A new paradigm for Continuous Alignment of Business and IT: Combining Enterprise Architecture Modeling and Enterprise Ontology

Knut Hinkelmann, School of Business, FHNW University of Applied Sciences and Arts Northwestern Switzerland, 4600 Olten, Switzerland and Department of Informatics, University of Pretoria, South Africa (e-mail: knut.hinkelmann@fhnw.ch)

Dimitris Karagiannis, Knowledge Engineering Research Group, University of Vienna, Austria (e-mail: dk@dke.univie.ac.at)

Barbara Thoenssen, School of Business, FHNW University of Applied Sciences and Arts Northwestern Switzerland, 4600 Olten, Switzerland (e-mail: barbara.thoenssen@fhnw.ch)

Robert Woitsch, BOC Asset Management GmbH, Vienna, Austria (e-mail: robert.woitsch@boc-eu.com)

Aurona Gerber, CSIR Meraka, Pretoria, South Africa and Department of Informatics, University of Pretoria, South Africa (e-mail: AGerber@csir.co.za)

Alta van der Merwe, Department of Informatics, University of Pretoria, South Africa (e-mail: alta.vdm@up.ac.za)

A new paradigm for Continuous Alignment of Business and IT: Combining Enterprise Architecture Modeling and Enterprise Ontology

Abstract— The ability to react timeously to continuous and unexpected change is called agility and is an essential quality of the modern enterprise. Being agile has consequences for the engineering of enterprises and enterprise information systems. We propose a new paradigm for next generation enterprise information systems, which shifts the development approach of model-driven engineering to continuous adaptation of the agile enterprise. We propose a metamodeling approach, which supports both human-interpretable representations, i.e. graphical models, and machine-interpretable representations, namely enterprise ontologies. Furthermore, we describe next generation enterprise information systems, which embed modeling tools and algorithms for model analysis.

1 Introduction

In this paper we deal with Next Generation Enterprise Information Systems in the context of Enterprise Engineering (EE). Giachetti [1] defines Enterprise Engineering as "the body of knowledge principles and practices to design an enterprise" where an enterprise is a "complex socio-technical system that comprises interdependent resources of people, information, and technology that must interact with each other and their environment in support of a common mission".

The ability of keeping up with continuous and unexpected change is an essential quality of modern enterprises and will become a necessity for existence. Dove [2] calls this characteristic *agility* and defines it as "the ability of an organization to thrive in a continuously changing, unpredictable business environment." The concept of the agile enterprise emerged in the early 1990s [3]. Such an agile enterprise rapidly adapts to changing business challenges and opportunities and it continuously improves to optimize costs, quality and speed of delivery. It enables top management to quickly implement new strategies and control key business parameters to gain a competitive advantage [4], which means that enterprise engineering is an ongoing activity. An enterprise is not designed just once, but an enterprise is, to varying degrees, redesigned many times [1]. Implemented business processes and information systems have to be continuously adapted. As changes may be triggered from the business as well as from developments in the technology, a *continuous alignment of business and IT* is needed.

The pace of change is continuously accelerating and managing the change is increasingly beyond the control of companies. The rate of technological progress increased throughout history. For example, in the car industry new models are developed within few months instead of years. In the banking industry, the time to market for new financial products is a few weeks instead of months [5]. Each new product or service requires new or adapted processes and information systems to produce the products and to deliver the services. Reduced time to market increases the demand for changes of business processes and information systems. Considering the multiyear nature of many enterprise engineering initiatives, the architecture at the start of a development might not be appropriate anymore when the new business processes and information systems are rolled out.

The grand challenge for today's enterprises, which we deal with in this research, is the *continuous alignment of business and IT in a rapidly changing environment*. According to Gartner [6] enterprises are facing a new era of enterprise IT, the 'digitalization' era, "a period characterized by deep innovation beyond process optimization, exploitation of a broader universe of digital technology and information, more-integrated business and IT innovation, and a need for much faster and more agile capability".

In order to deal with this grand challenge we propose an approach using model-based engineering that is visualized in Figure 1.1. The approach builds on the principles of model-driven enterprise engineering [7] and is supplemented with two innovative and challenging developments:
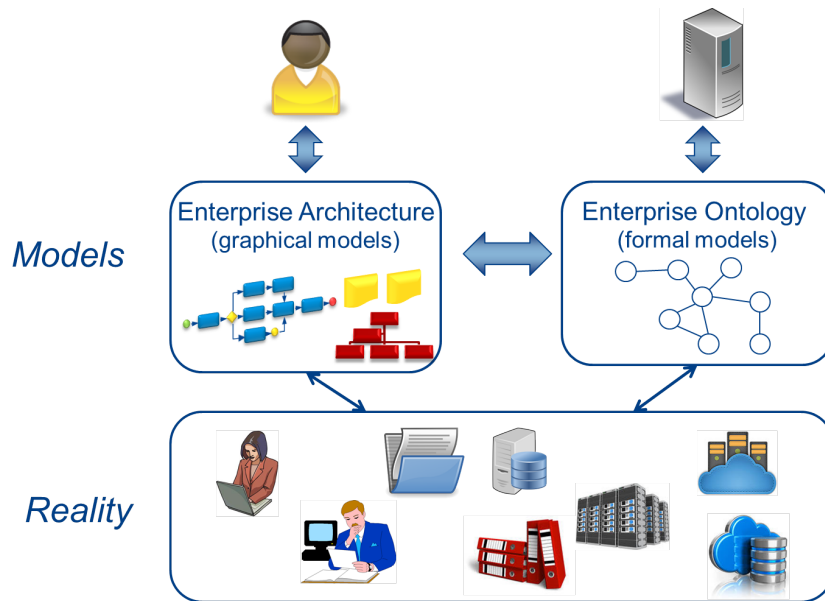
Figure 1.1: The modeling approach for continuous business-IT alignment

- Shift the paradigm of model-driven engineering from *development* to *continuous adaptation*. In contrast to software development it is unusual for enterprise engineering to follow a Greenfield approach and start from scratch. Instead, we typically adapt a 'running' enterprise. The challenge is to react on change in the business (e.g. due to an altered business strategy) and IT (e.g. due to innovative technology) alike and to continuously keep business and IT aligned. Models are used for designing and adapting enterprises and enterprise information systems before they are changed in reality.
- Support machine interpretable and human interpretable models: McCauley [8] defines an agile organization as "one that can sense opportunity or threat, prioritize its potential responses and act efficiently and effectively". In order to support in sensing, prioritizing and acting, the models should not only be passive storage of knowledge intended for human use but model processing in this context also demands automated operations on models that retrieve and interpret information for decision making. The focusing on machine interpretable knowledge is called knowledge engineering (KE) [9] and is distinguished from knowledge management (KM), which is focusing on human interpretable knowledge. The challenge is to keep both representations consistent.

To meet these challenges we propose a metamodel approach for next generation information systems, which builds on the knowledge engineering for business process management presented in [9]. These are the main characteristic of our approach:

- Graphical notations are provided, which can easily be understood by humans.
- Semantic lifting makes the semantics of metamodels explicit [10], [11] such that the analysis, adaptation and evaluation of models can be done by a machine. We propose an ontology to specify the semantics of the metamodel.

In the next section we provide some background information and State of the Art as well as more detail on metamodeling. We discuss solutions that are already available to realize our proposed approach for the next generation enterprise information systems. We also highlight challenges that still need to be solved in order to fully realize this approach. In chapter 3 we explain elements of modeling methods. Then we present our modeling method for continuous business-IT alignment in chapter 4. Finally in chapter 5 we summarize the contribution and give an outlook on future work.

2 Background

In this section we provide background information in relevant topics for continuous alignment of business and IT. We first discuss different types of agility followed by an explanation of business-IT alignment in the context of an agile enterprise. Lastly we provide background on enterprise modeling and show how it supports the alignment of business and IT.

2.1 Enterprise Agility

Cummins [4] divides agility into four dimensions: dynamism, adaptability, flexibility and awareness.

- *Dynamism* is defined as the ability to change the process definition of an enterprise. The need to change a process definition may result from process improvements to process innovation or process reengineering.
- *Adaptability* is the ability of an enterprise to react to exceptional circumstances or unexpected events during the performance of a process instance, which may or may not be foreseen.
- *Flexibility* is the ability to deal with a fair degree of uncertainty.
- *Awareness* is the ability to detect opportunities and risks.

Reichert and Weber [12] also distinguish between four types of agility[1] needs in Process Aware Information Systems (PAIS) namely variability, looseness, evolution and adaptation. *Evolution* represents the ability of the process implementations to change. Since business processes can evolve over time, it is not sufficient to implement them once and then to never touch the PAIS again. Evolution is equivalent to the dynamism in [4] and in this paper the focus is on evolution/dynamism as well as awareness.

2.2 Complexity and Change

In order to identify the need for changes, an organization has to continuously monitor itself and be prepared to react quickly to threads and opportunities. However, the challenge to react quickly is increased by the complexity of today's IT. According to Dietz [13] the most dominant problem identified in scientific as well as in popular science on enterprise management, is complexity and how it can be managed. He claims that because of the complexity of enterprises a conceptual model is needed that "only shows the essence of the operation of an enterprise" and therefore "the model abstracts from all realization and implementation" [13]. Hence, Chen et al. [14] consider enterprise architecture as the foundation of enterprise systems engineering with the goal to support stakeholders of an enterprise to manage system engineering and changes. Zachman regards enterprise architecture as the determinant of survival in the Information Age in order to deal with increased complexity and change of enterprises [15].

2.3 Enterprise Architecture (EA)

There are various definitions of enterprise architecture (EA). A definition that is in line with the ISO/IEC/IEEE 42010 standard [16] defines an enterprise architecture as "fundamental concepts or properties of an enterprise in its environment embodied in its elements, relationships, and in the principles of its design and evolution." An enterprise architecture is typically described using models. An architecture description is a work product used to express architecture. The description of the enterprise architecture is a helpful and necessary tool to understand complexity and manage change [17].

Due to the complexity of an enterprise architecture description, many frameworks were developed to assist in this task. A framework is a logical structure for classifying and organizing complex information. An architecture framework is defined by the ISO/IEC/IEEE 42010 standard as "conventions, principles and practices for the description of architectures established within a specific domain of application and/or community of stakeholders" [16].

There is a huge variety of EA frameworks (EAF). Matthes [18] points out that to date more than 50 frameworks are available. In his compendium Matthes gives a detailed description of 34 EA frameworks, based on clearly structured and well defined criteria. Here we briefly mention two EA frameworks, which are widely used. The purpose is to show that although the content is comparable the structure of the frameworks can differ.

The Zachman framework is of particular interest because according to [18] it is widespread with an approximate market share amounts between 22% and 25% and builds the basis for many other frameworks. The Zachman Framework is a two dimensional matrix [19]. Rows depict different perspectives of the role a stakeholder may take (named planner, owner, designer, builder and subcontractor), and columns represent the various aspects that should be considered. They are "different abstractions from or different ways to describe the real world" [20 p. 592]. The aspects (rows) are named based on the fundamentals of communication. The interrogatives What (data), How (function), When (time), Who (people), Where (network), and Why (motivation) build the basis for the concise description of complex ideas [19].

TOGAF is another well-known EA framework [21]. The overall enterprise architecture as composed of a set of closely inter-related architectures: Business Architecture, Information Systems Architecture (comprising Data Architecture and Application Architecture), and Technology (IT) Architecture [22].

2.4 Enterprise Architecture Descriptions

Zachman gives no advice on how the enterprise architecture description should look: intersections of

---

[1] Reichert and Weber call it "flexibility" but we use the term "agility" in order to be consistent in naming.

perspectives and aspects can be represented in models of various model types, like a data model or a process model. Those model types can in turn be represented in various languages. OMG has developed several specialized modeling languages for enterprise architecture modeling, for example Business Process Model and Notation (BPMN) [23], Case Management Model and Notation (CMMN) [24], and the Business Motivation Model (BMM) [25]. The purpose of these graphical modeling languages is to support communication between human stakeholders. They are not intended for machine interpretation - although there does exist execution engines for BPMN.

The ArchiMate Standard [22] introduces an integrated language for describing enterprise architectures. ArchiMate fits into the TOGAF framework as it provides concepts for creating a model that correlates to its three architectures (layers). According to [26] ArchiMate can be used to describe all aspects of the EA in a coherent way, while tailoring the content for a specific audience.

ArchiMate provides a graphical representation of its language elements based on UML class diagram but customized and limited to a small set of modeling constructs in the interest of simplicity of learning and use. The standard claims that architecture descriptions "are formal descriptions of an information system, organized in a way that supports reasoning about the structural and behavioural properties of the system and its evolution." [22][27]. However, the ArchiMate language has one shortcoming: it is intended for human interpretation and not suitable for *automatic* reasoning for two reasons. It is too coarse grained as it only contains basic concepts and relationships that serve general enterprise architecture modeling purposes [28].

2.5 The Enterprise Engineering Knowledge Space

Models are representing part of reality or a vision in an agreed modeling language. The term "knowledge space" is used to name what is represented in a model. The actual knowledge space represented in models is specified according to the four dimensions *form, content, interpretation*, and *use* (see **Error! Reference source not found.**).
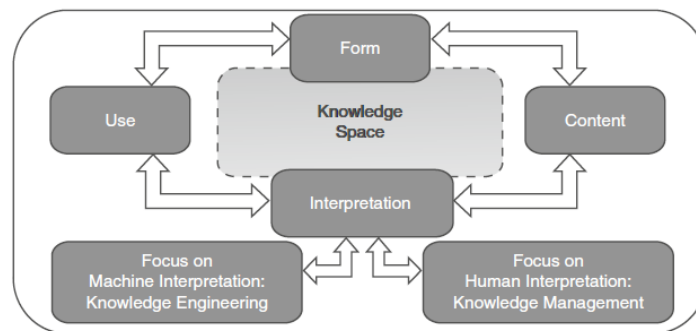

Figure 2.1: The four dimension of a knowledge space [9]

- The *form* represents the syntax and semantic.
- The *content* is seen as the domain in which knowledge engineering is applied. In the area of Enterprise Engineering the content comprises the enterprise architecture, which can be the as-is architecture or a planned to-be architecture. The model is then the enterprise architecture description.
- Depending on the intended *use* only a subset of the knowledge space's content might be of interest. Views and viewpoints are a means to specify which part of an architecture description is of relevance for one or more stakeholders to address specific concerns.
- The representation of knowledge is either focused on machine *interpretation* or on human *interpretation*. In this context *enterprise architecture* is typically represented by the means of graphical models which typically are cognitively more adequate for *human interpretation* and *enterprise ontologies*, on the other hand, are formal representations which can be *interpreted by machines*.

In the following sections we refer to the *human-interpretable, graphical* modeling as *enterprise architecture* and to *machine-interpretable, formal representations* as *enterprise ontology*. We propose a *modeling method* for Model-Driven Enterprise Engineering that allows for describing the knowledge space in both a human-interpretable form (enterprise architecture models) and machine-interpretable form (enterprise ontology).

2.6 Enterprise Ontology (EO)

Because of the complexity of enterprise architecture we consider machine intelligibility of enterprise architecture descriptions essential for agile enterprises. A machine-understandable and interpretable architecture

description would allow to answer questions like "which processes are affected by the replacement of an application?", "which roles are involved in the process?", "why did we decide to customize this specific application?"

As shown by [29] and [30] an enterprise ontology (EO) could meet this request. Describing enterprise architecture as an ontology started in the 1990s with TOVE [31], The Edinburgh Enterprise Ontology [32] and the organizational memory [33]. In contrary to EA enterprise ontologies are concrete representations of (generalized) enterprise architectures developed to be re-used in enterprises [34], adopted and enhanced to an enterprise's specific needs. Den Haan [35] has used an enterprise ontology to realize a Model-Driven Enterprise Engineering. ArchiMEO is an example of an enterprise ontology based on the ArchiMate standard. It contains the concepts of ArchiMate 2.0 and extends them by more generic concepts to express more specific elements, for example activities or types of business actors.

The advantage of having an ontological representation of an enterprise architecture that is machine understandable and hence allows for automation was proved in two research projects. One building an early warning system for risks in the supply chain [36]; the other linking enterprise architecture description with operational databases to provide an integrated view and management of enterprise entities spread over various data stores, represented in different ways and levels of granularity [37].

There are a variety of representation formalisms for ontologies which allow for machine interpretations. Recent approaches like RDFS and OWL were developed in the context of the semantic web [38]. There is no 'right' language to formally describe an enterprise ontology. The "choice of the language to use in a system or analysis will ultimately depend on what types of facts and conclusions are most important for the application" [39]. It is not the purpose of this paper to propose the appropriate ontology representation formalism. This is left to future research.

3 A Model-Based Approach for Enterprise Engineering

In the previous chapter we provided the background of human-interpretable and machine-interpretable enterprise modeling. In this section we provide the basis for integrating these two modeling approaches. According to [41] a modeling method consists of a modeling technique, which is further divided into a modeling language and a modeling procedure, as well as modeling mechanisms and algorithms [41]. The components of a modeling method and their relations are visualized in Figure 3.1. Each of the main components (modeling language, modeling procedure and mechanism and algorithms) is accented with a different color. In this section we explain each of the three main constituents in the context of enterprise engineering.
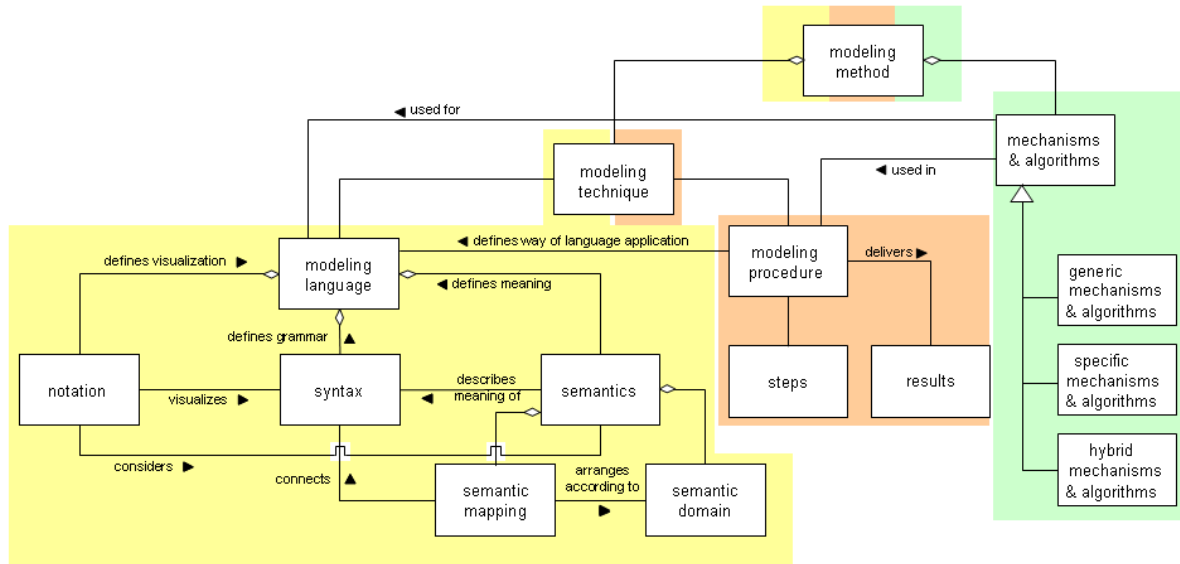

Figure 3.1: Components of modeling methods [41]

3.1 Modeling language

A modeling language is defined by *syntax, semantics*, and *notation* that provide the necessary modeling primitives in order to build the model. The concepts that describe the modeling language are defined in the metamodel language, which itself has to be represented in a modeling language. Figure 3.2 shows a layered model stack, which was proposed by Strahringer [42] and adapted by Karagiannis (e.g. [9], [41]). The stack could be extended indefinitely but typically 4 layers are sufficient.
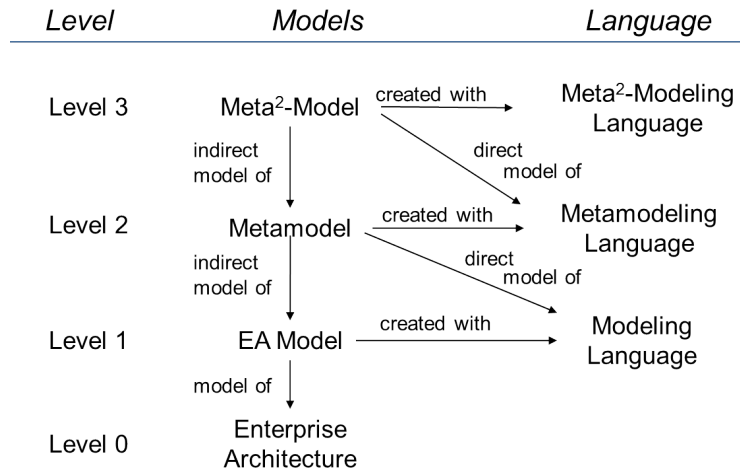
Level	Models	Language

Level 3 — Meta²-Model — created with → Meta²-Modeling Language

indirect model of ↓ — direct model of

Level 2 — Metamodel — created with → Metamodeling Language

indirect model of ↓ — direct model of

Level 1 — EA Model — created with → Modeling Language

model of ↓

Level 0 — Enterprise Architecture

Figure 3.2: Layered model stack for Enterprise Architecture (adapted from [41])

A prominent metamodeling framework is the Meta Object Facility (MOF), an OMG standard for Model-driven Engineering [43]. MOF is based on the UML infrastructure and thus it is a candidate for object-oriented enterprise modeling. MOF metamodels for modeling languages like BPMN, CMMN or BMM are typically modeled as UML class diagrams. ArchiMate is also a modeling language created with UML as metamodeling language. MOF has also been used to define metamodels for Ontology languages like OWL and RDFS [44]. MOF expresses abstract syntax and semantics but does not support the definition of the graphical representation, i.e. the notation or concrete syntax. Thus, MOF is applicable to define metamodels for a machine-interpretable modeling language, in our scenario for the *enterprise ontology*.

ADOxx® is a meta-metamodeling framework for defining graphical modeling languages. It has been researched at the University of Vienna (see for example [45]–[47]) and implemented in the commercial tool ADONIS®. The ADOxx® meta-metamodel provides the basic metamodeling classes that are necessary to define graphical modeling languages such as class, attribute, and relation. It also introduces several concepts for the enterprise architecture modeling, such as model types, views, and predefined classes for directed graphs for business processes and nondirected graphs for organizational structure.

The ADOxx® meta-metamodeling integrates concrete syntax and abstract syntax. The definition of the classes, attributes and relations defines the semantics and the abstract syntax of the modeling language (see left part of Figure 3.3). The concrete syntax corresponds to the graphical notation for the modeling elements (see right part of Figure 3.3). Each class has an attribute *GraphRep*. The value of this attribute is a script which defines the notation. Due to the expressiveness of the GraphRep script language, ADOxx® is a good fit to define metamodels for human-interpretable *enterprise architecture* modeling.
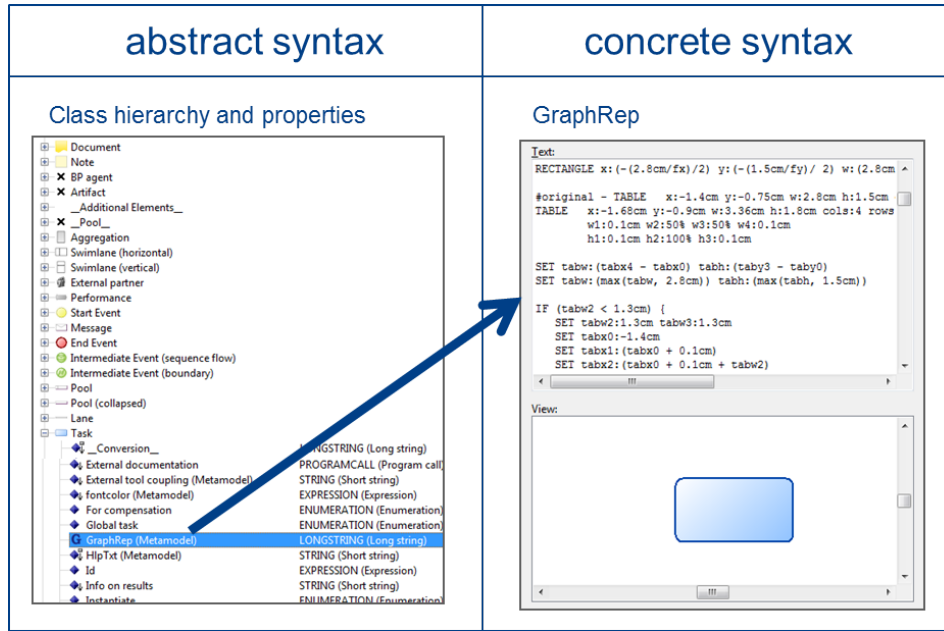
Figure 3.3: Abstract and concrete syntax of a task in ADOxx®

Hence for integrating enterprise architecture and enterprise ontology, the challenge is to integrate metamodels derived from frameworks like MOF – which are used for machine interpretable modeling languages – with metamodels derived from frameworks like ADOxx® – which are used for human interpretable graphical languages. We show some approaches in section 4.

3.2 Modeling procedures

The modeling procedures depend on the use of the knowledge represented in the models (see section 2.5). They support different tasks of enterprise engineering for example business process management, business-IT alignment, risk management, decision management, business analytics and supply chain management. ADOxx® is the metamodel framework of choice for the open models initiative (OMi, www.openmodels.at), a community of practice for the creation of modeling methods [48]. In a recent booklet of the OMi, 25 modeling methods for a variety of application domains are described [49]. In recent projects we have developed a modeling languages and procedures for strategic alignment of business and IT [50] for integrating enterprise risk management with business motivation, business processes and business rules [51].

Since modeling is a human task, it typically starts with graphical models, which are cognitively more adequate than formal methods for most stakeholders. The graphical models are used as a means for communication between the stakeholders involved in enterprise design.

A challenge is to extend these modeling procedures with phases of machine-interpretation, which has been realized for some specialized procedures only. For business process management there are procedures that include the execution of graphically generated process models. The idea of MDE is to transform models on higher abstraction levels into lower level models until the model can be made executable. In [37] we describe a procedure to generate meta data from enterprise architecture model.

Another challenge is to keep the connection between graphical and machine-interpretable models. If for example an information system, which implements a process, is changed this change should be mirrored back to the graphical model in order to keep both models consistent.

3.3 Mechanisms and Algorithms

Machine-interpretation of the models is implemented in modeling mechanisms and algorithms, which realize the model processing operations. To automate these operations, the modeling language should have a well-defined semantics and syntax. The concrete syntax is exploited for supporting the modeler in modeling design, e.g. by visualizing particular aspects of the model. Mechanisms and algorithms process the abstract syntax.

The challenges to engineer the agile enterprise demand for mechanisms and algorithms that can analyze the models in order to detect potential risks and to seize opportunities. The continuous alignment of business and IT can be supported by mechanisms, to identify information systems, which are affected by a change of the business. The other way round we might need mechanisms to identify business processes, which are affected by

modifications in the IT.

4 A Modeling Method for Continuous Business-IT Alignment

In this section we present our model-based approach for enterprise engineering. As already argued in the beginning, engineering the agile enterprise is an ongoing endeavor of design and redesign, which requires a continuous alignment of Business and IT. In the rest of this section we describe the elements of the modeling method consisting of the modeling procedure, the modeling language and the mechanisms and algorithms (see Figure 3.1)

4.1 Modeling Procedure for Continuous Business-IT Alignment

Our approach consists of four phases and is a variant of the Plan-Do-Check-Act cycle:
1. Establish/adjust goals: strategic and operative goals both for business and IT and their relations.
2. (Re-)Engineer the enterprise: modeling resp. adapting the business, application and technology architectures as well as their relationships
3. Implement the enterprise architecture and run the enterprise
4. Monitor the running of the enterprise and recognize adaptation needs

If in the monitoring phase a need for adaptation is recognized the cycle starts again. Enterprise models are particularly used for the identification of adaptation needs (Phase 4) and implementing changes (Phase 2).
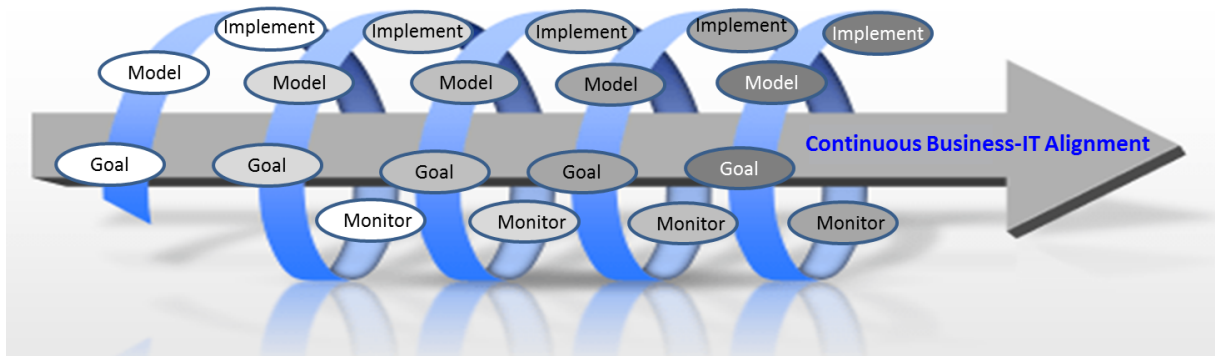

Figure 4.1 Continuous Business-IT Alignment

4.2 Metamodeling and Enterprise Ontologies

In this section we describe our modeling approach, which combines human-interpretable graphical enterprise architecture models and machine-interpretable formal models. The challenge is to keep both representations consistent.

A modeling language consists of notation, syntax and semantics [41] (see Figure 3.1). Höfferer [52] discusses the relationship between metamodels and ontologies for defining modeling languages. Metamodels and ontologies are different but complementary concepts. Ontologies basically provide the semantics of the modeling language constructs [10], [53] as well as the semantics of model instances. Metamodels provide the syntax of a modeling language; they define all available modeling constructs as well as valid ways to combine them. Some semantics is also included in language constructs. Therefore, there are two approaches to define the human-interpretable and the machine-interpretable modeling languages, including through Semantic lifting and Semantic Metamodels:

- Semantic lifting: The metamodels for the human-interpretable graphical enterprise architecture and the machine-interpretable enterprise ontology are strictly separated. Metamodels and ontologies are merged by transformation, which is called semantic lifting.
- Semantic Metamodels: The semantics of all modeling concepts is expressed by an ontology, which is extended by a metamodel to define the notation and syntax of the graphical modeling language. This has the advantage that the semantics is expressed only once.

In the following sections we give a description of these two approaches.

4.2.1 Semantic Lifting: Separating Metamodels and Ontologies

Figure 4.2 shows the conceptual architecture for semantic lifting. Different models of the enterprise architecture are created corresponding to different metamodels, which define primarily syntactical but also some semantic aspects of model elements. The ontologies define the machine-interpretable semantics

of the language concepts. Semantic lifting is closes related to semantic annotation, which combines the human-readable and machine-readable information (see [54] for a recent overview).
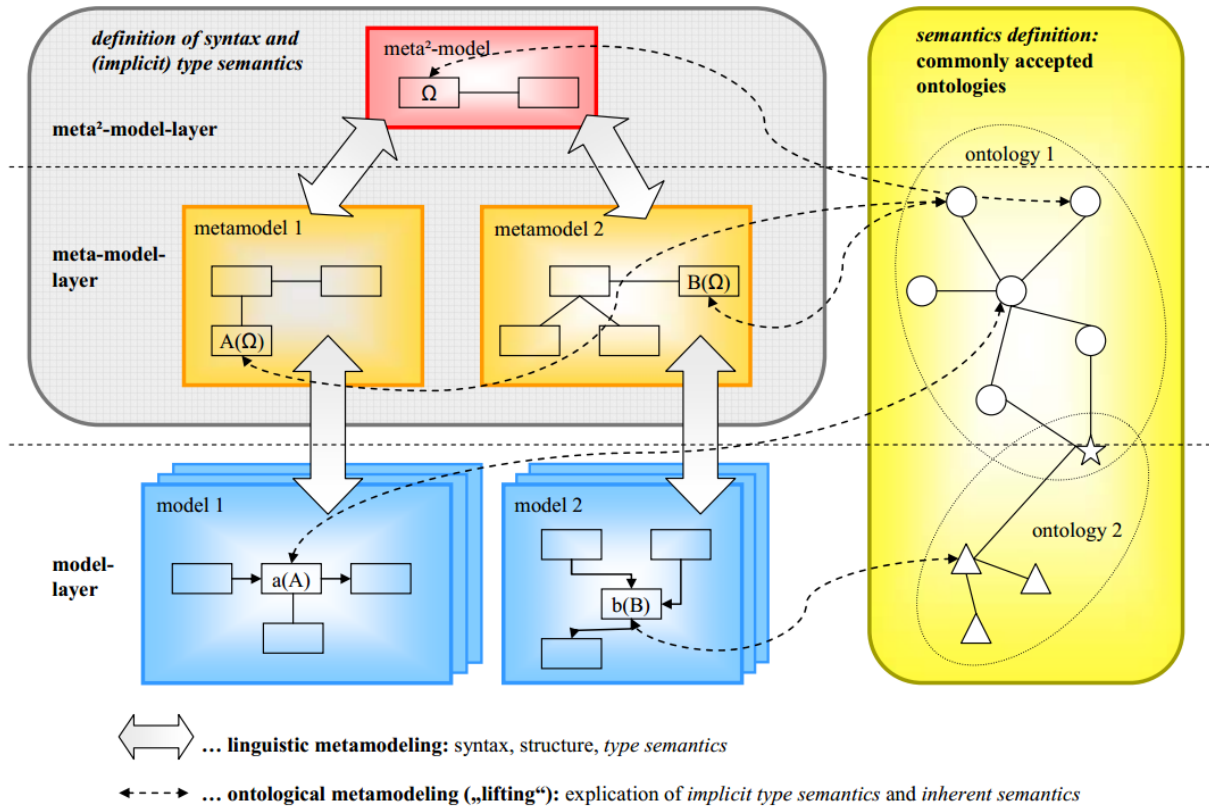
The ontologies are independent from the concepts for the graphical languages. The basis for interoperability is provided via linking model elements of the metamodels with ontology concepts. Since ontologies are, of course, also models, they need to use a language that is also defined by metamodels. Kühn has identified four kinds of merging patterns, which can be applied for integrating enterprise architecture and Enterprise Ontologies, as means to integrate different types of metamodels [55], [56] :

- Reference pattern: defines links that relate exactly one element in the EA metamodel to exactly one element in the ontology metamodel.
- Extension pattern: specifies how the EA model can be extended by concepts of the EO. New concepts can be integrated.
- Transformation pattern: rules are responsible for creating parts of one or more EA framework metamodels in an ontology. This mechanism enables for example the generation of an ontology from a business process.
- Merge pattern: The merge pattern can be regarded as a specialization of the transformation pattern, where a merge rule generates a part of the ontology from two or more EA framework models

The transformation between enterprise architecture and enterprise ontology, which makes the semantics of the graphical models explicit, is called lifting [10]. It has been implemented in ADOxx® [11].

In the project plugIT we applied this approach to enable a computer-supported business-IT alignment using semantic technologies [57]. Business people and IT providers externalise their knowledge via the use of graphical models. These models are then translated into instances of enterprise and domain ontologies to enable automated support of business and IT alignment.

The disadvantage of completely separating metamodels and ontologies is that they can have incompatible semantics. To overcome the problem, in the LearnPAd project (http://www.learnpad.eu) we initially agreed on a shared understanding of important terms before defining the metamodels and the ontologies. There is no way, however, to strictly enforce that the semantics of metamodels and ontologiesare consistent.

4.2.2 Semantic Metamodeling

In order to avoid the consistency problem between metamodels and ontologies we propose a semantic metamodeling approach. The ontology defines the complete semantics of all the concepts. The ontology is extended by a specification of the graphical notation, which corresponds to the concrete syntax of the modeling

language. The difference to the transformation approach is that the semantics is expressed only once for both human-interpretable enterprise architecture and machine-interpretable enterprise ontology. The semantic modeling can be regarded as a variant of the MOF metamodeling framework [43] where UML is replaced by an ontology language as a metamodeling language.

The graphical notation for each concept is defined separate from the semantic description (see Figure 4.3). A mapping is defined between concept definition and graphical definition [58]. This is a difference to the approach of Figure 1.1, where the graphical notation is part to the class definition.This semantic metamodeling approach has been prototypically implemented in the ATHENE system [59]. The concepts of the modeling language are defined using the Resource Description Framework RDFS 3.0.
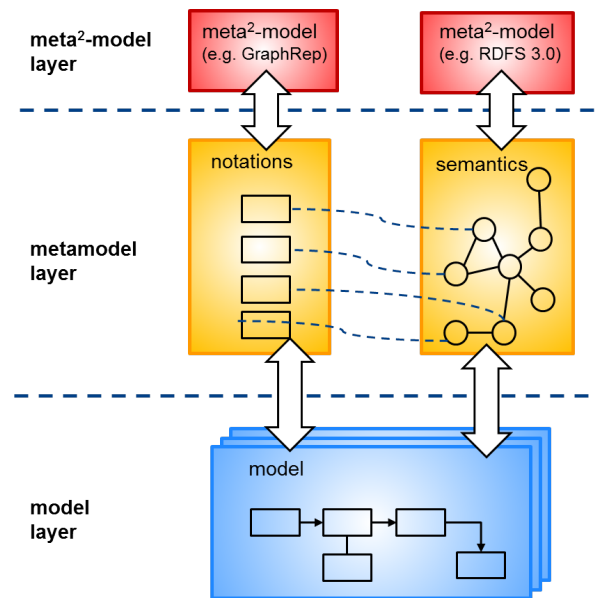

Figure 4.3 Semantic metamodeling

4.3 Mechanisms for Identification of Adaptation Needs

Enterprise engineering should be a conscious, purposeful endeavor, and managers should regularly review their business processes and information systems from multiple perspectives to ascertain whether they are meeting enterprise needs [1]. This is typically a human task. Business analysts exploit the human-interpretable, graphical models in order to detect potential risks and to seize opportunities.

Because of the complexity of the enterprise and the enterprise models it cannot be assumed that business analysts are able to detect all required changes and are able to assess the consequences of all potential influencers. This is where machine-interpretable models and enterprise ontologies could contribute.

In a recent study it has been shown that adaptation needs of an enterprise architecture can be identified by observing content of information systems [60]. Several events have been recognized, which can be checked automatically in order to trigger adaptations of the enterprise architecture. This requires the enterprise architecture to be represented in a machine-interpretable way with a formal semantics, as it is defined by an enterprise ontology. The actual change of the enterprise architecture and of information systems still requires human judgment. Thus, a combination of machine-interpretable models to identify adaptation needs and human-interpretable graphical models to support a business analyst in making appropriate decisions on how to implement the changes offers new opportunities for continuous business-IT alignment.

A similar approach was used to improve contract management in the DokLife project [61]. Monitoring obligations and liabilities is time consuming and error prone. Whereas Contract Management Systems (CMS) deal well with time-triggered obligations like periodical payments, they fail to trigger obligations based on events, as this knowledge is out of the systems' scope. In the DokLife project we introduced an approach to fill the gap as we relate information about the obligations managed in a CMS with background knowledge modelled in an enterprise ontology, e.g. processes to be triggered, responsible roles and required resources. This allows to trigger processes based on pre-defined events.

In the APPRIS project we have shown how an enterprise ontology can be applied to analyze early warning indicators for supply risk management [36]. An inference engine regularly assesses data from various internal and external information sources in order to identify potential procurement risks. Risks depend on enterprise knowledge which is represented in the enterprise ontology. Results of risk identification and assessment are

displayed in an easy-to-understand way for a human user to decide for appropriate actions. This means that the knowledge needs to be understood by both machines (for risk assessment) and humans (for deciding about actions). The combination of graphical and formal representations with a common semantics is an appropriate approach.

5 Conclusion

In this paper we proposed a new paradigm for next generation enterprise information systems, which shifts the development approach of model-driven engineering to *continuous adaptation* of the agile enterprise. Enterprise information systems are closely integrated with (1) model analysis tools which allow assessing influencers, to identify risks and to seize opportunities and (2) modeling tools for changing the enterprise. We propose a metamodeling approach for the implementation of these information systems, which support both human-interpretable graphical models with machine-interpretable enterprise ontologies. We showed that the integration is possible; it has been applied in several projects. It is still some time until commercial tools are available and business architects and IT architects are using this modeling paradigm.

It is a future long-term challenge to involve business people not only in the adaptation of enterprise architecture but also into the implementation and adaptation of enterprise information systems. Evolving application flexibility via embedded modeling tools has been identified in a recent study as one of the 10 most important technology trends in business application architecture [62]. The authors predict that future business applications will incorporate business-oriented graphical modeling tools that enable rapid, code-free modifications to business applications, including process orchestration, business rules, notification, organizational structures, embedded business intelligence, and even the assembly of new functionality from existing functional elements. To automate the modification and adaption of applications - or at least to support the human user in adapting the current models - a formal semantics of the models is essential. The modeling approach presented in this paper provides a solid basis for this future challenge.

References

[1]   R. E. Giachetti, *Design of Enterprise Systems: Theory, Architecture, and Methods*. CRC Press, 2011.

[2]   R. Dove, "Knowledge management, response ability, and the agile enterprise," *J. Knowl. Manag.*, vol. 3, no. 1, pp. 18–35, Jan. 1999.

[3]   R. Dove, "Agile Enterprise Cornerstones: Knowledge, Values & Response Ability," in *Business Agility and Information Technology Diffusion, IFIP TC8 WG 8.6 International Working Conference, Atlanta*, 2005, vol. 180, pp. 313–330.

[4]   F. A. Cummins, *Building the Agile Enterprise with SOA, BPM and MBM*. Burlington: Morgan Kaufmann Publishers, 2009.

[5]   M. Op't Land, E. Proper, M. Waage, J. Cloo, and C. Steghuis, *Enterprise Architecture*. Springer, 2008.

[6]   Gartner, "Taming the Digital Dragon: The 2014 CIO Agenda." Gartner, pp. 1–12, 2014.

[7]   M. Völter, T. Stahl, J. Bettin, A. Haase, and S. Helsen, *Model-Driven Software Development: Technology, Engineering, Management*. John Wiley & Sons, 2013.

[8]   D. McCauley, "Achieving Agility," in *Mastering the Unpredictable*, K. D. Swenson, Ed. Meghan-Kiffer Press, 2010, pp. 257–275.

[9]   D. Karagiannis and R. Woitsch, "Knowledge Engineering in Business Process Management," in *Handbook on Business Process Management 2*, Berlin Heidelberg: Springer, 2010, pp. 463–485.

[10]   G. Kappel, E. Kapsammer, H. Kargl, G. Kramler, T. Reiter, W. Retschitzegger, W. Schwinger, and M. Wimmer, "Lifting Metamodels to Ontologies: A Step to the Semantic Integration of Modeling Languages," in *Model Driven Engineering Languages and Systems, Proceedings of the 9th International Conference, MoDELS 2006*, LNCS 4199., O. Nierstrasz, J. Whittle, D. Harel, and G. Reggio, Eds. Genova, Italy: Springer-Verlag, 2006, pp. 528–542.

[11]   V. Hrgovcic, D. Karagiannis, and R. Woitsch, "Conceptual Modeling of the Organisational Aspects for Distributed Applications: The Semantic Lifting Approach," in *COMPSACW 2013, 2013 IEEE 37th Annual Computer Software and Applications Conference Workshops*, 2013, pp. 145–150.

[12]   M. Reichert and B. Weber, *Enabling Flexibility in Process-Aware Information Systems*, no. c. Berlin, Heidelberg: Springer, 2012.

[13]   J. L. G. Dietz, *Enterprise Ontology. Theory and Methodology*. Berlin Heidelberg: Springer-Verlag, 2006.

[14]   D. Chen, G. Doumeingts, and F. Vernadat, "Architectures for enterprise integration and interoperability : Past , present and future," *Comput. Ind.*, vol. 59, pp. 647–659, 2008.

[15]   J. A. Zachman, "John Zachman's Concise Definition of The Zachman Framework," *Zachman International*, 2008. [Online]. Available: http://www.zachman.com/about-the-zachman-framework.

[16]   ISO/IEC, "ISO/IEC/IEEE 42010 Systems and Software Engineering - Architecture Description," 2011.

[17]   J. A. Zachman, *The Zachman Framework For Enterprise Architecture : Primer for Enterprise Engineering and Manufacturing By*, Electronic. Zachman International, 2003.

[18]   D. Matthes, *Enterprise Architecture Frameworks Kompendium*. Heidelberg: Springer-Verlag, 2011.

[19]   J. A. Zachman, "The Zachman Framework: The Official Concise Definition," *Zachman International*, 2008. [Online]. Available: http://test.zachmaninternational.com/index.php/the-zachman-framework.

[20]   J. F. Sowa and J. A. Zachman, "Extending and formalizing the framework for information systems architecture," *IBM Syst. J.*, vol. 31, no. 3, 1992.

[21]   The Open Group, "TOGAF 9 V91." 2011.

[22]   The Open Group, *ArchiMate 2.0 Specification*. The Open Group, 2012.

[23]   OMG, "Business Process Model and Notation (BPMN) Version 2.0," 2011.

[24]   OMG, "Case Management Model and Notation (CMMN), V 1.0," 2013.

[25]   OMG, "Business Motivation Model, Version 1.1," Object Management Group OMG, 2010.

[26]   H. Franken, S. van Dijk, and B. van Gils, "Three Best Practices for Successful Implementation of Enterprise Architecture Using the TOGAF® Framework and the ArchiMate® Modeling Language," *Blog*, 2013. .

[27]   K. Hinkelmann, A. Martin, S. Nikles, and B. Thönssen, "ArchiMEO : Formal Semantics for ArchiMate," no. Mc. .

[28]   B. Thönssen, "Automatic, Format-independent Generation of Metadata for Documents Based on Semantically Enriched Context Information," University of Camerino, 2013.

[29]   D. Kang, J. Lee, S. Choi, and K. Kim, "An ontology-based Enterprise Architecture," *Expert Syst. Appl.*, vol. 37, no. 2, pp. 1456–1464, 2010.

[30]   K. Hinkelmann, E. Merelli, and B. Thönssen, "The Role of Content and Context in Enterprise Repositories," in *2nd International Workshop on Advanced Enterprise Architecture and Repositories - AER 2010*, 2010, no. Cmd.

[31]   M. S. Fox, M. Barbuceanu, and M. Grüninger, "An organisation ontology for enterprise modeling: Preliminary concepts for linking structure and behaviour," *Comput. Ind.*, vol. 29, no. 1–2, pp. 123–134, Jul. 1996.

[32]   M. Uschold, M. King, S. Moralee, and Y. Zorgios, "The Enterprise Ontology," *Knowl. Eng. Rev.*, vol. 13 (Specia, 1997.

[33]   A. Abecker, A. Bernardi, K. Hinkelmann, O. Kühn, M. Sintek, and H. E. R. T. Knowl-, "Toward a Technology for Organizational Memories," *IEEE Intell. Syst. their Appl.*, vol. 13, no. 3, pp. 40–48, 1998.

[34]   M. Uschold, M. Healy, K. Williamson, P. Clark, and S. Woods, "Ontology reuse and application," in *International Conference on Formal Ontology in Information Systems FOIS'98*, 1998, vol. 179, p. 192.

[35]   J. Den Haan, "An Enterprise Ontology based approach to Model-Driven Engineering," TU Delft, Delft University of Technology, 2009.

[36]   S. Emmenegger, K. Hinkelmann, E. Laurenzi, and B. Thönssen, "Towards a Procedure for Assessing Supply Chain Risks Using Semantic Technologies," in *Knowledge Discovery, Knowledge Engineering and Knowledge Management*, A. Fred, J. L. G. Dietz, K. Liu, and J. Filipe, Eds. Springer Berlin Heidelberg, 2013, pp. 393–409.

[37]   K. Hinkelmann, M. Maise, and B. Thönssen, "Connecting Enterprise Architecture and Information Objects Using an Enterprise Ontology," in *IEEE International Conference on Enterprise Systems*, 2013.

[38]   D. Allemang and J. Hendler, *Semantic Web for the Working Ontologist Effective Modeling in RDFS and OWL Second Edition*, Second Edi. Morgan Kaufmann Publishers, 2011.

[39]   R. J. Brachman and H. J. Levesque, *Knowledge Representation and Reasoning*. Morgan Kaufmann Publishers, 2004.

[40]   F. Baader, "What's new in Description Logics," *Informatik-Spektrum*, vol. 34, no. 5, pp. 434–442, Apr. 2011.

[41]   D. Karagiannis and H. Kühn, "Metamodelling Platforms," in *Proceedings of the Third International Conference EC-Web at DEXA 2002*, 2002.

[42]   S. Strahringer, *Metamodellierung als Instrument des Methodenvergleichs: eine Evaluierung am Beispiel objektorientierter Analysemethoden*. Aachen: Shaker, 1996.

[43]   OMG, "OMG Meta Object Facility (MOF) Core Specification Version 2.4.2," 2014.

[44]   OMG, "Ontology Definition Metamodel," vol. 09, no. May. Object Management Group OMG, 2009.

[45]   D. Karagiannis, S. Junginger, and R. Strobl, "Introduction to Business Process Management Systems Concepts," in *Business Process Management*, B. Scholz-Reuter and E. Stickel, Eds. LNCS: Springer-Verlag, 1996.

[46]   S. Junginger, H. Kühn, R. Strobl, and D. Karagiannis, "Ein Geschäftsprozessmanagement-Werkzeug der nächsten Generation - ADONIS: Konzeption und Anwendungen," *Wirtschaftsinformatik*, vol. 42, no. 5, pp. 392–401, 2000.

[47]   D. Karagiannis and P. Höfferer, "Metamodels in Action: an Overview," in *ICSOFT 2006 – First International Conference on Software and Data Technologies*, 2006.

[48]   D. Karagiannis, W. Grossmann, and P. Höfferer, "Open Model Initiative - A Feasibility Study," Sep. 2008.

[49]   OMiLAB, *Open Model Laboratory Booklet*. University of Vienna, 2014.

[50]   K. Hinkelmann and A. Pasquini, "Supporting Business and IT Alignment by Modeling Business and IT Strategy and its Relations to Enterprise Architecture," in *2nd IEEE Conference on Enterprise Systems*, 2014.

[51]   N. Sudakova and K. Hinkelmann, "Explicitly Modelling Relationships of Risks on Business Architecture," in *eChallenges e-2014 Conference Proceedings*, 2014.

[52]   P. Höfferer, "Achieving Business Process Model Interoperability Using Metamodels and Ontologies.," in *European Conference on Information Systems*, 2007, pp. 1620–1631.

[53]   G. Kramler, G. Kappel, T. Reiter, E. Kapsammer, W. Retschitzegger, and W. Schwinger, "Towards a semantic infrastructure supporting model-based tool integration," in *GaMMa '06: Proceedings of the 2006 international workshop on Global integrated model management*, 2006, pp. 43–46.

[54]   Y. Liao, M. Lezoche, H. Panetto, N. Boudjlida, and E. R. Loures, "Semantic annotation for knowledge explicitation in a product lifecycle management context: A survey," *Comput. Ind.*, vol. 71, pp. 24–34, 2015.

[55]   H. Kühn, "Methodenintegration im Business Engineering," University of Vienna, 2004.

[56]   H. Kühn, F. Bayer, S. Juninger, and D. Karagiannis, "Enterprise Model Integration," in *Proceedings of the 4th International Conference EC-Web 2003   Dexa 2003*, 2003, pp. 379–392.

[57]   R. Woitsch, D. Karagiannis, D. Plexousakis, and K. Hinkelmann, "Business and IT alignment: the IT-Socket," *Elektrotechnik und Informationstechnik*, vol. 126, no. 7–8, pp. 308–321, Jul. 2009.

[58]   S. Nikles and S. Brander, "Separating Conceptual and Visual Aspects in Meta-Modeling," in *Workshop on Advanced Enterprise Architecture and Repositories*, 2009.

[59]   K. Hinkelmann, S. Nikles, B. Thönssen, and L. von Arx, "An Ontology-Based Modeling Tool for Knowledge-Intensive E-Government Services," in *MeTTeG07, Proceedings of the 1st International Conference on Methodologies, Technologies and Tools Enabling E-Government*, 2007, pp. 43–56.

[60]   M. Frei, "Event and Data Sources for Enterprise Architecture Documentation," University of Applied Sciences and Arts Northwestern Switzerland, 2015.

[61]   B. Thönssen and J. Lutz, "Semantically Enriched Obligation Management. An Approach for Improving the Handling of Obligations Represented in Contracts," in *Proceedings of 4th Conference on Knowledge Management and Information Sharing (KMIS2012)*, 2012.

[62]   J. Hoppermann, P. D. Hamerman, and G. Lawrie, "The 10 Most Important Technology Trends In Business Application Architecture Today," Cambridge, MA, Sep. 2013.