Using Enterprise Models to Configure Service-oriented Architectures

Martin Juhrisch

Project MIRO
Westfälische Wilhelms-Universität Münster
Röntgenstr. 9 – 13
48149 Münster
juhrisch@uni-muenster.de

Abstract: The possibility of a direct link between business workflows and the supporting application system is often seen as the critical strength of the SOA paradigm. Though in practice, today's web-service technology is often used just to invoke detailed functions of particular application systems. The basic SOA does not address management and other concerns that apply to all components in an SOA. However, a real advantage of SOA will not evolve until web-services become configurable at the business level. For that kind of configuration the gap between analysis and design phase has to be bridged. Hence, we introduce a modeling grammar to model service function interfaces allowing a redesign of enterprise systems functionalities at the business level. We illustrate the usefulness of our approach with a pilot study conducted at the University of Munster (Germany).

1 Introduction

Because of the political course and increased competition today's German universities have gained significant resources for quality management in the recent past. Simultaneously a strengthened process orientation is indicated within the whole administrative scope. Due to the intention of quality improvements but also to the increasing financial pressure every university is subjected to, the focus of rationalization is more and more related to the efficiency and effectiveness of its business process structure. Prerequisites for an effective intervention into the process domain are in the first instance the entire analysis of the problem domain and its consistent documentation in the course of business process reengineering activities [HC94].

Large potential for modernizing the university administrations lies in process reorganization with information and communication technology (ICT). In 2005, four German universities created a research program to improve and standardize their administrative processes. Therefore, reams of processes were analyzed and core functions identified. As a result, automatable core functions have been implemented as web-services using the service-oriented architecture (SOA) paradigm.

In contrast to previous software architectures, SOA offers the possibility to establish a closer connection of business process flow – represented by enterprise models – and information system infrastructure – represented by web-services [Er05]. The integration of SOA services to enterprise models follows this idea trying to close the gap between IT and organizational domain.

The connection between elements with a direct business reference like business objects and elements of an information system architecture aims at the enabling of sustainable enterprise architectures. This raises the question in which way a connection between structural respectively process organization – documented in enterprise models – and a software system could be established? Classical approaches like the object oriented analysis and design (OOAD) failed with similar objectives in the past or enable an automated transformation between both domains only under ideal conditions [KG06].

Goal of this study is now to introduce a method allowing the configuration of SOA services at a business level and therewith carry out an adjustment between the target state organizational model and the application system. The configuration demands on the one hand to be able to estimate the applicability of a service for a certain business scenario and on the other hand a high flexibility of the service, which matters regarding its integration in the information system architecture. Industrial standards of the W3C concerning the service interface description cover already today all information necessary to access the web-service's value [W3C03]. A deficit exists concerning the way of establishing a business access to a web-service in order to estimate its business value and its compatibility in a business scenario. Since we cannot automatically transform an informal (respectively semi-formal) service description into a formal WSDL, an automatic test on feasibility of the service configuration in an enterprise model is hampered. If we accept this task as a creative human decision, technologies become interesting that allow a semi-automatic support. The approach presented in this paper seizes this idea and proposes an integration framework based on a catalog system.

The paper is structured as follows. The next section gives a brief theoretical introduction into the specifics of enterprise modeling and service-oriented architectures. Section 3 introduces a framework showing how enterprise and web-service models can be integrated using a web-service modeling approach embedded in the method engineering field. The paper ends with a discussion, summarizing the proposed ideas and exposing open questions regarding the realization of the application integration.

2 Background

Enterprise modeling

In the course of business process reengineering activities modeling approaches are an important technique for an effective intervention into the problem domain [HC94]. Essential benefit is attained through reduction of complexity by abstraction which facilitates analysis of complex systems ([Ba94]; [FS94]). Hence, within the scope of first stage problem analysis and documentation, visual models are established – so called enterprise models representing business requirements for the actual enterprise situation [FL03]. The semantic mightiness of enterprise modeling language constructs has to cover non-formal aspects supporting a deep understanding of the business domain as well as formal aspects in order to prepare the mapping of an organizational target state to a system implementation [Fr99]. Thus, we need semi-formal languages to model problems, which are not well-structured, highly subjective, and finally not objectively well formed [HR00].

Most grammars, however, do not possess a sufficient number of language constructs to model all phenomena in a domain [ADS99]. Consequently, we might need a multitude of modeling languages to model the dynamic changing business domains [WW02]. For process documentation purpose, e.g., we use modeling techniques like Petri-Nets [De98] or the Event-Driven Process Chain (EPC) [Sc00]. On the other hand in the course of the design of executable business process, languages like the Business Process Modeling Notation (BPMN) [Wh05] have been developed. They enable business domain experts to model understandable graphical representations of business processes with the intention to generate executable Business Process Execution Language (BPEL) artifacts [IBM02]. A BPEL file contains a XML based graph structure defining a flow of link elements to web-services. Hence, business processes can be performed by a defined flow of web-service invocations. In the course of an increasing number of enterprise modeling grammars, meta-CASE tools are needed to support language independent modeling. Software that supports language independent modeling is subsumed under the concept of meta-CASE tools like MetaEdit+ [KRT05] or Cubetto toolset [Cu07].

Service-oriented architectures

A SOA is a method to design system landscapes based on the concepts: contract, service, and interface. The SOA paradigm intends that a service is related to a semi or fully automated activity in a business process according to the terms of a contract, which determines the characteristics of the activity's implementation [DG05]. The functions contractually agreed within the service are realized by the interface of an application. As service function we define the differentiated and independently useable functions of a service, which can be used by other services [Oe05]. The SOA paradigm differs according to its service function concept from other software paradigm as a service function represents an entire business activity and its reuse is intended on system landscape level. This may be the case if a SOA has reached a stage of development where the premises for a reuse of software solutions in an organization are complied. Then we speak about service as the possibility for multi reuse of business solutions including their implementation in software [DG05]. For detailed information about organizational premises for software reuse we reference Dietzsch and Goetz (see [DG05]).

This paper focuses on a web-service based realization of the SOA paradigm according to the recommendations of the W3C consortium [Er05]. W3C's layered architecture for today's web-service technology focuses on three principle core elements described in detail by Muschamp [Mu04]. The Simple Object Access Protocol (SOAP), the Web-Service Description Language (WSDL) as well as the Universal Description, Discovery and Integration Language (UDDI) [Wa02] have become de facto standards for XML messaging, web-service description and registration. In addition to these main protocols web-service composition requires higher levels of description. In the course of this demand orchestration and choreography languages base upon several high level standards like WS-BPEL or WSFL (Web-service Flow Language) [AGP03]. Compared to the core protocols this high level languages take a step forward by integrating web-services in business process models. Essence is the integration of business processes across enterprise's boundaries by modeling web-services in directed graphs in the order of their chronological sequence.

One main problem in implementing a SOA is the appropriate definition of service function granularity. The literature does not specify a generally accepted guideline [Oe05]. The basis for this decision can be a supported business process, the derivable services and the importance of individually design objectives [Oe05]. Hence, the decision is done individually, dependent on the characteristics of the particular business process. This leads to the situation that business activities are assigned to service functions of varying size in a non-automatable decision process. To simplify the implementation of our approach we assume that a business activity can always be mapped to a granular identical service function.

3 Integration Framework

As mentioned before, the establishment of a direct linking between organizational and IT domain can only be done with the provision that we use a set of entities simultaneously in both domains. Up to now, existing approaches are not suitable to the transformation from enterprise models to software design models (see [KG06]; [Ou07]). The main problem from a modeling view is the use of language constructs originally intended exclusively for the software design [KG06]. Thus, design decisions may exert influence on the way the analysis of the organizational domain is modeled. In addition an automatic transformation demands a complete input containing all information relevant to the transformation. Though, the completeness of enterprise models cannot be subject independently assured. Furthermore, it is impossible to evaluate the correctness or significance of modeled information.

Authors proclaiming an automatable transformation assume that functions of a web-service correspond to particular business activities in a business process [IBM02]. However, this idea may be arguable if we talk about functional requirements models and not about target state organizational models (enterprise models). Functional requirements model reference the software system in a high-abstract manner. Hence, in reducing the level of abstraction we could assign design functionality to web-service functionality. This dependency implies a homomorphism between service functions of a web-service and business functions in a functional requirements model. However, this is admittedly not adaptive for our sense of integration between enterprise models and the SOA domain.

The present paper proposes an approach, which supports a decision process between business problem and IT solution but not an automatic transformation. The decision oriented approach is based upon the experiences of the software architect with adjustment between target state organizational models and reusable SOA components. Previous research is more likely to show that this decision process cannot be formalized and therewith automated, too [KG06]. Formalization would require that all variables with an impact to the dependency between solution alternatives and their consequences to the software design have been operationalized in a functional way. Enterprise models can inform business experts about the context of the information system; however they are not able to be operationalized. Hence, we propose a business configuration of SOA services in a target state organizational model using a framework that consist of a catalog system for SOA services in conjunction with a modeling tool. Out of a bulk of documented software solutions the one is chosen that suits the best in a certain business context.

Building business interfaces for web-services

At the University of Munster the semantic gap between IT and organizational domain is bridged by building up a service catalogue for all university web-services and their integration into a modeling CASE tool environment.

As every web-service implements a certain web-service interface, and since the language WSDL does not consist of appropriate language constructs for a business level configuration, we require a complementary representation of service functions respectively their signatures at business level. Every service function signature represents real business value implemented by an automated business activity. If we could create more business like service function signatures, we would be able to configure SOA functionality in target state organizational models. The question is now, how to describe a business level signature of a service function?

Business Objects are an approach to assign the object orientation paradigm not only to implementation level in software engineering but also to the level of enterprise modeling [We99]. A business objects represents simultaneously a part of the enterprise model (outer perspective) and of the system design (inner perspective). Within the scope of business objects, representations of real world entities are basically characterized by a distinct name and a description of their semantic and aim in their business domain. The linking of services causes a mapping between input and output business objects of different service functions. A semantic heterogeneity between business objects representing identical real world concepts resolves in bad mapping possibilities as each involved service has its own understanding of the business object. Hence, to reduce the semantic heterogeneity we propose the one-of definition of a business object in an SOA environment. This can be done in an enterprise service bus (ESB) [Oe05] in terms of a central architecture control. We assume that all business objects are mapped to a set of attributes and each linked service can handle this objects respectively can perform a mapping of these objects to its own. This leads to standardized sets of business object types (BOT) and attribute object types (AOT), which we can use to describe the service interface. To foster the definition of a service function interface using this business object types we introduce a certain modeling view on meta-model level (see Figure 1).

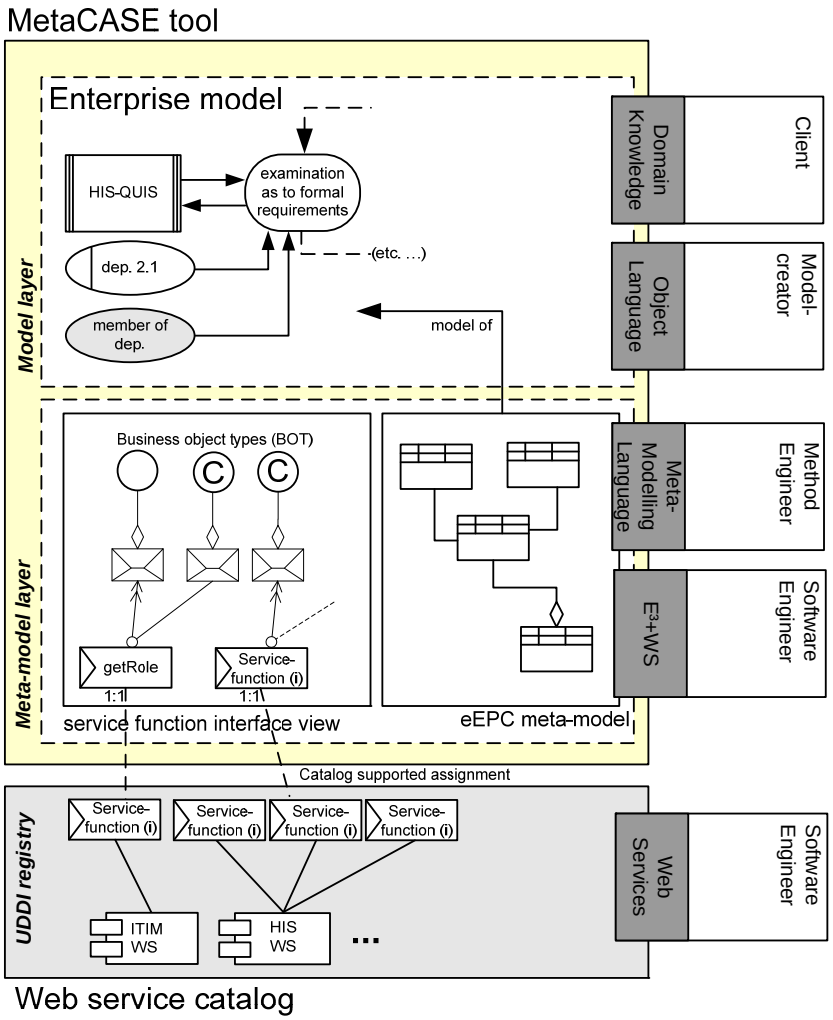

Figure 1: Integration of enterprise models with service registry

An service function interface represents one-to-one a service function signature of a web-service and therewith establish the reference between a business function at business level and its implementation in IT. Afterwards we can configure existing service function implementation in an enterprise model as we have information about what a service function does. Admittedly, until now we cannot configure service functions depending on information, how a service function processes its business function. The BOT just solve the problem of semantic heterogeneity of service functions and foster a higher degree of their university wide understanding. Semantic heterogeneity means that web-service functions implement domain specific concepts in different ways. Though, pragmatic heterogeneity references the varying understanding of an implementation of a business function [Ov04].

Modeling approach

Our modeling approach supports distributed modeling of service function interfaces and not a central modeler. Main problem in a distributed modeling approach are the use of semi-formal modeling languages, which share strength and weaknesses with natural languages. As a result of a high degree of freedom within the modeling activity, we have naturally no standardized set of modeling elements. This will lead to a lack of comparability what becomes an issue as we need to configure SOA service within a single enterprise model. Thus, to enable cross-model references we propose the use of BOT representing a limited set on business objects with a common semantic in the organizational and IT domain. Thus, the software architect is specifically restricted in his expressive power when modeling a service function interface as only the predefined set of BOT can be instantiated by the assignment of AOT.

To be language independent, we extend an existing method of the method-engineering discipline. Thereby, service interface related constructs will be integrated into a meta-modeling language. The method engineering is based upon the existing E³-Model (E³; [Gr04]) classified as meta-meta-model on M3-Level of the Meta-Object-Facility Architecture [OM02]. Abstracting from a determined notation, as a first step we define a standardized set of BOT and AOT on M3-Level. Afterwards, we extend the E³-Model containing all constructs to model a meta-model and a service function interface. In dividing the meta-model of E³+WS in separate views, we can simplify the process of construction. Secondly, we turn our attention to the construction of the meta-model. We assume that all contained constructs and relationships between them are described over the E³-Model language conventions within an eE³-View (extended E³-View). Afterwards, we generate service function interfaces within a separate service function interface view. Thus, we generate separately a modeling language that can be understood and adopted by target group users of the E³+WS method (see Figure 2).

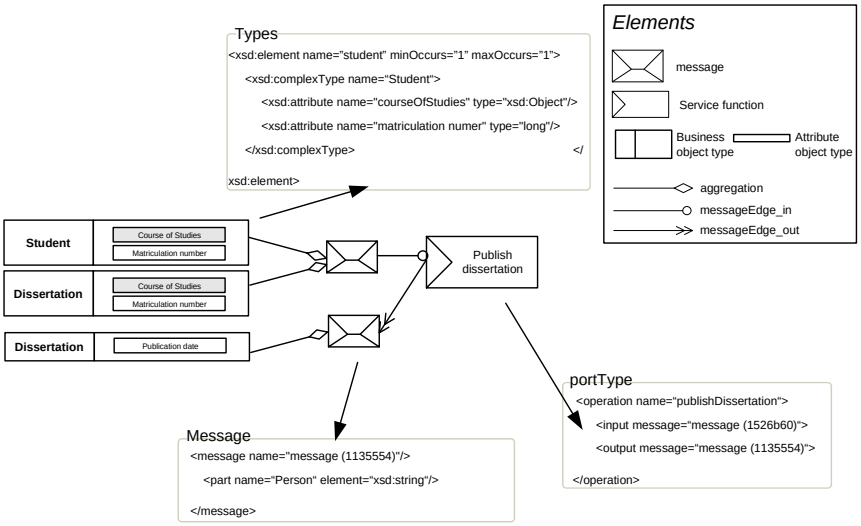

Figure 2: Service function interface modeling language

Centric element of our modeling approach is the service function. A service function interface consists of an appropriate parameter assignment. Therefore, we define special message objects that act as containers for input respectively output parameter objects. Messages are connected with service functions over so called messageEdge_in or messageEdge_out constructs depending on the role of the message regarding their related service functions. Thereby a service function can be connected to maximum one messageEdge_in and always exactly one messageEdge_out (see Figure 2). Hence, it is assumed that the possible service functions communicate either with a request-response or with a notification pattern [Wa02].

To define a single message parameter, we use the predefined BOT and AOT dividable into simple or complex objects. A BOT cannot be instantiated like ordinary object types but parameterized in the service function interface view using the predefined set of AOTs. Simple AOT are assigned to simple data types. These types comprises on the one hand predefined XML schema data types and on the other hand own data types declared within the WSDL types tag. With self declared data types we are able to define for example appropriate report formats as potential result layouts for a model migration operation. A complex AOT corresponds to a BOT respectively design activities of a method engineer. Therefore, an E³-data type is introduced to establish ties between complex AOT and meta-model patterns. In the first move, we only tie E³ object types down to complex AOT. BOT and AOT are connected over a non-directional edge. For more detailed information of the E³+WS modeling notation we reference the work of Weller et al. [WJE06].

The main requirement for this framework represents the avoidance of any restriction to the ordinary modeling task on object level. Both clients and modeling experts (model creator) should not notice the structure of a service function interface (see Figure 1). Usually the construction of enterprise models requires “a specific competence that is neither covered by software engineering nor by organization science” [Fr99] so implementation related information has to be kept away. Additionally the integration of implementation aspects after the analysis phase of the system engineering would not facilitate the adaptation process of enterprise systems functionality.

Using service function interfaces within enterprise models

To sum up, the proposed modeling approach tends to result in a set of service function interfaces. A business function can implement one or more interfaces analog to the object oriented paradigm. Using the interfaces we can configure business functions to business processes in a way that allows an automatic mapping to the SOA. Two business functions match up if the first one passes business objects with the appropriate attribute assignment to the second accordingly to their interface information. A coarse-grained modeling approach is specially qualified for the integration of service function interfaces. Due to the reduced degree of freedom during the modeling, we can sequentially configure business functions dependent on their implemented interfaces. Figure 3 illustrates a modeling method using concepts that enable a more simplified way of modeling business processes resulting in an easier understanding for domain experts.

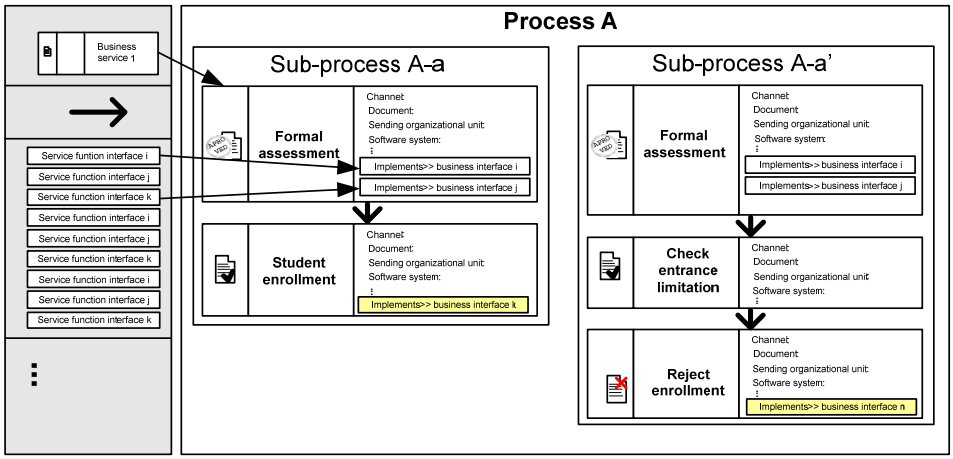

Figure 3: Integration of service function interfaces in enterprise models

We extend a modeling approach of Becker et al. with the concept of service function interfaces for web-services [Be06]. Business process behavior is carried out by a certain amount of business service blocks. This block would normally be described in detail using a modeling language like EPC in ARIS [Sc00]. However this kind of description of the component behavior would not enable an easy configuration [Be06]. A process represents sequentially modeled business services. A business service has to be instantiated through the parameterization with certain values. One value is the implemented service function interface. The modeling methods documents the business process by using predefined, university domain specific business services from a repository. The high granular description prevents the refinement of model elements and the modeling of conditional ramifications in the process flow. For a complete picture about coarse-grained modeling languages we reference to Becker et al. [Be06].

4 Discussion

At present only a few catalog implementations manage web-services functionality with an outside business perspective. The approach presented in this paper builds upon the ideas for a uniform specification of web-services and the standardization activity with UDDI. We realized a model-driven development of service function interfaces and their assignment to web-service functions in a UDDI repository. Regarding the configuration of a SOA within enterprise models, the presented modeling approach allows us to combine business functions in terms of a meaningful semantic process chain. Though, without information about the concrete implementation of a web-service function we have the problem to indicate how a certain business function is executing its task. We rely on the knowledge of the software engineer who is modeling the service function interface. For now, we cannot be sure if this process chains makes sense from a pragmatically point of view. This is part of our future research.

References

[ADS99] Agarwal, R.; De, P.; Sinha, A. P.: Comprehending object and process models: An empirical study. IEEE Trans.Software Engrg., (25:4), 1999; pp 541-556.

[AGP03] Ardissono, L.; Goy, A.; Petrone, G.: Enabling conversations with web services. In Proceedings of the second international joint conference on autonomous agents and multiagent systems, AAMAS'03, ACM Press, Melbourne, 2003; pp 819-826.

[Ba94] Balzert, H.: Development of Software-Systems: Principles, methods, languages, tools. BI, Mannheim, Germany, 1994.

[Be06] Becker, J. et. al.: Model Based Identification and Measurement of Reorganization Potential in Public Administrations – the PICTURE – Approach. The Tenth Pacific Asia Conference on Information Systems (PACIS'06), 2006.

[Cu07] Cubetto toolset, Specifications at http://www.semture.de, 2007.

[De98] Desel, J.: Petri nets and business process management. Geschäftsstelle Schloss Dagstuhl, Saarbrücken, 1998.

[DG05] Dietzsch, A.; Goetz, T.: Nutzen-orientiertes Management einer Service-orientierten Unternehmensarchitektur. In (Ferstl, O. K.; Sinz, E. J.; Eckert, S.; Isselhorst, T.): Wirtschaftsinformatik 2005, eEconomy, eGovernment, eSociety; pp 1519-1538.

[Er05] Erl, T.: Service-oriented architecture: concepts, technology, and design. Prentice Hall PTR, 2005.
[FL03] Fettke, P.; Loos, P.: Classification of reference models: a methodology and its application. Information Systems and e-Business Management, (1:1), 2003; pp 35-53.
[Fr99] Frank, U.: Conceptual Modelling as the Core of the Information System Discipline – Perspectives and Epistemological Challenges. Proceedings of the 5th America's Conference on Information Systems, AMCIS'99, AIS, Milwaukee, 1999; pp 695-697.
[FS94] Ferstl, O. K.; Sinz, E. J.: Grundlagen der Wirtschaftsinformatik. 2. Auflage, Oldenbourg, München
[FWK02] Fremantle, P.; Weerawarna, S.; Khalaf, R.: Enterprise Service – Examining the emerging field of Web Services and how it is integrated into existing enterprise infrastructures. In Communication of the ACM (45:10) 2002; pp 77-82.
[Gr04] Greiffenberg, S.: Method Engineering in Business and Government. Dr. Kovac, Hamburg, Germany, 2004.
[HC94] Hammer, M.; Champy, J.: Business Reengineering: a manifesto for business revolution. HaperBusiness, New York, 1998.
[HR00] Harel, D.; Rumpe, B.: Modeling languages: Syntax, semantics and all that stuff, part I: The basic stuff. Weizmann Institute of Science, 2000.
[IBM02] IBM: Business Process Execution Language for Web Services, Version 1.1. http://www-128.ibm.com/developerworks/library/specification/ws-bpel, 2002.
[KG06] Karow, M.; Gehlert, A.: On the Transition from Computation Independent to Platform Independent Models. In AMCIS Conference Proceedings, 2006; pp 3913-3921.
[KRT05] Kelly, S.; Rossi, M.; Tolvanen, J. P.: What is Needed in a MetaCASE Environment? In (Frank, U. Ed.) Enterprise Modelling and Information Systems Architectures, 2005; pp 22-35.
[Mu04] Muschamp, P.: An introduction to Web Services. In BT Technology Journal, (22:1), 2004; pp 9-18.
[Oe05] Oey, K. J.; Wagner, H.; Rehbach, S.; Bachmann, A.: Mehr als alter Wein in neuen Schläuchen: Eine einführende Darstellung des Konzepts der serviceorientierten Architekturen. In (Aier, S.): Unternehmensarchitekturen und Systemintegratio, GITO Verlag Berlin, 2005; pp 197-220.
[Ou07] Ouyang, C.; Dumas, M.; ter Hofstede, A. H. M., van der Aalst, W. M. P.: Pattern-based translation of BPMN process models to BPEL web services. In International Journal of Web Services Research (JWSR), 2007.
[OM02] Object Management Group: Meta Object Facility (MOF) Specification, version 1.4, 2002.
[Ov04] Overhage, S.: A Standardized Framework for the Specification of Software Components. In Proceedings of the NODe'04, Erfurt, 2004.
[Sc00] Scheer, A.-W.: ARIS – Business Process Modeling, Springer, Berlin, 2000.
[W3C03] W3C, http://www.w3.org/TR/2003/WD-ws-arch-20030808/wsa.pdf, 2003.
[Wa02] Walsh, A. E.: Uddi, Soap, and WSDL: The Web Services Specification Reference Book. Prentice Hall Professional Technical Reference, New Jersey, 2002.
[We99] Weske, M.: Business-Objekte: Konzepte, Architekturen, Standards. In Wirtschaftsinformatik, (41:4), 1999, pp 4-11.
[Wh05] White, S.: Using BPMN to Model a BPEL Process. BPTrends, (3:3), 2005; pp 1-18.
[WJE06] Weller, J.; Juhrisch, M.; Esswein, W.: Towards using visual process models to control enterprise systems functionalities. In Intl. J. Networking and Virtual Organisations, (3:4), 2006; pp 412-424.
[WW02] Wand, Y.; Weber, R.: Research Commentary: Information Systems and Conceptual Modeling – A Research Agenda. Information System Research, (13:4), 2002; pp 363-376.