On the Relevance of Models in Information Systems Engineering

Dr. Ralf-Detlef Kutsche
TU Berlin and Fraunhofer ISST/FIRST/FOKUS

Colloquium @ UPC / ESSI
Barcelona, Sept 18, 2013

Models, Models, Models!

E = m * c²

AGENDA

- My Personal History & Background

- Focus Area: Model Based Software & Data Integration
  - Languages and Models (... the basics ...)
  - Metalanguages and Metamodels, i.e. „playing with languages“

- Project Experiences 2007 thru 2013
  - Application towards Business Scenarios (... software and data integrators: only SMEs !!)
  - BIZCYCLE: Model Based Software & Data Integration
  - BIZWARE: Domain Specific Languages

- My Present Work: Teaching Relevance of MBSDI and Future Project Perspectives

Large Scale Information Infrastructures
Specifications & Modeling ... my personal history ...

- PAtientenDatenKOMmunikation PADKOM (Patient Data Communication) / Heterogeneous Distributed Information Management System HDMS at German Heart Centre Berlin DHZB (1989 – 1993+)

- LandesUmweltInformationsSystem LUIS Brandenburg (1994 – 2000+) (State Environmental Information System Brandenburg)


PADKOM / HDMS @ DHZB
since 1989

- Development of the HDMS …
- … as an object-based, distributed, heterogeneous information system in the domain of cardiology at German Heart Centre of Berlin DHZB …
- … basically enabling communication and an integrated view of the various kinds of patient data … (later also: data integration)

LUIS
Brandenburg
since 1994

- Integration of environmental information sources via object-oriented middleware …
- Object-based approach …
- Metadata-based approach …

TMF Germany
since 1998

- Integrating approx. 200 medical centers performing clinical studies wrt. information standardization & exchange, and wrt. standardization of processes

European Migration Network
since 2004

Goals of the European Union:
- Comprehensive view on the migration and asylum situation
- Harmonization of European strategic policy & legislation

Task for the European Commission:
"Pilot Implementation" of the European Migration Network EMN w.r.t.
- Contact Information
- Publications
- Legislation
- Statistics
- …
Lessons Learned from the Experiences …

• Model-based development is essential in order to guarantee flexible, stable and sustainable software solutions and information systems.

• Metadata definition, standardization and management is essential in order to allow for data exchange and integration.

• Semantic concepts help to improve ‘real’ semantic integration.

• Graphical languages help in faster perception of complex structures and relations and often help to avoid inception phase errors.

• Object-oriented paradigm helps to unify structural (data) view and dynamic (functions, processes, interactions, workflows) views.

• Rich middleware platforms are essential in order to allow for complex interoperability (e.g. transactions, security, ...).

• Solid mathematical / theoretical background is absolutely useful.

What is a Model? – Some Definitions…

A model is information ...

- on something (content, meaning)
- created by someone (sender)
- for somebody (receiver)
- for some purpose (usage context)


Abstraction is the key-concept to build models

- derive information from different viewpoints
- derive the essence, the characteristics, the lawfulness (‘Gesetzmäßigkeit’) from a set of different individuals
- make relationships between concepts visible by deleting details

Examples of Models

- **Mathematical Models**
  e.g. CIRCLE (an idealized set of points in two-dimensional space)

- **Architectural Models**
  e.g. KING'S CASTLE (an idealized wish of an old/new building in the center of Berlin)

- **Technical Models**
  e.g. a prototype of a new car, but also all previous technical sketches, specifications, calculations, simulations, ... (an idealized imagination of several communities)

In our setting of Software and Information Systems:

- Models of the real world in order to achieve software solutions with given properties:
  - Mapping of the reality
  - Abstraction from (too many) details of the reality: reduction/simplification
  - Pragmatics/Feasibility

What is Language?

- **A language** is a dynamic set of visual, auditory, or tactile symbols of communication and the elements used to manipulate them.

- A set of commonly accepted symbols is only one feature of language; all languages must define the structural relationships between these symbols in a system of grammar. Rules of grammar are one of the characteristics sometimes said to distinguish language from other forms of communication. They allow a finite set of symbols to be manipulated to create a potentially infinite number of grammatical utterances.

- Languages can be (among other classifications) subdivided into:
  - Formal language / Artificial language, mathematical and other languages created for a specific purpose, like e.g. first order logic (formal) or modeling/programming languages (artificial/semiformal)
  - Natural language, a language used naturally by humans

(A collection of statements from Wikipedia, modified and shortened according to my intentions)

Syntax / Semantics / Pragmatics (I)

- **Syntax**: Syntax is the study that relates signs to one another.

- **Semantics**: Semantics is the study that relates signs to things in the world and patterns of signs to corresponding patterns that occur among the things the signs refer to.

- **Pragmatics**: Pragmatics is the study that relates signs to the agents who use them to refer to things in the world and to communicate their intentions about those things to other agents who may have similar or different intentions concerning the same or different things.


Syntax / Semantics / Pragmatics (II)


In the linguistics of both natural and computer languages, the terms *syntax, semantics and pragmatics* are used to categorize descriptions of language characteristics.

- **Syntax**: The syntax of a language describes the structure and composition of allowable phrases and sentences of the language.
- **Semantics**: But syntax itself is devoid of meaning, simply telling us what strings are valid and how they may be parsed or decomposed. The meaning of these syntactic elements must be provided through semantics. In essence, we may think of providing syntactic elements as inputs to a semantic function, which in turn provides some representation of the meaning of the elements as output.
- **Pragmatics**: Pragmatics is the third general area of language description, referring to practical aspects of how constructs and features of a language may be used to achieve various objectives.
Sowa (2000) writes about Ogden & Richards' (1923) triangle of meaning: "The triangle has a long history. Aristotle distinguished objects, the words that refer to them, and the corresponding experiences in the psyche. Frege and Peirce adopted that three-way distinction from Aristotle and used it as the semantic foundation for their systems of logic. Frege's terms for the three vertices of the triangle were Zeichen (sign) for the symbol, Sinn (sense) for the concept, and Bedeutung (reference) for the object."

Literature:
Example: Modeling „Knowledge“

- Example: Customer Relationship Management (CRM) Ontology
  (starting point, finally very complex with lots of logical constraints in the background ...)

(Source: BIZCYCLE Project, TU Berlin, 2009)

Software Modeling: System Functionality

- Example: Modeling (coarse) functionality of a system for telephone orders (UML use case diagram)


Software (Behaviour) Modeling: Statecharts

- Example: Modeling the (concurrent) states of a student during a university project course (UML statecharts)


Modeling example: Activity & Document Flows

- Health Care Example:
  Combined Workflow and Document Flow Diagram including the dimensions of time, space (local org units), and responsibility
  (Patient registration für heart surgery, excerpt)

(Source: Ralf-D. Kutsche, 1990, Deutsches Herzzentrum Berlin)
Another important concept that applies to all aspects of language description is that of **metalanguage**. Metalanguage, in general, refers to the language in which a subject language is being described. For example, BNF (Backus-Naur Form) is a metalanguage widely used to describe the syntax of programming languages. Similarly, there are formal metalanguages for describing the semantics of programming languages, particularly associated with the approaches of axiomatic semantics and denotational semantics. In most cases, however, a less formal approach to semantic description is taken, using English as the metalanguage.

It is important to note that metalanguages are indeed languages in their own right. In particular, one should expect that metalanguages each have their own syntax, semantics and pragmatics, which in turn must be described by a **metametalanguage**. Typically, a combination of English prose and standard mathematical notation is used as metametalanguage.


---

**Language Definition: Example Use Case M.M.**

- metamodeling: abstract syntax of a modeling language, i.e. its concepts!

(from: OMG UML Specification v1.5, OMG document formal/03-03-01, March 2003)

**Language Definition: another example ...**

- BNF (Backus-Naur-Form), here mixing textual and graphical elements!

(Source: an industrial training by Ralf Kutsche, 2011)
UML 2 Discussion of MOF levels

![Diagram: Four-layer Metamodel Hierarchy]

Figure 7.8 - An example of the four-layer metamodel hierarchy

Further Reading in Metamodeling


AGENDA

- My Personal History & Background
  - Focus Area: Model Based Software & Data Integration
    - Languages and Models (... the basics ...)
    - Metalanguages and Metamodels, i.e. „playing with languages”
  - Project Experiences 2007 thru 2013
    - Application towards Business Scenarios (... software and data integrators: only SMEs !!)
    - BIZCYCLE: Model Based Software & Data Integration
    - BIZWARE: Domain Specific Languages
  - My Present Work: Teaching Relevance of MBSDI and Future Project Perspectives

BIZCYCLE @ TU Berlin 2007 – 2010

Methodology, Design & Development of the Model-based Interoperability Platform MBIF for Software and Data Integration

Our Project team:
- Dr. Ralf-D. Kutsche, Dr. Nikola Milanovic
- Henning Agt, Gregor Bauhoff, Timo Baum, Mario Cartsburg, Hatice Elmasgünes,
  Daniel Kumpe, Michael Shtelma, Jürgen Widiker
  + Students ...
Motivation 1970 – 2070 ...

- Integration of heterogeneous distributed IT-systems is one of the major problems and cost-driving factors in the software industry (approx. 50% of total IT-cost, 80% of total software cost in integration issues ...)

- There is an increasing need to systematically address integration in accidental information infrastructures and IT-architectures, that have grown over time in an uncontrolled manner in heterogeneous enterprise environments

- Integration needs:
  - Data and information integration
  - Software integration (full interoperability)

Model-based SWE / SDI : lots of promises

- Model-based development helps in sustainability (models are more stable than code ..., abstraction level of documentation, ...)

- Model-based integration saves approx. 70% of integration costs (statements by Bran Selic IBM/Rational, and: experiences from our BIZYCLE experiments)

- Fully model-based approach (including process mgmt, and generated code/ model execution) can save even more than 90% of the total development cost (statements by András Pataricza, Univ. Budapest)

BIZYCLE Vision 2006

- Methodology & technology platform for enterprise-wide (even: cross-enterprise) integration of software solutions and components in business intelligence
- Transparent access to all business relevant applications
- (Semi-) Autamitized integration of applications for various kinds of business process improvement
- Cross-system evaluation of enterprise relevant data (KPIs..)

BIZYCLE – slogan: (our industrial partners)

Plug and Play Your Business!

or: (at university)

Generate Software Connectors (semi) Automatically !

BIZYCLE Philosophy and Methodology 2006

- Integration scenarios are modeled at different abstraction levels:
  - Computation independent model level (CIM)
  - Platform independent model level (PIM)
  - Platform specific model level (PSM)

- Integration methodology:
  - Component/Interface Analysis
  - Platform Specific (Meta-) Models
  - Transformation to Platform Independent (Meta-) Models
  - Conflict Analysis
  - Connector Generation
**(Meta-)Modeling Methodology**

- **Methodology:** Applying MDA methodology (CIM, PIM, PSM meta-/modeling)

- **Application:** Integration Scenario Modeling (CIM level)
  (starting from UML diagrams for use cases, interactions, activities, etc. ... later using the MBIF toolsuite ...)

- **Research:** Metamodel Development

- **Application:** Modeling Interface Descriptions / Metamodel Instantiation for Integration Scenarios (PSM level)
BIZYCLE (Meta-) Models

- **CIM**: Computation Independent Model for integration requirements
- **PSM**: Platform Specific Models for interface descriptions (many!)
- **PIM**: Platform Independent Model for interface descriptions
- **SM**: Semantic Model (domain ontology)
- **AM**: Annotation Model for semantic annotations
- **CM**: Connector Model

Computation Independent Metamodel (CIM)

- Captures business scenario with control and data flow
- Scenario requirements
- Support conflict analysis

CIM Example

- Web shop and Payment System

BIZYCLE – Modeling Interface Description

- Describe ‘technical’ information about interfaces that are to be integrated:
  - Interface static signature (types)
  - Interface behavior
  - Communication protocols
  - Non-functional properties

- Systems under study, i.e. development of PSMMs:
  - ERP (e.g., SAP R/3 BAPI/IDOC)
  - Relational and XML databases
  - J2EE and .NET components/applications
  - Web Services
  - Flat files (e.g., XML, CSV...)
Platform Specific Metamodel for J2EE

- Communication

<table>
<thead>
<tr>
<th>CommunicationChannel</th>
<th>JavaNamingProviderURL</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMI_BOP</td>
<td>javaNamingFactoryInitial</td>
</tr>
<tr>
<td>JNDI_Context</td>
<td>JnpSocketFactory</td>
</tr>
<tr>
<td>JNDI_Property</td>
<td>JnpNamingFactoryURL_PKGS</td>
</tr>
<tr>
<td>User</td>
<td>UserTimeout</td>
</tr>
<tr>
<td>Password</td>
<td>JnpSetTimeout</td>
</tr>
</tbody>
</table>

Platform Independent Metamodel

- PIM interface descriptions are generated from PSM interface descriptions by model transformation
- Abstract platform specific issues from PSM interface descriptions by generalizing interface properties
- Presents BIZ CYCLE middleware with a unified view of component interfaces
- Goal: to enable conflict analysis of heterogeneous components

Semantic Metamodel

- Domain Ontology for Integration Scenario
- Controlled vocabulary
- Based on Resource Description Framework (RDF)
- Subject – Predicate – Object (RDF - triple)
- Needed for Annotations to PSM, PIM and CIM Elements
Semantic Model Example

- Set of RDF triples: a *Trade* domain ontology

Annotation Metamodel

- Connects semantic to different models (CIM, PSM, PIM)

Connector Metamodel

Example Connector Model
Integration Conflict Types

- **Semantic Conflicts**
  - Address semantical aspects of elements to be integrated, i.e., differences in the meaning they convey
- **Behavior Conflicts**
  - Concern dynamic aspects and are primarily caused by functional constraints of interfaces to be integrated
- **Property Conflicts**
  - Address the characteristics of components to be integrated, e.g., QoS properties
- **Data Structure Conflicts**
  - Pertain static aspects of components participating in the integration process and are caused by the differences in the data structures
- **Communication Conflicts**
  - Are differences in communication aspects, e.g., conflicting communication protocols

Semantic Annotations

- **Data-oriented annotations** are realized by references to domain objects, i.e., ontology concepts which describe data elements
  - **CIM Level Annotations**: Annotation of business objects, i.e., data elements participating in the integration scenario
  - **PSM Level Annotations**: Annotation of data elements described in various platform-specific models, e.g., data records, segments, and fields of a SAP R/3 system
  - **PIM Level Annotations**: Annotation of data elements described in the platform-independent models which are results of PSM-to-PIM transformation
- **Function-oriented annotations** are realized by references to domain functions, i.e., ontology concepts which describe actions
  - Annotation of interfaces on the PSM, PIM, and CIM levels
Full BIZCYCLE Integration Process

Domain Specialist

Platform Specialists
(DBS, SAP, XML, J2EE, ...)

Business Architect

Ontology Editor

Interface Editor

Process Editor

Integration Specialist

Connector

M2C Transformation (JET, OAW)

Model Interpretation

M2M Transformation (ATL, Viatra)

Model-based Integration Framework (MBIF)

Integration Platform

Runtime Environment

Repository

► Based on standard Eclipse
► Automatic interface extractors
► Model editors
  ▪ Flow/semantic models (CIM, SM, part of CM): graphical (GMF)
  ▪ Structural models (PSMs, part of CM): graphical + tree-based
► Model transformer
  ▪ PSM + AM -> PIM + AM
  ▪ CIM + CAM + PIMs -> CM
► Conflict analyzer/resolver
► Code generators
  ▪ PSM -> connector code (application endpoints)
  ▪ CM -> connector code (business logic between endpoints)

Repository Integration Platform

MBIF: 1.7 mio lines of code, thereof 1.3 mio generated

Model-based Integration Framework (MBIF)

Integration Platform

Runtime Environment

Repository

► GlassFish ESB
  ▪ Sun-BPEL for business logic
  ▪ Java BPEL extension for EAI patterns
  ▪ Message- and service-oriented realization
  ▪ Standardized target environment => portable code generators
  ▪ Powerful QoS/management support
► Model interpreter
  ▪ Independent Java component
  ▪ EMF and Java Reflection for PSM interpretation
  ▪ Low target system requirements
  ▪ Supports online model changes

Model-based Integration Framework (MBIF)

Integration Platform

Runtime Environment

Repository

► Subversion
  ▪ Versioning support for all kinds of artifacts
  ▪ Teamwork support
  ▪ Supports artifact metadata
► Jena framework
  ▪ Based on RDF and MySQL
  ▪ Supports artifact relations for consistency preservation
► Apache Maven
  ▪ Artifact relation management
  ▪ Artifact life cycle/build management
  ▪ Existing integration with SVN
Our Project team @ TU Berlin:
Dr. Ralf-D. Kutsche, Dr. Nicole Natho
Henning Agt, Yan Li (partially), Yuexiao Li
+ many part-time students:
Amauri Albuquerque, Andreas Büscher, Nico Franecke, Amir Matallaoui,
Silvia Sandy-Martinez, Kamran Mohtadi, Lakshmi Vuyyuru, Andreas Wolf

Publications

- Executable Domain Specific Language for Message-based System Integration, M. Shtelma, M. Cartsburg and N. Milanovic, ACM/IEEE 12th Int. Conf. on Model Driven Engineering Languages and Systems (MODELS), Denver, USA, 2009
- (Meta-) Models, Tools and Infrastructures for Business Application Integration, Kutsche R., Milanovic N. Proc. 7th Int. Workshop on Conceptual Modeling Approaches for e-Business (EComo 2008), Klagenfurt, Austria, 2008

BIZWARE – Solution for the Complete Software Life Cycle

BIZWARE – Solution for the Complete Software Life Cycle

Softwarelebenszyklus

- Spezifikation
- Definition Anforderungen
- Analyse und Bewertung
- Komponenten entwickeln
- Komponenten integrieren
- Anwendungen betreiben
- veränderte Anforderungen

BIZWARE Konzerngruppe

BIZWARE Domain Specific Languages
“Methods, Languages, Tools and Infrastructure for the Model and Software Factory of the BIZWARE Regional Project Group“
**What are Domain Specific Modeling Languages?**

- Small modeling languages, tailored for a specific domain
- Domain specific (graphical) notation
- Easier modeling for domain (NOT modeling!) experts
- Reuse, Combination, …

Defining an DSML:
- Abstract syntax
- Concrete syntax
- Language constraints (Semantics)

---

**General Purpose Mod. Languages (like UML) vs. DSLs**

**UML and DSL: the Lego Metaphor**

- UML is like a pile of basic LEGO blocks with few colors, sizes and shapes
- DSL is like the special LEGO kit for Medieval Knight Castle
- With the basic LEGO kit you can build only average castles
- With the special Castle kit you can build excellent castles
- On the other hand, with the basic LEGO kit you can also build average cars or planes
- But with the special Castle kit you can build only really terrible cars or planes
Prototyope of a DSL for GUIs
- Graphical User Interface Development based on Wireframes –

Collaboration TU Berlin and Akquinet/Tech@Spree

GUIs in Different Devices / Basic GUI Elements

Classical computers, tablets, smart phones, technical controllers, healthcare devices, etc. …

• Shapes, Colours, Frames, …
• Text Fields
• Images
• Tables
• Combo Boxes
• Buttons
• …

GUI Meta Model

BIZWARE GUI DSL & Other DSLs
… finally, GUIs to be integrated with insurance applications and mobile devices…
Abstract Syntax

Our Application

```
"uModel": { "name": "ID" },

Page = Model

"Form": { "name": "ID" },

action: action "STRING"

("id": "ID")

("class": "style": "STRING")

("id": "ID")

("aElements": ")->

Page = Elements

"Button": "Input": "Radio": "Checkbox": "TextArea": "SelectBox": "DataEntry": "Img": "Link": "YNQuestion": "Div": "Label"

```

Example - Login Page

```

```
<div class="row">
  <div class="control-group">
    <div class="control-label">
      <div class="input-large">
        <input id="username" name="username" type="text" placeholder="Username" />
      </div>
    </div>
    <div class="controls">
      <label class="required">Username</label>
    </div>
  </div>

  <div class="control-group">
    <div class="control-label">
      <div class="input-large">
        <input id="password" name="password" type="password" placeholder="Password" />
      </div>
    </div>
    <div class="controls">
      <label class="required">Password</label>
    </div>
  </div>

  <div class="control-group">
    <div class="controls">
      <button class="btn">Login</button>
    </div>
  </div>
</div>
```

```
Case study: Domain specific language in facility management (DSLFM)

Master thesis Kamran Mohtadi

Definition: Facility Management

- “Facility management is a profession that encompasses multiple disciplines to ensure functionality of the built environment by integrating people, place, process and technology.” (International facility management association)

- Dimensions of facility management according to European Network of Facility management / Standard EN15221: (http://www.eurofm.org/knowledge/en15221/)
  - Space and infrastructure
  - People and organization

  in detail:
  - Cleaning services
  - Pest control
  - Catering & vending services
  - Furniture procurement
  - Office supplies
  - Maintenance
  - Logistics
  - Event management
  - Energy management
  - Utilities
  - Landscaping & parking
  - Design and construction services
  - Lease
  - Rental & space management
  - Corporate real estate services
  - Workplace management
  - Safety & security services
  - (internal) Relocation services
  - Document management
  - Hospitality/reception management
  - Travel services
  - Property administration

Computer Aided Facility Management (CAFM)

- CAFM (Computer Aided Facility Management Software) or TIFM (Total Integrated Facilities Management Software) software supports the specific processes of facility management and also the people involved in these processes
- Core processes supported in CAFM systems:
  - Stock documentation, land management, move management, contract management, rent management, operating cost management, cleaning management, key management, energy controlling and maintenance management.

- eTask.FM-Portal: central platform structure as the basis for the connection of all other modules.
- eTask.FM-Portal is based on several data tables, which store data regarding processes, people, emails, activities, …

Case study: DSLs in Facility Management (DSLFM)

- Status before introducing DSLFM:
  - Central database is configured manually for each new process
  - Manual configuration according to MS Visio based process / data flow diagrams called “Infographs”
  - Infograph models widely accepted within the company and by the customers
  - Problems:
    - Manual configuration of the database for each scenario change is time consuming and error-prone
    - Identification of similar processes
  - Goals of the master thesis:
    - Automation of database configuration
    - Keeping the graphical notation of Infograph models
    - “Real” modeling environment including code generation (MS Visual Studio)
Case study: DSLFM – Infograph models

- Domain analysis through Infograph models and additional data from eTask

Results: DSLs in Facility Management (DSLFM)

- DSLFM / Model transformation from Infographs to “real” FM Models in MS Visual Studio
  - A Domain Specific Language for Facility Management
  - Developed in MS Visual Studio DSL Tools
  - Similar GUI to Infograph models
  - Proved to automatically generate in the biggest given scenario up to 11000 lines of SQL code for database configuration
  - Avoidance of errors through constraints and validity checks
  - Already in productive use by eTask in a few scenarios, replacing infograph models and manual configuration of the eTask.FM-Portal DB
Research Goals for Henning Agt’s PhD Thesis

- **Vision 1:** Provide automated suggestions of semantically related model elements for domain modeling (focus on domain terminology and conceptual design)

- **Vision 2:** Try to remove errors from domain models (focus on relationships)

Summary BIZWARE Strategy

- Continuation of the experiences in BIZCYCLE on a new basis:
  - Addressing the complete software development cycle, but (feasible) restrictions of manageable domains: analysis, requirements definition, specification, design, development, integration, operation, evaluation, etc.

- Focus on the foundation of domain-specific languages:
  - Meta modeling, model instantiation, model transformation, multi modeling, model integration, semantic modeling (ontologies, semantic annotations, etc.), test generation, process support

BIZWARE – Research Results

**Diploma and Master Theses 2011 - 2013**


- **Kamran Mohtadi** (Diploma thesis, TU Berlin, April 2013, Advisor: Dr. Ralf-D. Kutsche) „Development of Domain Specific Languages in the Software Industry - a Practical Case Study”.

- **Silvia Teresa Sandy Martinez** (Master thesis, UPC Barcelona/TU Berlin, July 2013, Advisors: Henning Agt, Dr. Ralf-D. Kutsche) “Patterns in Domain Models”.

BIZWARE – Research Results

**Scientific Conferences 2011 - 2013**


AGENDA

- My Personal History & Background

- Focus Area: Model Based Software & Data Integration
  - Languages and Models (… the basics …)
  - Metalanguages and Metamodels, i.e. „playing with languages“

- Project Experiences 2007 thru 2013
  - Application towards Business Scenarios (… software and data integrators: only SMEs !!)
  - BIZCYCLE: Model Based Software & Data Integration
  - BIZWARE: Domain Specific Languages

- My Present Work: Teaching Relevance of MBSDI and Future Project Perspectives

My Modeling Background / History

- Lectures in Information & Software Modeling / Student Projects
  - Grundlagen der Informationsmodellierung (TU Berlin, since 1994)
  - Advanced Modeling (TUB, since 2008; HCMUS, VietNam, since 2010)
  - Heterogeneous Distributed Information Systems HDIS (TUB, since 1996)

- Projects for Industry and Public Services (FIS / HDIS, since 1989)
  - Health Care (German Heart Center Berlin, TMF Clinical Studies, …)
  - Environmental Information Systems („Landesumweltinformationssysteme“)
  - European Migration Network (Society: Migration & Asylum, Politics, …)
  - Consultancy for Software Enterprises (Software Modeling, …)

- Industrial Cooperation Projects: BIZCYCLE & BIZWARE
  - SME project partners in the areas of: Health Care, Production/Logistics, Facility Management, Publishing, Finance, … (funded by German gov.)

Personal Intro / related to IT4BI

- Dr. Ralf-Detlef Kutsche has a position as ‘Academic Director’ in the Database and Information Management (DIMA) group at TU Berlin (previously CIS group, since 1994), and as research coordinator at different Fraunhofer institutes for applied research (now: FOKUS)
  - Ralf-Detlef.Kutsche@tu-berlin.de / Ralf-Detlef.Kutsche@fokus.fraunhofer.de

- Member of the IT4BI steering board / selection committee since 2012

- Teaching activities @ TU Berlin:
  - Advanced Information Modeling (Bachelor CS, TCS, BI, et al.)
  - Heterogeneous and Distributed Information Systems (Master CS, TCS, BI, IT4BI, et al.)
    - Development and integration of modern distributed, heterogeneous information systems based on the concepts of
      - metamodel (i.e. language), model and data integration
      - software architecture and interoperability, based on classical middleware ideas and patterns
      - metadata management for Business Intelligence.
  - Basic Courses in Programming and Databases (Bachelor CS & BI)
  - Modeling Seminars & Student Projects (Master CS, TCS, BI, et al.)

Some Topics for forthcoming Masters &PhDs

Modeling Semantics / Creating Ontologies:


- Semantic Annotations for Information Integration / Metadata

Modeling Language Design:

- Graphical Domain Languages (Graphical DSMLs)

- Concrete Syntaxes based on Icons / Drawings / Domain Knowledge

- Refinement Techniques for Graphical Languages