Preface

The area of Information Systems Engineering (ISE) combines organizationally oriented topics often dealt with in Management Information Systems (MIS) with more technical concerns found in Software Engineering (SE). This area bridges different areas of information systems development such as analysis, requirements specification, and design. The International Journal of Information System Modeling and Design (IJSMD) publishes original research and practical results on the advances in system analysis and design, and this book is based on the papers published in 2011 in this journal.

The mission of IJSMD is to provide an international forum for modeling experts and design professionals for exchanging innovative ideas. The journal enables presentation of original work on the development of models for blending software and enterprise engineering, managing complexity of design, visualization of integration, and evolution. As we understand, modeling on all levels is an important area. Modeling languages can be divided into classes according to the core phenomena classes (concepts) that are represented and focused on in the language. This has been called the perspective of the language (Krogstie 2012), and eight core perspectives have been identified:

1. **Behavioral perspective**: Languages following this perspective go back to the early sixties, with the introduction of Petri-nets. In most languages with a behavioral perspective the main phenomena are “states” and “transitions” between “states.” State transitions are triggered by “events”.

2. **Functional perspective**: The main phenomena class in the functional perspective is the “transformation.” A transformation is defined as an activity, which based on a set of phenomena transforms them to another set of phenomena. This perspective goes back to at least the seventies with DFDs. Modern approaches for process modeling typically combine functional and behavioral modeling.

3. **Structural perspective**: Approaches within the structural perspective concentrate on describing the static structure of a system. The main construct of such languages is the “entity.” Data modeling with ER modeling as a standard example can also be traced back to the seventies, but later also other approaches like semantic nets, conceptual graphs, and ontologies can be said to have structural aspects, although ontologies also have powerful means for the representation of rules.

4. **Goal and Rule perspective**: Goal-oriented modeling focuses on “goals” and “rules.” A rule is something that influences the actions of a set of actors. In the early nineties, one started to model so-called rule and goal hierarchies, linking rules and goals of different abstraction levels. This is often combined with actor-oriented modeling (see below).

5. **Object-Oriented perspective**: The basic phenomena of object oriented modeling languages are similar to those found in most object oriented programming languages; “Objects” as a specific type of entity with unique id and a local state that can only be manipulated by calling methods of
the object. Objects have a life cycle. The process of the object is the trace of the events during the
existence of the object. A set of objects that share the same definitions of attributes and operations
compose an object class. Most object-oriented modeling today is done using UML or extensions
to UML.

6. **Actor and Role perspective:** The main phenomena of languages within this perspective are “ac-
tor” (also termed agent) and “role.” The background for modeling in this perspective comes both
from organizational science, work on programming languages, and work on intelligent agents in
artificial intelligence.

7. **Communication perspective:** The work within this perspective is based on language/action theory
from philosophical linguistics. The basic assumption of language/action theory is that persons
cooperate within work processes through their conversations and through mutual commitments
taken within them. Popular in the nineties in particular, but less work is done within this perspec-
tive today.

8. **Topological perspective:** This perspective relates to the topological ordering between the different
concepts. The best background for conceptualization of these aspects comes from the cartography
and CSCW fields, differentiating between space and place. “Space” describes geometrical ar-
rangements that might structure, constrain, and enable certain forms of movement and interaction;
“place” denotes the ways in which settings acquire recognizable and persistent social meaning
through interaction.

As we see, there are a number of different approaches to conceptual modeling, each emphasizing
different aspects of the perceived reality. Towards the end of the eighties and early nineties, several re-
searchers claimed that one perspective is better, or more natural, than others. Later one has realized that
all perspectives might be useful in different settings supporting different goal of modeling. Modeling is
usually done in some organizational setting and one can look upon an organization and its information
system abstractly to be in a state (the current state, often represented as a descriptive “as-is” model)
that are to be evolved to some future wanted state (often represented as a prescriptive “to be” model).
Obviously, changes will happen in an organization independent of what is actually planned, thus one
might in practice have the use for many different models and scenarios of possible future states, but we
simplify the number of possible future states in the discussion below.

The state includes the existing processes, organization and computer systems. These states are often
modeled, and the state of the organization is perceived (differently) by different persons through these
models. Different usage areas of conceptual models are:

1. **Human sense-making:** The descriptive model of the current state can be useful for people to make
sense of and learn about the current perceived situation.

2. **Communication between people in the organization:** Models can have an important role in hu-
man communication. Thus, in addition to support the sense-making process for the individual, a
model can act as a common framework supporting communication between people both relative
to descriptive and prescriptive models.

3. **Computer-assisted analysis:** This is used to gain knowledge about the organization through
simulation or deduction, often by comparing a model of the current state and a model of a future,
potentially better state.
4. **Quality assurance:** Ensuring that the organization acts according to a certified process developed for instance as part of an ISO-certification process.

5. **Model deployment and activation:** To integrate the model of the future state in an information system directly, making the prescriptive model the descriptive model. Models can be activated in three ways:
   a. Through people, where the system offers no active support.
   b. Automatically, where the system plays an active role, as in most automated workflow systems.
   c. Interactively, where the computer and the users co-operate.

6. To be a prescriptive model to be used in a traditional system development project, without being directly activated.

7. Achieve acceptance of new solutions (either through model deployment or through models being the basis for traditional model deployment) due to the model acting as a common ground.

In the material of the book, we find examples of the six first modeling perspectives, some papers with approaches combining different perspectives, and we have structured this book roughly according to these perspectives, ending with some more general methodologically oriented papers.

**PROCESS MODELING**

In its modern form, process modeling combines functional and behavioral aspects. Despite the increasing maturity of process management technology, not all business processes are adequately supported by BPM technology and workflow systems. In “Object-Aware Business Processes: Fundamental Requirements and their Support in Existing Approaches,” Künzle, Weber, and Reichert investigate support for unstructured and knowledge-intensive processes. Support for such processes is often missing in traditional approaches and tools, especially since they cannot be straightjacketed into predefined activities. A common characteristic of these processes is the role of business objects and data as drivers for process modeling and enactment. This chapter elicits fundamental requirements for effectively supporting such object-aware processes, i.e., how to perform their modeling, execution, and monitoring. Imperative, declarative, and data-driven process support approaches are evaluated and how well they support object-aware processes is investigated. A tight integration of process and data as major steps towards further maturation of process management technology is considered.

When process models support a more well-defined process, there is potential for reuse of process knowledge between systems. Belhajjame and Brambilla investigate in the article “Ontological Description and Similarity-Based Discovery of Business Process Models” support for reuse of process models. Just like any other artifact, business process models can be stored in repositories to be shared and used by third parties, e.g., as building blocks for constructing new business processes. The success of such an approach depends partly on the availability of effective search tools to locate business process models that are relevant to the user purposes. A handful of researchers have investigated the problem of business process discovery using as input syntactical and structural information that describes business process models. This work explores an additional source of information encoded in the form of annotations that semantically describe business processes. Business processes can be semantically described using the so called abstract business processes. These are designated by concepts from an ontology which additionally captures the relationships between concepts, and thus also potential relationships between process models.
This ontology can be built in an automatic fashion from a collection of (concrete) business processes, and this work illustrates how it can be refined by domain experts and used in the discovery of business processes, with the purpose of reuse and increase in development productivity.

Process modeling and workflow is utilized in many areas. In “Towards Next Generation Provenance Systems for E-Science,” Khan, Hussain, Janciak, and Brezany investigate so called scientific workflow for e-Science. e-Science helps scientists to automate scientific discovery processes and experiments, and promote collaboration across organizational boundaries and disciplines. These experiments involve data discovery, knowledge discovery, integration, linking, and analysis through different software tools and activities. Scientific workflow is one technique through which such activities and processes can be interlinked, automated, repeated, and ultimately shared amongst the collaborating scientists. Workflows are realized by the workflow enactment engine, which interprets the process definition and interacts with the workflow participants. Since workflows are typically executed on a shared and distributed infrastructure, the information on the workflow activities, data processed, and results generated (also known as provenance), needs to be recorded in order to be reproduced and reused. A range of solutions and techniques have been suggested for the provenance of data collection and analysis; however, these are predominantly workflow enactment engine and domain dependent. This chapter includes a taxonomy of existing provenance techniques and a novel solution named VePS (The Vienna e-Science Provenance System) for e-Science provenance collection.

**STRUCTURAL MODELING AND SEMANTICS**

Structural modeling is often used for representing the data and information in an information system. Information overload is perceived as a common problem in organizations and enterprises, which calls for new organizational and technological approaches for more pertinent and accurate information supply. In “Modeling Information Demand in an Enterprise Context: Method, Notation, and Lessons Learned” by Lundqvist, Sandkuhl, and Seigerroth, the authors address the problem of information overload by proposing a method for information demand modeling, which contributes to capturing and understanding the information demand of roles in organizations. This method consists to a large extent of an application of enterprise modeling techniques. Illustrated by a case from the automotive industry, lessons learned from information demand modeling are presented and discussed. This includes the specific perspective taken in the method for information demand analysis, common challenges experienced in demand modeling, and the validity of recommendations from using participative enterprise modeling for information demand modeling. Furthermore, the paper introduces the notation applied for information demand models and discusses refinement process of this notation.

Recently, Semantic Web technologies, such as ontologies, have been proposed as key enablers for integrating heterogeneous data schemas across business and governmental systems. Algorithms designed to align different, but related ontologies have become necessary as differing ontologies proliferate. The paper “Ontology Alignment Quality: A Framework and Tool for Validation” by Sampson, Krogstie, and Veres looks into the need for ontology alignment in this landscape. The process of ontology alignment seeks to find corresponding entities in a second ontology with the same or the closest meaning for each entity in a single ontology. This research is motivated by the need to provide tools and techniques to support the task of validating ontology alignment statements, since it cannot be guaranteed that the results from automated tools are accurate. The authors present a framework for understanding ontology
alignment quality and describe how AlViz, a tool for visual ontology alignment, can be used to improve the quality of alignments. An experiment was undertaken to test the claim that AlViz supports the task of validating ontology alignments. A promising result found that the tool has potential for identifying missing alignments and for rejecting false alignments.

Web services as a new distributed system technology have been widely adopted by different fields in the information systems areas, such as Enterprise Application Integration (EAI), Business Process Management (BPM), and Virtual Organization (VO). However, lack of possibility to represent semantics in the current Web service standards has become a major barrier to provide service discovery and service composition. To tackle the semantic issues of Web services the paper “CbSSDF: A Two-Layer Conceptual Graph Approach to Web Services Description and Composition – A Scenario-Based Solution Analysis and Comparison with OWL-S” by Du, Munro, and Song proposes a comprehensive semantic service description framework—CbSSDF and a two-step service discovery mechanism based on CbSSDF—to help service users to easily locate result when using the approach to retrieve the required services. The authors give a detailed explanation of CbSSDF, and then evaluate the framework by comparing it with OWL-S to examine how the proposed framework can improve the efficiency and effectiveness of service discovery and composition. The evaluation is carried out by analyzing the different proposed solutions based on these two frameworks for achieving a series of tasks following a defined scenario.

Most information systems development methodologies are based on conceptual modeling of static and dynamic views, which are represented by totally different types of diagrams. In “Modeling Approach for Integration and Evolution of Information System Conceptualizations,” Gustas points to the need for understanding of the interplay among interactive, behavioral, and structural aspects of specifications to identify semantic integrity problems between business process and business data. Typically, semantic inconsistencies and discontinuities between collections of conceptual representations are not easy to detect and to comprehend for information system designers due to static and dynamic aspects of models being visualized in isolation. The goal of his paper is to present a modeling approach for semantic integration and evolution of static and dynamic aspects of conceptual models. Visualization of interplay among structural, interactive, and behavioral aspects of computation-neutral representations helps to understand crosscutting concerns and integrity problems of information system conceptualizations. The main advantage of the presented conceptual modeling approach is stability and flexibility of diagrams in dealing with the evolutionary changes of requirements. Therefore, the developed modeling foundation is targeted to both business managers and information system designers for the purpose of computation-neutral integration and evolution of information systems specifications.

OBJECT-ORIENTED MODELING

Object-oriented modeling can be said to be another way of combining behavioral, functional, and structural aspects. Although the field of object-oriented modeling has spurred a lot of approaches since it appeared in the late eighties, over the last 10-15 years most work has centered on UML, or extensions to UML. In “Ontological Rules for UML-Based Conceptual Modeling: Design Considerations and a Prototype Implementation” by Lu and Parsons, they look upon how to support different goals of modeling with UML. UML is used both as a language for object-oriented software design, and as a language for conceptual modeling of different applications domains. Given the differences between these goals, UML’s origins in software engineering might limit its appropriateness for conceptual modeling. In this
context, Evermann and Wand have proposed a set of well-defined ontological rules to constrain the part of
the language to be used in construction of UML diagrams to reflect underlying ontological assumptions
about the real world. The authors extend their work using a design research approach that examines these
rules by studying the consequences of integrating them into a UML CASE tool. The paper demonstrates
how design insights from incorporating theory-based modeling rules in a software artifact can be used
to shed light on the rules themselves. In particular, the authors distinguish four categories of rules for
implementation purposes, reflecting the relative importance of different rules to achieve the underlying
goal of the modeling and the degree of flexibility available in enforcing them. They propose distinct
implementation strategies that correspond to these four rule categories and identify some redundant
rules as well as some rules that cannot be implemented without changing the UML specification. The
rules are implemented in an open-source UML CASE tool.

In object-oriented design, excessive coupling between object-oriented classes is widely acknowledged
as a maintenance problem that can result in a higher propensity for faults in systems and a ‘stored up’
future problem. In “A Longitudinal Study of Fan-In and Fan-Out Coupling in Open-Source Systems,”
Mubarak, Counsell, and Hierons explore the relationship between “fan-in” and “fan-out” coupling met-
rics over multiple versions of open-source software. More specifically, the relationship between the two
metrics is explored to determine patterns of growth in each over the course of time. The JHawk tool was
used to extract the two metrics from five open-source systems. Results show a wide range of traits in
the classes to explain both high and low levels of fan-in and fan-out. Evidence was also found of certain
“key” classes (with both high fan-in and fan-out) and “client” and “server”-type classes with high fan-
out and fan-in, respectively. This chapter provides an explanation of the composition and existence of
such classes as well as for disproportionate increases in each of the two metrics over time. Finally, it was
found that high fan-in class values tended to be associated with small classes; classes with high fan-out
on the other hand tended to be relatively large classes.

GOAL AND ACTOR MODELING

While traditional approaches in business process modeling tend to focus on “how” the business processes
are performed (adopting a behavioral description in which business processes are described in terms of
procedural aspects), in goal-oriented business process modeling, the proposals strive to extend tradi-
tional business process methodologies by providing a dimension of intentionality to business processes.
One of the key difficulties in enabling one to model goal-oriented processes concerns the identification
or elicitation and structuring of the goals themselves. In “A Method for Eliciting Goals for Business
Process Models based on Non-Functional Requirements Catalogues,” Cardoso, Almeida, Guizzardi,
and Guizzardi presents a case study conducted in a Brazilian hospital following such an approach. The
case obtained several goal models represented in i*/Tropos, a well-known example of an approach to
Goal-Oriented Requirements Engineering (GORE), each of which correspond to a business process also
modeled in the scope of the study. Existing catalogues of Non-Functional Requirements (NFR) were
helpful in goal elicitation, uncovering goals that did not come up during previous interviews prior to
these catalogues’ use.

In “An Approach for E-Service Design Using Enterprise Models” by Henkel, Johannesson, and Per-
jons, the authors look upon value creation and innovation that require collaboration with customers and
vendors in agile and flexible networks of actors. To realize such networks, organizations are embrac-
ing service oriented models and architectures using e-services for business communication. A major issue for a service-oriented organization is to design and offer e-services that are adapted to the needs, wants, and requirements of different actors such as customers and vendors. This is a challenging task as different customer groups and vendors will have different requirements, which may vary over time, resulting in a large number of e-services. In this chapter, the authors suggest using enterprise models as being adequate instruments for design and maintenance of e-services. More specifically, an approach for designing e-services based on value and goal models, which will ensure that the constructed e-services will satisfy the needs and wants of customers is presented. A project from the Swedish health care sector is used to demonstrate and evaluate the proposed approach.

**METHODOLOGY AND TOOLS FOR ANALYSIS AND DESIGN**

Development and evolution of information systems has gone from a focus on structure (e.g. in projects following a waterfall model) and in-house development, to the use of more agile approach utilizing external resources such as packaged and open source systems. One important type of reuse is the use of packaged systems such as ERP-systems. Empirical studies on requirements engineering for inter-organizational Enterprise Resource Planning (ERP) systems have demonstrated that the ERP vendor-provided prescriptive models for ERP roll-outs make tacit assumptions about the ERP adopter’s context, e.g. on how they are organized. This, in turn, leads to the implementation of suboptimal solutions when these presumptions do not fit the current organization. Specifically, these models assume that ERP implementations happen within a single company, and so they pay only scant attention to the stakeholders requirements for inter-organizational coordination. Given this backdrop in the article “What Practitioners Think of Inter-Organizational ERP Requirements Engineering Practices: Focus Group Results” Daneva and Ahituv look in more detail on 13 practices for engineering the ERP coordination requirements proposed by Daneva in previous publications. This chapter reports a confirmatory study evaluating the importance of those practices. Using an online focus group, the authors collected and analyzed practitioners’ feedback and their experiences to understand the extent to which the proposed practices are indeed observable. The study indicated very low variability in practitioners’ perceptions regarding 12 of the 13 practices, and considerable variability in their perceptions regarding the role of modeling inter-organizational coordination requirements. The contribution of the study is twofold: (1) it adds to the body of knowledge in the sub-area of RE for ERP; and (2) it adds to the practice of using qualitative research methods in empirical RE.

Even if the current approaches to information systems development are more agile, they still have to relate to external legislative and organizational frameworks. In “The Impact of Regulatory Compliance on Agile Software Processes with a Focus on the FDA Guidelines for Medical Device Software” by Mehrfard and Abdelwahab Hamou-Lhadj, the authors address the difficulty of complying with different regulations. This has become more evident as a large number of regulated businesses are mandated to follow an ever-increasing set of standards and regulations. These regulations often drive significant changes in the way organizations operate to deliver value to their customers. This chapter focuses on the impact of the Food and Drug Administration (FDA) regulations on agile software development processes, which in many ways can be considered as just another type of organizational processes. Particular focus is placed on the ability for Extreme Programming (XP) to support FDA requirements. Findings show that XP fails to meet many of the FDA guidelines for medical device software, which increases the risks
of non-compliance for organizations that have adopted XP as their main software process. The results of this study can lead the work towards designing an extension to XP for FDA regulation, and might also carry over to support ways of dealing with external regulations in other fields in an agile approach.

One approach for using external resources is the use of Open Source Software (OSS). OSS has reached new levels of sophistication and acceptance by users and commercial software vendors. In “Predicting OSS Development Success: A Data Mining Approach” the authors Raja and Tretter create tests and validate a model for predicting successful development of OSS projects. Widely available archival data was used for OSS projects from Sourceforge.net (SF). The data is analyzed with multiple Data Mining techniques. Initially three competing models are created using Logistic Regression, Decision Trees, and Neural Networks. These models are compared for precision and are refined in several phases. Text Mining is used to create new variables that improve the predictive power of the models. The final model is chosen based on best fit to separate training and validation data sets and the ability to explain the relationship among variables. Model robustness is determined by testing it on a new dataset extracted from the SF repository. The results indicate that end-user involvement, project age, functionality, usage, project management techniques, project type and team communication methods have a significant impact on the development of OSS projects.

Methodologies for information systems development needs to be adaptable. Method Engineering (ME) is a discipline that aims to bring effective solutions to the construction, improvement, and modification of the methods used to develop Information Systems (IS). Situational Method Engineering (SME) promotes the idea of retrieving, adapting, and tailoring methodology components, rather than complete methodologies, to the specific context. Existing SME approaches use the notion of context for characterizing situations of IS development projects and for guiding the method components selection from a repository. However, in the reviewed literature, there is no proposed approach to specify the specific context of method components. In “Towards Method Component Contextualization,” Kornyshova, De- neckère, and Claudepierre provide a detailed vision of context and a process for contextualizing methods in the IS domain. This proposal is illustrated with three case studies: scenario conceptualization, project portfolio management, and decision-making.

CONCLUDING REMARKS

One can argue that the main reason why humans have excelled as species is our ability to represent, reuse, and transfer knowledge across time and space. Whereas in most areas of human conduct, one-dimensional natural language is used to express and share knowledge, we see the need for and use of two and many-dimensional representational forms to be on the rise. One such representational form is called (conceptual) modelling. A conceptual model is traditionally defined as a description of the phenomena in a domain at some level of abstraction, which is expressed in a semi-formal or formal diagrammatical language. Modeling is an important part of both information systems development and evolution, and organizational development in general (e.g. used in enterprise modeling/enterprise architecture). As illustrated in this book, the field of information systems modeling and design includes numerous modeling methods and notations that are typically evolving. Even with some attempts to standardize (e.g. UML for object-oriented software design), new modeling methods are constantly being introduced, many of which differ only marginally from previous approaches. These ongoing changes significantly impact the way information systems, enterprises, and business processes are being analyzed and designed in practice.
Whereas modeling techniques traditionally was used to create intermediate artifacts in systems analysis and design, more and more modeling methodologies take a more active approach to the exploitation of this particular form of knowledge representation. In approaches such as Business Process Management (BPM), Model Driven Architecture (MDA), and Domain Specific Modeling/Domain Specific Modeling Languages (DSM/DSL), Enterprise Architecture (EA), and Active Knowledge Modeling (AKM), the models are used directly to form the information system of the organization. At the same time, similar techniques are used also for sense-making and communication, model simulation, quality assurance and requirements specification in connection to more traditional forms of information systems development as illustrated with the overview of goals of modeling in the introduction.

Given that modeling techniques are used in such a large variety of tasks with very different goals, it is important for appropriate use of the techniques to have a proper overview of different use of modeling, and guidelines for what make a model sufficiently good to achieve the decided goals. An important direction for the field is to extend the work on quality of models and modeling languages in this setting, to be able to evaluate and enhance current information modeling methods and methodologies. Evaluation of modeling approaches remains a challenge in the information systems field. Although work has been done in this direction by many researchers over a long period, it is a need to get the results from this research into practical use.

John Krogstie
Norwegian University of Science and Technology, Norway

REFERENCES