PERSPECTIVES ON SEMANTIC BUSINESS PROCESS MODELING – A GENERIC APPROACH

Martin Juhrisch* Gunnar Dietz† Werner Esswein‡

*Technische Universität Dresden, juhrisch@uni-muenster.de
†Multimedia Kontor Hamburg gGmbH, g.dietz@mmkh.de
‡Technische Universität Dresden, werner.esswein@tu-dresden.de

This paper is posted at AIS Electronic Library (AISeL).
http://aisel.aisnet.org/pacis2009/63
PERSPECTIVES ON SEMANTIC BUSINESS PROCESS MODELING – A GENERIC APPROACH

Martin Juhrisch
University of Technology Dresden
Chair of Business Informatics, esp. Systems Engineering
Dresden 01062, Germany
martin.juhrisch@tu-dresden.de

Gunnar Dietz
Multimedia Kontor Hamburg GmbH
Project eCampus
Hamburg 22081, Germany
g.dietz@mmkh.de

Werner Esswein
Professor of Business Informatics
University of Technology Dresden
Chair of Business Informatics, esp. Systems Engineering
Dresden 01062, Germany
werner.esswein@tu-dresden.de

ABSTRACT
With the rise of SOA advance, organizations require methods and tools to analyze services regarding their fit to the needs of an institution. Research on the evaluation and selection of services has mainly adopted a non-functional perspective so far. Hence, in this paper we propose a framework for analyzing services from a business process perspective. Since we cannot automatically adjust an informal (respectively semi-formal) business process description with a formal service model, an automatic test on feasibility of the service configuration in an enterprise model is hampered. Additionally the rising number of service developers as well as the comparability requirement for models resp. model evaluation necessitates a convention-oriented constructive restriction of the degree of freedom in specific conceptual models. In this paper a generic method is represented that by adopting of Description Kits allows a restriction of freedom in modeling regarding the aspects of natural language in specific conceptual models and enables a restrictive use of existing modeling languages. As a use case the configuration of service-oriented architectures is discussed. By describing both business processes and services in such a way, we are able to formulate a mapping algorithm, which allows for analyzing and evaluating the suitability of services for specific requirements.

Keywords: Business process modeling, Description Kits, Service-oriented Architectures, Web service.

1 INTRODUCTION

The effective support of existing business processes through the direct operational link with the supporting application system is one of the main promises of the Service-oriented Architecture (SOA) paradigm (Papazoglou et al. 2006; Kossmann and Leymann 2004). The application system is created by the combination of services, which provide defined business functionality, and can be used as part of a larger processing sequence.
In an organization there exists a variety of interrelated services and their underlying processes. On the other hand, there exist a number of services that show reorganization potential for the process landscape. These include services from document management, workflow management or archiving systems, etc. The complexity of a process landscape in conjunction with the multitude of available services confront decision-makers with the problem of finding adequate areas of application for services in their process landscape. The analysis of organizational processes is a prerequisite for an efficient and goal-oriented IT support of workflow- and organizational processes.

To enhance the benefit of a model creation and thus to increase the use of models in general, they should not, however, only be used for the identification of service candidates, but the model based management of service-oriented architectures. A model-based configuration of SOA allows automating tasks previously performed manually. As a result, the required implementation knowledge to cope with the task is reduced, which reduces concurrently the overall time and effort for solving the task.

This promotes the goal-oriented configuration of SOA since the absence of implementation details allows concentrating on the problem salvation from a business perspective. A model-based method is conceivable in the following scenarios:

A first scenario is the identification of potential services out of business models. The identification is a challenging task if the process models are created in a distributed environment as well as in the case of strong interdependencies among the models. Secondly, the review of whether certain business requirements can be implemented through service compositions. A review is being considered as a comparison between requirements and implementation models (Priemer 1995). Requirements models must be aligned with the services available, and then be prepared for the translation into service compositions. Third, changes in the deployment of services and the adjustment to the existing business processes. Modifications within the service implementation need to induce changes in the involved requirements models. This necessitates a proof that clarifies which services are used by which processes.

The scenarios are not entirely separable. Once the model matching identifies gaps in the underlying SOA, an interchange between the scenarios can be observed. Ultimately, all scenarios are connected with each other. A method will be presented for all described scenarios allowing a planned, consistent procedure. Since the method is designed to at least semi-automate the model comparison, we require a shape of the models that allows an appropriate evaluation. This requires to partially describing real-world phenomena using a formal semantic. As a result, algorithms based on a set of rules perform the operations on the models.

2 DISTINGUISHING THE MODELLING PROBLEM

In Business Informatics models play a central role in general as well as especially in system development scenarios (Ferstl und Sinz 2001). The applicability of models for structuring, describing and analyzing a certain problem by restructuring it just enables the possibility for solving this problem within the modelling domain with the intention to transfer the solution into the problem (or real world) domain (see Figure 1).
Models for solving functional problems have applications in many domains ranging from development of organizations to application system development (Rosemann 2003; Kugeler 2000). The initial application for models has been the domain of software development. On the base of analysis models the creation of design or implementation models follows.

2.1 Classical Requirements Models

For the requirements analysis one uses models to represent functional requirements to a certain application system (Teubner 1999). Requirements models are created by using conceptual modeling languages. In contrast to pure organizational models the application system to be developed is part of the consideration by means of being responsible for operational tasks (Balzert 2001; Sommerville 2001). One important requirement to the language constructs in this case is to avoid in any case restrictions to the modeling task itself within the functional domain (Frank 1999). Information that touches implementation details cannot be part of conceptual models if the freedom for finding a consensus during the analysis phase should not be influenced. Therefore these models do not support the transformation task. This becomes obvious when looking at the sharp increase at the left part of the Abstraction-Transformation diagram (see Figure 1). Therefore the ability of an automatic transformation to implementation models must be doubted. Classical requirements models lack in applicable styles with regards to both language and content. Both requirements can’t be met by analysis models (Juhrisch and Weller 2008). An automatic analysis of conceptual models fails due to the lack of being applicable for unambiguous interpretation by a machine. Since requirements either come from organizational models or from the problem domain itself, they always arise from a material domain and are artifacts in need to be interpreted. Techniques that propagate to use formal instruments to describe requirements models ignore the principle of language adequacy as part of principals of proper modeling as evaluated by (Schütte 1998) with regards to applicability for the modeling goal. Therefore up to now there are no adequate preconditions to meaningful quantification and operationalization. In Figure 1 this fact is illustrated as the semantic gap.
2.2 Implementation Models

In the classical understanding of system development it follows now the IT design follows. The result should be a design model that is a blueprint for a solution for the problem that has been analyzed. The main goal is the construction of a system architecture (Balzert 2001). Finally the system development procedure comes to an end by implementing the design model. The set of information described in classical implementation models is primarily depended on the development paradigm or methodology in use (Balzert 2001). Their development is done independently from the application layer. This fact results in a sharp descend from the design model to the implementation (see Figure 1). For service models in the sense of a service-oriented architecture (SOA) this is different however. Service models are specific implementation models. Services offer a well defined functionality that encapsulated business needs and can be used as part of greater processes (Turowksi 2001). Instead of developing an application system as a whole, in a design model certain processes can be composed as required in a process oriented way (Turowski 2001). With respect to the system development by means of composition of SOA services the paradigm of a strictly sequentially procedure of analysis phase and design phase has to be rethought. Services get the role of objects to be analyzed and should be thought of as system parts to be integrated. Therefore the main role of system design is not anymore the transformation of analysis models to design models, but models of services are used for a comparison with design models, either to identify service candidates for an implementation or to find existing services that can be used for (parts of) process within analysis models. Such a comparison requires that service models or a catalog of services exist with applicable service descriptions.

The model comparison can be understood as a mapping of solutions in the form of existing design artifacts to problems. The destination system of the transformation represents a set of alternative solutions for a certain problem in question. The developer then inspect these services with respect to their applicability to the certain problem and utilizes them. This has consequences to both described modeling phases of the application system development, since the analysis phase is already influenced by existing services. On the other hand, design criteria for services have to be fitted to the business needs, i.e. they have to be oriented to the application domain.

3 REQUIREMENTS ANALYSIS

On the one hand requirements to a method with the goal to compare models as described will be determined. They are determined by the certain goal of the model comparison. Furthermore there are requirements as a result of formal objectives to certain model transformation in question.

The method should be able to semantically compare conceptual models within different domains. This means to resolve language conflicts and structure conflicts as well as to resolve the domain conflict (see Dietz et al. 2009). Quality requirements for models of the analysis phase have to be kept.

R1: It will be evaluated how the method in question is able to distinguish between model concepts for describing real world phenomena and model concepts that have a relevancy for application development within the source model. That means restrictions to the normal modeling process for domain experts should be avoided in any case. The reason for R1 is that the describing instruments really should contribute to an understanding of the models and that the gap between generally described usage requirements and precise technical specifications can be closed.

R2: Resolving the domain conflict by the possibility of using artifacts from different domains commonly or the possibility of comparing these artifacts at least semi-automated respectively. Therefore a procedure model or an algorithm should be able to compare certain object
descriptions within the requirement model and a given object description within the service model.

R3: Resolving language conflicts: The method should resolve the semantic heterogeneity of specialized language constructs. Semantically disjoint and domain specific language constructs should be placed at the modeler’s disposal.

To do this one needs to create a language community, which persons that want to model functional and persons that want to model technical take an active part in. To resolve language defects one needs to establish a consensus for a terminology during the process of language determination. This terminology has to be included into the language based metamodel for the modeling language. Therefore the method has to include the process of establishing this consensus.

R4: User control for the model transformation. The software developer and the domain expert should have the possibility to have a widespread control of the model transformation. Even if the method can resolve language conflicts and structure conflicts, the domain conflict can’t be avoided completely. The reason for this is that the material semantic could not be completely deduced from its syntactic form. There are no conditions that allow mapping certain concepts in a unique way. Due to this fuzzyness the domain experts should be able to control the mapping results. This controlling includes tasks that could not be fully automated by the method, e.g. some parameterizations in the beginning or some decisions between transformation alternatives during the process of the transformation.

R5: A congruent explication of real world phenomena by a model creation process that uses conventions in connection to semantically disjoint language constructs. These conventions should enforce a consistent and standardized usage of the remaining freedom of the modeling language to improve the degree of automation (or to make it possible at all).

R6: Taking non-functional requirements into account. The model comparison method should consider non-functional requirements to services when mapping these to processes or process functions respectively.

R7: Definition of a model migration. The method should allow a mapping between exactly two modeling grammars. In this scenario both models should have some arbitrary (but determined) grammar. To do this one needs to define a procedure model and an algorithm, how to create a destination model from the source model and mapping policies.

All these requirements can be deduced from the formal goals of the method. Requirements that could be deduced from formal goals of some method specify the design of some new method or the evaluation of an existing one (Gehlert 2007). So these requirements address the whole process. A method is demanded, which should be carried out by using minimal resources. This implies an automation of all tasks that could be automated. Furthermore – adopting the concept of truth used in this article – the model transformation is done correctly exactly when there is a consensus of the product. So it is required that the transformation process is comprehensible. This transparency should be accomplished by a detailed documentation of the method.

4 MODEL BASED MANAGEMENT

Since modeling methodologies are normally just concerned with the creation of models and don’t generally focus on the ability to solve functional problems they may be able to have no reference to the problem domain. Therefore they are not very useful for model based problem solving. A model based methodology – in contrast to a method bundle (Becker, et.al. 2001) – may be not only composed of modeling methods. To be applicable in the sense of model based problem
solving – and this has to be done systematically – the method artifacts not only have to be adapted (normalized) in the sense of a method bundle, but also include another aspect going further. The methodology has to describe the model creation and modification for solving problems within the modeling space as well as the model utilization, especially the transformation to the problem domain. To solve functional problems with the help of a modeling language, the fact that one has to consider a certain problem domain adds to the task to describe the modeling language and the process of modeling.

In the sense of this article a model based methodology therefore not only consists of integrated language and procedure descriptions of the model language(s) to use, but also of artefacts of the problem solving techniques and a reference to the problem domain. All this should be subsumed as the intentional aspect of modeling.

1 Guidelines (that result in a constrained form of modeling) include the intentional aspect of the modeling process and can’t be described by classical means of metamodeling (data or process modeling respectively). This intentional aspect interacts with the language and modeling process descriptions (see Figure 1).

The term “problem domain” includes several aspects, all based on the requirement of solving a problem within a certain situation. This is deeply influenced by the operational but also cultural context and certain use cases (actual vs. theoretical comparison, aggregation of domains).

Only domain specific guidelines (constraints) with influences on the language and process based metamodel can complete a model based methodology for solving functional problems. This includes disposing the modeling language and procedure model for being applicable for solving the problem. The conventions result in a constrained use of modeling techniques or artifacts, constraints adapted to the problem domain in question. This should be done in a generic way. Generically should be described how real world phenomena and problems should result in guidelines for a restricted modeling. A real relation between the (real world) object and the model should be established. This aspect is not originally covered by a modeling procedure model.
A multitude of modelers should independently describe their universe of discourse within their individual domain. Domains could be distinct like cross-organizational or cross-language domains, or unequal in the perspective of describing a certain object.

In the case of SOA management, domains differ in their underlying semantic between a material semantic of conceptual models and a formal semantic of IT services (Juhrisch and Weller, 2008). Now we can address the problem as follows: On the one hand the modeling process in each domain should not be affected by implementation details. However, the individually designed conceptual models should afterwards being applicable for the model driven problem solution (creation of service composition). Indeed, for the application of automated problem solution techniques we require a subject invariant understanding of the modeled data.

To reach a conjoint understanding of certain data within two (or more) separated domains we influence the modeling process. The use of guidelines helps to control the process of description of certain data either in conceptual or design models (see Figure 1). Thus, a common understanding of shared models is forced by following the guidelines. Afterwards, the models can be utilized for the model based problem solution.

Here, we present a generic approach since:

1. The procedure of controlling the modeling process is depending on the particular problem that has to be solved in a model driven way. The problem could be the model driven configuration of enterprise systems, the management of identity management systems or the model based identification of service candidates (Juhrisch and Weller 2008; Dietz et al. 2009).
A domain specific modeling language is always established over time; its vocabulary and grammar are not static but a result of the use of language in a speech community. The determination of domain specific language constructs at meta-model level seems to be problematic with respect to the background of dynamic change.

We restrict the freedom in modeling within conceptual models in order to limit the language vocabulary to an amount of domain-specific, semantically disjoint language constructs. With this not only the designed conceptual model but also the modeling language has a semantic connection to the application domain (Pfeiffer 2007). Therefore, from the application point of view semantically meaningful operations in conceptual models can be defined already at a language level.

5 THE DESCRIPTION KIT APPROACH

In the present paper Description Kits (DK) are introduced that cover restricted describable ancillary information in adequately enriched conceptual models. DK represent the consensus of the speech community in terms of the amount and structure of certain linguistic concepts relevant for the business analysis. The DK approach is generic enough to restrict every kind of modeling information in their description relating to the present modeling purpose. Concrete descriptions of business information in analysis models concretize the imagination of the modeler at purely linguistic level within the scope of given DKs.

We refer to (Dietz et al. 2009) for a detailed description of this approach. Central idea is focus on the gap between language creation and language usage in the sense of the metamodeling methodology, which introduces several modeling layers for language creation and language usage (on metamodel layer or layer 1 the language itself is defined to be used by means of instantiation on ordinary modeling layer or layer 0). This gap is filled by introducing and additional layer 0* that serves as the layer for the definition of guidelines in the sense of description kits (see Figure 2). Figure 3 shows the different layers of the approach from different viewpoints. The next chapter, which describes how to use the approach in the use case of service-oriented architecture, should demonstrate the general idea.

![Figure 3. Layers of the Description Kit approach](image-url)

The DK approach comes with a set of algorithms to support the comparability and transformation of models. We again refer to (Dietz et al. 2009) for a detailed description.
6 PILOT STUDY

With the continuous consolidation of Web service standards a change away from the implementation and deployment of services to the point of service management is taking place in the discussion about the service-oriented architectures. The indicators: a number of standardization requests and the number of large projects in this area are evident for a growing demand on management methods for illustrating business requirements on service compositions. An application of the Description Kit approaches is represented that enables a targeted coordination between the relevant business processes and SOA in specific conceptual models.

Referred to the abstraction levels of the Model Driven Architecture MDA (MDA 2003) the illustration classifies as a transformation task between computation independent models (CIM) and platform independent models (PIM). Now, from the present modeling purpose it can be deduced that an automatic transfer requires a translation of real-world phenomena in a form that enables an appropriate automation. Reduction of semantic heterogeneity inside the modeling elements in CIM and PIM is the deciding step to an automatable transfer. The main hypothesis is that a connection can be established between an organization and IT domains, provided that an amount of linguistic concepts are used simultaneous in CIM and PIM.

Therefore, in the following an exemplary use of the DK approach to overcome semantic gaps between semi-formal problem description in CIM and formal solution in PIM is used. The goal is to reach the comparability of CIM and PIM.

For the modeling aim here the freedom in matters of modeling of business objects or parts of them is restricted. The approach centers on objects, since they are of high importance both in the analysis and in the design phase and qualify well for an automatic analysis for service candidates in CIM and for adjustments between CIM and PIM. In the case of the DK method the following steps need to be taken:

Level 1 (meta-model level): At meta-model level a hierarchic DKL is developed. Figure 4 illustrates a DKL that serves for the modeling of Object, Service and Role Description Kits at level 0*. The DKL contains general DK concepts and (not shown) the relation to the modeling language used. The concept Description Kit has to be emulated on the existing meta-model of the modeling language – in the present case it is referred to the meta-model of the event-driven process chain (EPC) as part of the ARIS method (Scheer 2000).

Level 0*: On level 0* concrete DKs are defined using the concepts of layer 1. DKs for business objects, services and role at level 0* are developed. These DKs are adaptive, thus they can be adapted to the modified requirements to the modeling of business objects during the modeling at object model level. In this case, DKs build adaptive model elements corresponding to the consensus between software architects and specialists concerning the amount and structure of certain linguistic concepts relevant for the analysis and the design.

Level 0 (model level): On the normal modeling level (level 0) the DKs are used to define concrete descriptions. These are concrete objects or object states on the one hand (as an active or inactive student) and concrete process description in the sense of interfaces. The modeling of objects and services will be limited to the vocabulary predetermined by the DK.
6.1 Conceptual Modeling with Descriptions

After defining DKs on layer 0* these can be used on ordinary modeling level (layer 0). To use them they have to be instantiated. The instantiation of a Description Kit creates a concrete description on the model layer. However, such a description doesn't need to describe a concrete real world phenomena, but can also describe only some aspects. E.g. in Figure 5 the description "Student" is still abstract in the sense that it doesn't represent a concrete student but the conditions to a person to be a student. On the analysis phase this description can be used for the definition of concrete business objects. For that reason, state transition models are used that – based on an extension of the EPC – describe functional requirements (see again Figure 5).

Thereby, a semantic relationship between an event and descriptions is created and interpreted as an operational condition. This becomes clear by gathering the conditions of the allocated
descriptions. The business logic to be automated is then documented as an amount of situation transitions in descriptions.

6.2 Service Modeling with Descriptions

With the E-method (Greiffenberg 2004) a model for services is created that includes an illustration of a service interface by means of graphic characters and links the method signatures with instantiated DKs “Object” (DKO).

When defining the message parameters the DKOIs are again to be used. Due to the fact that when modeling CIM or PIM in both cases DKO instances are used, the bridge between these two domains will be closed. It is assumed that the service function of a Web service is suitable for a certain operational context if the initiating resp. resulting event of a process function in a CIM comprehends DKO instances with a condition according to the one existing in the model of the service function. The configuration automatically chooses suitable service functions and – if necessary – an adaptation of a target model is recommended to the modeler. As a result the EPC model is in the best case adapted to the existing SOA. An adapted target model can be transmitted as a reference model – professional solution with underlying implementation – to other organizations.

6.3 Case Study and Project Description

In recent years several German universities started research programs to improve and standardize their administration processes (Dietz et al. 2009). The increasing complexity of IT-based services and infrastructure creates an additional pressure to multilayered changes e.g. aroused by the Bologna process (European project for creating academic degree standards) or modern e-government. The six public universities of Hamburg, namely the University of Hamburg, the Hamburg University of Technology, the Hamburg University of Applied Sciences, the HafenCity University, the Hamburg University for Music and Drama and the University of Fine Arts, together with the university libraries in Hamburg have launched together with the MultimediaKontor Hamburg a project named eCampus, which is currently in the third phase eCampus III, running from 2009 to 2010.

While the implementation phase is still in progress the focus however moves from technical implementation details to the creation of a high service level and the optimization of processes. Therefore, starting with a pre-implementation process analysis now needs for managing the high complexity of changing requirements was one of the reasons for searching for a model-driven way to control the change management. Due to the cooperation of several universities with different requirements especially the aspect of distributed process modeling, comparison of models and model transformation came into the focus, which brought forth a cooperation with the University of Technology Dresden (Germany) and the usage of the modeling tool “cubetto Toolset” (cubetto, 2009).

In a first step, university processes have been analyzed and documented using business process models. In a second step, the identification of core functions within the documented processes was planned to implement services for a software architecture based on the SOA paradigm. Due to the large amount of process models and its strong relationships the task of identifying these service candidates, however, has been realized to be very difficult. As the process models are documented electronically using the cubetto Toolset, the process of identifying the service candidates can be supported by an automatic analysis of the process models with help of the algorithms that come with the DK approach.
7 CONCLUSIONS AND RECOMMENDATIONS

The approach introduced above achieves the first step towards a model driven configuration of SOA via business process models. Only by introducing constraints to restrict the modeling freedom a semi-automatic mapping from processes in enterprise models to services of a given SOA is possible. A service candidate can be found by interpreting the state of DKO.

But the approach is generic and can be used in nearly every modeling domain. Using the description kit approach for modeling role data can be used to configure identity management systems in a model driven way in the sense of (Juhrisch et al. 2009) or for realizing modeling methods like PICTURE (Becker et al. 2007).

A first realization was made by using the E3-method (and using the cubettoÆ Toolset). However, the realization of constraints in the sense of Chapter 3.1 is not easily possible by using the E3-method since formulating language constructs using metamodelling for constraints can be quite complex. A direct support of the description kit approach by the modeling tool would be better.

The additional layer 0* also doesn’t fit seamlessly into classical language-based metaisation, but this additional layer is needed in our approach since something more than just defining a modeling language happens. However, in contrast to the approach presented in (Juhrisch and Weller 2008), where Pattern constructs were used, no modification of the metametamodel is needed anymore.

In contrast to other solutions this approach realizes the restriction of the modeling freedom in a generic way, which means:

1. Comparability of models in shared modeling projects
2. Domain experts can easily create models without too much modeling experience
3. Automated analysis of models, even automated or semi-automated configuration of business logic or SOA
4. Easy adaption of the modeling semantic to changing requirements or to domain changes
5. Usage as a description kit language or as additional descriptions to ordinary modeling. This allows the reuse of existing enterprise models.

REFERENCES


