:** notion of situation and augmented, envisioned, striven situation considered by a community of practice within a task setting, within a context (time, space, discipline, infrastructure, ...) within the basis and grounding

**Purpose:** notion of the *artefact* for the goal that can be used due to existence of methods that has utility due to its capacity and potential

**Function:** notion of the *instrument* for the purpose that can and is to be used in utilisation scenarios or usage spectra with an added value

(The model emphasises what is important to the CoP.)
The Usefulness, Usability and Usage of the Model

A model is an instrument that is adequate and dependable that is going to be used depending on its purpose.

Therefore, we need to enhance the model to become used.

Functional models are models for which methods $M_D$ exists for utilisation in dependence on the end/purpose/function $D$. (The capability of a model.)

Effective models are functional models that thoroughly function in the application scenario (‘usage games’). (The model as a tool and its value.)

Conceptual models are models that incorporate concepts or conceptions.

The model is an INSTRUMENT!!!
The Anti-Profile of an Instrument

- Useless for what scenario?
- Efficiency and effectiveness is not given for: description & prescription, realisation & coding, theory development, theory refinement, causality consideration, inexplicability, demonstration, prediction, explanation, master complexity, understanding.
- Essential parameters are missing; instead mediating or dependable parameters are given; dummy or pseudo dependance.
- What cannot be adequately represented?
- In which case users need a special understanding?
- In which cases the instrument cannot be effectively used?
- Which biases, which background is palmed off?
- Assumptions, postulates, paradigms, schools of thought that are severe.
- Models might condition conclusions.
The Model is an Instrument

functioning in utilisation scenarios and use spectra

Judgement 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.

Instrument as such with some appreciation (Wertschätzung, Auszeichnung) well-shaped instrument on the basis

- of some criteria in dependence on intended utilisation and criteria for that it is accepted in a CoP, and
- that it is syntactically, semantically, pragmatically well-shaped,

fits to the intended use, and

is appropriate for the use spectra.
An Instrument that is Used as a Model

Given origins Α (e.g. artefacts).

Any (real or virtual) instrument can be used as a model if it is **adequate** for Α, i.e., well-defined, Α-analogous, Α-reduced (or Α-focused), and Ω-purposeful,

and

**dependable** (or **workmanlike**, **professional**):
justified (corroborated, rational coherent and conform, falsifiable, stable and plastic) and
sufficient (of firm quality and evaluated).

The instrument has its **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).
Reasoning with Models

Models as mediators
Models for inspiration
Models for description
Models for prescription
Models for realisation & coding
Models for theory development and theory refinement
Models for causality consideration
Models for demonstration
Models for prediction
Models for explanation and understanding
Models for mastering complexity
Model Adequacy

**Analogous** to the origins (e.g. artefacts) to be represented according to some analogy criterion;

**More focused** (e.g. simpler, truncated, more abstract or reduced) than the origins being modelled

Sufficient to satisfy its **purpose**.

plausible reasoning on the basis of abduction, induction
analog / autoepistemic / default / defeasible /non-monotonic ...
reasoning, also reasoning by (counter-)examples

Assumption: An instrument is **well-formed**, i.e. it satisfies a well-formedness criterion.
Dependability: Justification

(Empirical) explanatory statement: based on the profile and background, corroboration, viability

typically given by an argument calculus (e.g. observations, reflections)

Rational coherence: inner model coherence; reflects norms and standard accepted by CoP (e.g. controlled redundancy); reflects common practices

Validatable (falsifiable): against the origins it represents (e.g. things or other models) based on validation models abductive calculus or model checking

Stable and plastic: represents a family of origins; not overfitted to a small set of origins (or singleton) - no need to constant adaptation; general to certain extent; based on analogy; used as a masterpiece (‘beau ideal’); may be reflected by another model (e.g. visualisation model)
Dependability: Sufficiency based on Quality Safeguard

Quality in use: performance while using, parsimony, understandability, aptitude or eligibility, attractiveness, usability, effectivity, integratability, functionality, stability, added values in utilisation scenario, freeness or restrictedness from side conditions, ...

External quality: extensibility, adaptability, replacebility, efficiency, stability, validatable, testable, analysable, exactness, error-proneness, spacial and temporal behaviour, dependability, maintainable, safe, ...

Internal quality: value of the instrument, e.g. inner coherence, toleratable redundancy, simplicity, evolution-proneness, constructivity, modularisation, coherence, efficiency, ...

Evaluation procedure: typically using a number of specific evaluation methods
tolerated discrimination or dissimilarity or non-adequacy; modality (necessity, contingency or possibility, relativity) within the context; confidence of the evaluation
The Background of an Instrument to be used as a Model

Each artefact has its background consisting of the

(1) **basis** $B$ (basement, paradigms, postulates, restrictions, theories, culture, foundations, conventions, commonsense);

(2) **grounding** $G$ (concepts, foundations, language as carrier, assumptions, thought community, practice);

filtered against utilisation scenario and in dependence on

((i) **context** $C$) (application domain or discipline, school of thought, time, space, granularity, scope);

((ii) **community of practice** $P$) (interest, portfolio, profile, roles, specific plays, rights, obligations, current practice).
The Directives for an Instrument to become a Model

Context $\mathcal{C}$, e.g. application domain or discipline, school of thought, time, space, granularity, scope, infrastructure, ...

Community of practice $\mathcal{P}$ (author, developer, implementer, documenter, user, ... (with their interest, portfolios, profiles, roles, specific plays, rights, obligations, current practice)
governs the viewpoints, orientation and background of users involved

Profile goal, purpose, function of the instrument (current versus intended state, methods, usage scenarios envisioned)

Origins to be represented by the model
  e.g. artefacts, other models, with purpose invariance
The Ladenness in Modelling

“Interest-ladenness”
based on viewpoints on origins and usage spectra

Origins of interest in the reality

Function, goal and purpose bundle

Part of reality

Elements of interest

Usage of background

Model element

Modell as combined result and partial point of view in a modelling process

“Background-ladeness”

“Pragmatism-ladeness”
CoP + Context + Usage Portfolio

Intensional elements of background

Evolution of model elements

Focused activity

Modality

Confidence

Exactness

“Development process ladenness”

Usage of background

Current Date

Context

Previous modelling experience

accumulating experience

Content

Information

Concept

Topic

Service
Model
21.10.2015
©β/A.Da.

Overview
Services
Towards ...
Adequate
Dependable
Background
The MODEL
Informative
Service
Model Notion: Conception Frame

- **Instrument used as a model**
  - **Adequate**
    - **Dependable**
      - **Sufficient**
        - **Justified**
          - **Grounding**
            - **Basis**
              - **Background**
                - **Context**
                  - **When, where**
                    - **Usage spectrum and scenarios**
                      - **In what way, how**
                        - **Community**
                          - **Who**
                            - **Artefacts**
                              - **What, whereof**
                                - **Service**
                                  - **Model**
                                    - **Overview**
                                      - **Services**
                                          - **Towards**
                                            - **Adequate**
                                              - **Dependable**
                                                - **Background**
                                                  - **The MODEL**
                                                    - **Informative**
                                                      - **Service**
                                                        - **Content**
                                                          - **Information**
                                                            - **Concept**
                                                              - **Topic**
                                                                - **Services**
                                                                    - **21.10.2015**
                                                                        - © Β/A. Da.
                                                                                   - **interest, profile, roles, specific plays,**
                                                                                   - **rights, obligations, current practice**
                                                                                   - **functions in the scenario**
                                                                                   - **utilisation pattern**
                                                                                   - **implied purpose and goal**
                                                                                   - **utilisation methods**
                                                                                   - **capacity, potential**
                                                                                   - **which level of detail is really necessary**
                                                                                   - **empirical corroboration**
                                                                                   - **coherent conform**
                                                                                   - **validatable falsifiable**
                                                                                   - **stability plasticity**
                                                                                   - **internal quality**
                                                                                   - **external in use**
                                                                                   - **assurance evaluation**
                                                                                   - **basement, paradigms, postulates, restrictions,**
                                                                                   - **theories, culture, foundations, conventions, commonsense**
                                                                                   - **concepts, foundations, language as carrier,**
                                                                                   - **assumptions, thought community, practice**
                                                                                   - **application domain or discipline, disciplinary thought**
                                                                                   - **time, space, granularity, scope**
The Kiel Notion of the MODEL

Background of an instrument used as a model: (1) basis $B$ (basement, paradigms, postulates, restrictions, theories, culture, foundations, conventions, commonsense); (2) grounding $G$ (concepts, foundations, language as carrier, assumptions, thought community, practice); (3) context $C$ (application domain or discipline, school of thought, time, space, granularity, scope); (4) community of practice $P$ (interest, portfolio, profile, roles, specific plays, rights, obligations, current practice).

A MODEL is an instrument that satisfies for artefacts $A$ it represents:

Adequate for $A$, i.e., well-defined, $A$-analogous, $A$-reduced (or $A$-focused), and $D$-purposeful.

Dependable: (or workmanlike, professional), i.e., justified (corroborated, rational coherent and conform, falsifiable, stable and plastic) and sufficient (of firm quality and evaluated).

Functional models are models for which methods $M_D$ exists for utilisation in dependence on the end/purpose/function $D$.

Effective models are functional models that thoroughly function in the application scenario (‘usage games’).

Conceptual models are models that incorporate concepts or conceptions.
Service
Model
21.10.2015
©/A.Da.

Overview
Services
Towards...
Adequate
Dependable
Background
The MODEL
Informative
Service

Context
Develop, construct, configure, orchestrate, choreography

Model
as an instrument

Community of practice
role, right, obligation, play

Usage
business cases
installation, consolidate

Portfolio: Utilisation scenarios and use spectra

Development methods
Describe/characterise/picture/depict/represent/redevel
Create/produce/form/shape/design/arrange/organise/model
Evolve/migrate/redevelop
Corroboration/viability/justify/support/found/establish/prove/argue/substantiate/motivate/constitute/show/why
Evaluate/quality/argument/assess/internal quality/external quality/quality_of_use/modality/confidence

Content
Information
Concept
Topic

Paradigms, culture, background, foundations, theories,

Concepts, languages, routine, training, infrastructure, assumptions, though community, thought style, pattern, methodology, guidelines, practice

Basis

Grounding

Postulates, restrictions, authorities, conventions, commonsense
Informative Instruments

Metaphor: instruction leaflet

(1) serving to instruct or enlighten or inform
clarifying, ostensive, explanatory, instructive, demonstrative, expository, interpretive

(2) providing or conveying information
revealing, consultatory, exemplifying, advisory, telltale, instructive,

(3) tending to increase knowledge or dissipate ignorance
illuminating, enlightening

utilisation scenario: explanation, informed selection, and appropriation of an opportunity
information and directions for usage or
package insert (pharm), information and direction for use
enclosed label, product insert

enable people in directed, purposeful, rewarding, realistic, and trackable deployment of a service
within a given utilisation scenario

Goal: explanation, informed selection, and appropriation
Informative Models

(1) to give or impart knowledge of a fact or circumstance to

(2) to supply (oneself) with knowledge of a matter or subject:

(3) to give evident substance, character, or distinction to; pervade or permeate with manifest effect

(4) to animate or inspire.

(5) to train or instruct; to make known; disclose; to give or impart form to.

cargo

description of its adequacy and dependability

SMART and SWOT statements

It informs a potential users 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.
The Cargo of a Model

Mission: functions (and anti-functions or forbidden ones), purposes of the usages, potential and of the capacity

Determination: basic ideas, features, particularities, and the usage model of the given instrument

Meaning: main semantic and pragmatic statements about the model
describes the value of the instrument according to its functions in the usage scenarios
its importance within the given settings

Identity within the five kinds: actual, communicated, accepted, ideal, desired identity
Utility of an Informative Model

- 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.

SMART: how simple, how meaningful, how adequate, how realistic, and how trackable is the instrument quick shallow understanding

SWOT: strengths, weaknesses, opportunities, and threats of the given instrument to select the most appropriate instrument
Specification of the Problem Space

*State space* collection of all those states reachable from the *initial state*; desirable states, i.e. are *goal states*; properties such as suitability

*Actions* move from one state to another state under certain conditions; effect of the actions is observable; blocked or enabled depending on conditions used at some cost

*Goal test* whether a given state or state set satisfies the goals

*Problem solution controller* evaluates the actions; preferred; optima; based on evaluators beyond the *general Polya frame*

*Problem* based on abstraction techniques

*Context*

*Forces:* Constructors, associations, collections, classification, operations, views, derivatives, ...

*Solution*
Characteristics of an Informative Model

- 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.
Content Quality of an Informative Model of a Service

- PURE: positively stated + understandable (what is the service) + reward, ethically correct
- easy-to-survey and to understand,
- right formatting and form,
- supports elaboration and surveying,
- avoids learning efforts for their users,
- provides the inner content semantics and its inner associations,
- might be based on icons and pictographs, and
- presents the annotation and meta-information


Service W*H-Specification Frame

classical rhetorical frame
Hermagoras of Temnos: Quis, quid, quando, ubi, cur, quem ad modum, quibus adminiculis

Who, what, when, where, why, in what way, by what means

• $W^4$: wherefore (purpose), whereof (origin), wherewith (carrier, e.g., language), and worthiness ((surplus) value)

• secondary characterisation $W^{17}H$:
  • user or stakeholder or community of practice 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.
Service W*H-Specification Frame: Resulting Tasks

(1) \(W^4\) (Purpose, Origin, Carrier, Value): critical review of application portfolio, pre-cut of application domain (necessary, potential, never), languages (environment), profit and motivation

(2) \(W^{17}H\) (Community of practice): abilities, views, attention and personality, value of opinion and contribution, reliability, roles and responsibilities

(3) \(W^{17}H\) (Specifics of application domain): expectations, quality, foci, variations, future and evolution

(4) \(W^{17}H\) (Specifics of solution): programs, service after roll-out, quality and pricing, time and version management, coverage and capacity, integration, future migration

(5) \(W^{17}H\) (Context): environment, infrastructure, temporal constraints, integration with other solutions

Application systems as socio-technical systems.
# The W*H Specification Frame for the Conceptual Model of a Service

<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></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Content</td>
<td>Supporting means</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

... ...
The W*H Specification Frame for the Conceptual Model of a Service

<table>
<thead>
<tr>
<th>Service</th>
<th>Service Name</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Annotation</th>
<th>Source</th>
<th>Where of?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Party</td>
<td>Supplier</td>
<td>By whom?</td>
</tr>
<tr>
<td>Consumer</td>
<td>To whom?</td>
<td></td>
</tr>
<tr>
<td>Producer</td>
<td>Whichever?</td>
<td></td>
</tr>
<tr>
<td>Activity</td>
<td>Input</td>
<td>What in?</td>
</tr>
<tr>
<td></td>
<td>Output</td>
<td>What out?</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Added Value</th>
<th>Surplus Value</th>
<th>Worthiness?</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Where at?</td>
</tr>
<tr>
<td>Context</td>
<td>Systems Context</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Story Context</td>
<td>Where about?</td>
</tr>
<tr>
<td></td>
<td>Coexistence Context</td>
<td>Wither?</td>
</tr>
<tr>
<td></td>
<td>Time Context</td>
<td>When?</td>
</tr>
</tbody>
</table>
Conceptual Model for IT Service System

- Fundamental elements for developing applications;
- Organising 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 synchronise 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 Service Model

The W*H framework to service description

- with the fundament,
- with four governing directives,
- with technology pillars for development and usage,
- and with the application roof.
The Service Model

Planning the service

with foundation, background as the fundament,
with two of the four governing directives,
with freedom to choose technology
with partial consideration the application.
Building the service
using the background from the fundament,
with two of the four governing directives,
with technology for development
and with the application roof
Using the service

without consideration of the fundament,
with one of the four governing directives,
with technology pillars for usage
and with the application roof.
The Service Model

Evolving the service (classical case – without migration and revision)
Without reconsideration of the fundament,
with three of the four governing directives,
with technology pillars for development and usage
and with the application roof
Summarising

- Services are often considered only at the implementation layer
- Services provide an added value for the user, for the application
- Distinguish between application domain **description**, requirements **prescription**, and information system **specification**
- System solutions include social systems, computation systems, and organisational issues
- **W** specification frame for the full description
- All **W**-questions can be mapped to technology in a way that the systems developer understand the issue
- Our framework captures all requirements including the users
Thank you!

thalheim@is.informatik.uni-kiel.de
Enhancing Entity-Relationship Schemata for Conceptual Database Structure Models

ER 2015, Stockholm
www.is.informatik.uni-kiel.de/~thalheim/models.htm

20.10.2015

Bernhard Thalheim, (Marina Tropmann-Frick)
Christian-Albrechts-University Kiel, Germany (& Lomonosov University Moscow, Russia)
Many Kinds of Models

**Situation model:** reflection of a given state of affairs

**Perception model:** reflection of my current understanding of world

**Formal model:** based some formalism, within a well-based formal language

**Conceptual model:** based on formal concepts, conceptions

**Experimentation model:** guideline for experiment

**Mathematical model:** in the language of mathematics

**Computational model:** based on some (semi-)algorithm

**Physical model:** physical instrument

**Visualisation model:** for representation using some visualisation

**Representation model:** for representation of some other notion

**Diagrammatic model:** using a specific language, e.g. UML, electrotechnics

**Exploration model:** for property discovery

**Heuristic model:** based on some Fussy, probability, plausibility correlation

**Prediction model:** based on continuation assumptions, steadiness

... model:
In Which Cases is an (e)ER Schema a Model ???????

**Description scenario:** describing all viewpoints of business users

**Prescription scenario:** providing a blueprint for database system construction

**Explanation scenario:** what can be found in the DB and what not

**Documentation scenario:** what is currently found in a database system

**Communication scenario:** understanding the database

**Negotiation scenario:** sufficiency and necessity of all constructions

**Inspiration scenario:** ideas for an implementation

**Exploration scenario:** discover all elements of the database system

**Definition scenario:** define all elements in a proper form

**Prognosis scenario:** tell me what I could use in future

**Reporting scenario:** schema as a result of design

**Informative scenario:** why + wherefore + worthiness + wherewith
The Model is an Instrument
functioning in utilisation scenarios and use spectra

Judgement 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.

Instrument as such with some appreciation (Wertschätzung, Auszeichnung) well-shaped instrument on the basis of some criteria in dependence on intended utilisation and criteria for that it is accepted in a CoP, and that it is syntactically, semantically, pragmatically well-shaped, fits to the intended use, and is appropriate for the use spectra.
An Instrument that is Used as a Model

Given origins $\mathcal{A}$ (e.g. artefacts).

Any (real or virtual) instrument can be used as a model if it is adequate for $\mathcal{A}$, i.e., well-defined, $\mathcal{A}$-analogous, $\mathcal{A}$-reduced (or $\mathcal{A}$-focused), and $\mathcal{O}$-purposeful,

and

dependable (or workmanlike, professional):
justified (corroborated, rational coherent and conform, falsifiable, stable and plastic) and sufficient (of firm quality and evaluated).

The instrument has its 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).
### Examples of Functions, Scenario, and Spectra for Models

A model has a function (with some playout of this function) in [a] utilisation scenario(s) that are part of a use spectrum.

<table>
<thead>
<tr>
<th>Spectrum</th>
<th>Scenarios</th>
<th>Functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>(0) system development</td>
<td>description &amp; prescription</td>
<td>image &amp; blueprint for realisation</td>
</tr>
<tr>
<td></td>
<td>realisation &amp; coding</td>
<td>survey tool &amp; pre-image</td>
</tr>
<tr>
<td></td>
<td>realisation &amp; coding</td>
<td>inspiration tool</td>
</tr>
<tr>
<td></td>
<td>coding &amp; delivery</td>
<td>documentation tool</td>
</tr>
<tr>
<td></td>
<td>consolidation &amp; extension</td>
<td>archive, blueprint &amp; recording tool</td>
</tr>
<tr>
<td></td>
<td>proper realisation &amp; coding</td>
<td>ideal &amp; prototype</td>
</tr>
<tr>
<td>(2) theory development</td>
<td>step-wise reconsideration, refinement, theory evolution</td>
<td>auxiliary aid for theory construction, mediating tool for scientist, consideration tool, hypothesis structurer</td>
</tr>
<tr>
<td>(8) empiric research</td>
<td>causality detection among events, inverse modelling</td>
<td>rough illustrator, hypothesis structurer, dimensional tool, correlation aid</td>
</tr>
<tr>
<td>(10) teaching</td>
<td>aid for pedagogues</td>
<td>illustrator, visualiser</td>
</tr>
</tbody>
</table>
Information System Models in General

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.

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.

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.

IV. Explanation and prediction (EP): Says what is, how, why, when, where, and what will be.

Provides predictions and has both testable propositions and causal explanations.

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 artefact.
A Database Schema in the Extended Entity-Relationship Model: The Hotel Reservation

- Country
  - Code
  - Currency
  - Name

- Hotel Chain
  - Code
  - Name
  - URL

- Star Rating
  - Code
  - Image
  - Period

- Hotel
  - AddDetail
  - Email
  - Contact(...)

- Location
  - City
  - Postcode
  - Street

- Room
  - Smoking?
  - Number
  - Floor
  - ActualRate
  - StandardRate

- Room Type
  - Code
  - Description

- Rate
  - Type
  - Code
  - Description
  - Amount
  - SpecialOffer

- Rate Type
  - Code
  - Description

- Condition
  - From
  - Specific Characterisation
  - DefaultValue

- Characteristic
  - Code
  - Description

- Booking
  - BookingNo
  - RoomCount
  - BookingDate

- Booking Status
  - Code
  - Description

- Agent
  - AgentID
  - Contract

- Agent
  - Name
  - History

- Guest
  - GuestNo
  - Address(...)
  - Member
  - SpecialRequ
Is a Schema in the Extended ER Model (HERM) a Model?

**HERM allows representation of nested sentences but without abstractions!**

**Grounding:** compositionality; oneness of world; separation into collections; predicate logics based interpretation; sets; object-relational; inductivity; homomorphic representation

**Basis:** orthonormalised representation; relatively independent objects; diagrammatic representation; relative stability; redundancy bounded; explicit representation; OR infrastructure with corresponding mapping; lowest meaningful granularity; weak value-based representation; implicit inherited constraints; flat diagrams; fixed granularity

**Deployment:** sufficient for DBS construction and application description; for documentation; partially for exploration and assessment (validation, verification, testing); not for explanation or prognosis or optimisation or integration

**CoP:** IS professionals

**Context:** stable applications; evolution-sensitive

**Quality:** internal and external quality; problematic quality of use

**Justification:** corroboration depending on orthonormalised representation; falsifiability only if co-design; complete stability; no modality; closed world confidence; variants of representation due to MVD
Entity-Relationship Schema: Not a Database Model; Really???

Discussion started by Fabian Pascal, John Sullivan following Codd

Database: formal logical representation of some real world segment of interest

Database modelling as the design of a database structure that satisfies the business requirements and provides a strong technical platform for growth and integration.

Model goal: construction

Model purpose: design & action, realisation (downward to the database designers and programmers)

Model function: description & prescription for implementation, coding for performance

- database types
- database constraints
- database operations

Relational schema: Database model

Designer’s view: tables, system, network, interface, flow, rule
ERM schema is for business not for databases

Model goal: perception

Model purpose: communication (upwards towards the solution architect, business analysts, etc.)

Model function: H2H dialogue, explanation, survey, sign-off, reflection

- database types reflecting reality
- business rules
- business activities

ERM schema: Information systems model

Architect’s view: ER, functions, links, use, TA, rules
Background of IS Schemata

**Purpose is system construction**: 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

**Purpose is communication**: viewpoint and flavour representation; flexible usage (full logical independence); variable name space representation; methods for reasoning, understanding, presentation, exploration; methods for explanation, check, appraise, experience
Adequacy of eER Schemata

(1) Well-defined: 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

(2) A-analogous: structural analogy (homomorphic, but not qualitative, functional) resulting in structural alignment; metaphysical, epistemological and heuristic adequacy

(3) A-reduced (or A-focused): compactness, no repetition, high-level descriptive abstraction; conceptual minimal

(4) D-purposeful: either for construction of another artefact (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)
Dependability of eER Schemata

(1) Justification: embedded in the understanding of the application domain, external corroboration; internal corroboration due to the languages; acquisition and elicitation backed conform to standards accepted in CoP; Salami slice tactics; correctness; restriction to essential business items; approved, closed world schemata; partially evolution prone, partial flexibility; simple diagramming with overlay diagrams

(2) Sufficiency: completeness, naturalness, minimality, system independence, flexibility, self-explanation, ease of reading and using of firm quality and evaluated mainly quality in use properties with no error tolerance no or only necessary redundancy; no imposed implementation restrictions; internal and external qualities as blueprint without requirement for completeness of constraint sets; natural keys, no mega-attributes; complete confidence in all its components
Conceptional ≠ Conceptual

Concepts: used for classification or are all the knowledge that the person has, and associates with, the concept’s name.

Conceptions: however systems of explanation.

Conceptional modelling: performed by a modeller that directs the process based on his/her experience, education, understanding, intention and attitude

Conceptual: something is using/incorporating/integrating concepts

Conceptual models: products that are used by other stakeholders, incorporate concepts, use a language as a carrier, are restricted by the expressiveness of this carrier.
Enhancing the eER Schema

Communication and negotiation scenario
exchange of meanings through a common understanding of notations, signs and symbols within an application area also back-and-forth process: reconcile, compromise, agree model: negotiable and debatable propositions about the understanding no well-developed justificatory explanations “worthiness”, “what”, “wherein”, “wherefrom”, “for what”, “where”, “whence”

Conceptual database structure model:
database schema
\[ (\mathcal{S}, \{ (\mathcal{V}_i, \Phi_i) \mid i \in \mathcal{U} \}, \{ (\mathcal{P}_i, \Psi_i) \mid i \in \mathcal{U} \}, \mathcal{A}, \mathcal{D}) \]
Enhancing the eER Schema

Conceptualisation scenario

conceptualisation of database applications
shuffled with discovery of phenomena of interest
  analysis of main constructs
  focus on relevant aspects within the application area.

  incorporates concepts from the application domain
“why”, “what”, “which basis”

Conceptual database structure model:
database schema
  ⊕ a collection of views for support of business users
  ⊕ mapping for schema elements to the common concept field
  ⊕ declaration of model adequacy and dependability

(S, V, M, A, D)
# Enhancing the eER Schema

## Description scenario

specification how the part of the reality of interest augmentations of current reality are targeted
“what”: structure of an envisioned database; “where used” “why”, “whereto”, “for when”, “for which reason”, “wherefore”

## Conceptual database structure model:

**database schema**

- ⊕ collection of views for support of business users
- ⊕ collection of a commonly accepted reality models with explicit association to views
- ⊕ declaration of model adequacy and dependability

\[
(S, \mathcal{B}, (\mathcal{R}, \mathcal{S}), \mathcal{A}, \mathcal{D})
\]
Enhancing the eER Schema

Prescription scenario

blueprint for or prescription of a database application
especially structures and constraints in such applications
structure and how and where to construct the realisation
also with transformation profiles
“how”, “when”, “where”, “how to do”

Conceptual database structure model:

database schema
⊕ collection of views for both support of business users
   and system operating
⊕ realisation template
⊕ declaration of model adequacy and dependability

$\left( S, \mathcal{V}, \mathcal{T}, \mathcal{A}, \mathcal{D} \right)$.
Enhancing the eER Schema

Documentation scenario
structure, purpose, operation, restrictions, and other requirements
“what will be”, “how used”, “by what means”, “in what way”,
“supporting means”, “wherewith”, surplus value,
context (whereat, where about, wither, when), sources (whereof),
activity (what in, what out), party (by whom, to whom, whichever)

Conceptual database structure model:
database schema
⊕ association to origin (relational, ...) schema
⊕ collection of views for both support of business users
and system operating (option)
⊕ collection of operating structures (option)
⊕ interpretation of origin notions by concepts (option)
⊕ some kind of declaration for adequacy and dependability (option)

(S, (R, M), [V, ] [O, ] [I, ] [A, ] [D]) .

Many practitioners primarily use the eER model this way!!!
Thank you!

thalheim@is.informatik.uni-kiel.de