Using Process Models for the Design of Service-Oriented Architectures: Methodology and E-Commerce Case Study

Oliver Thomas, Katrina Leyking
Institute for Information Systems (IWi)
at the German Research Center for Artificial Intelligence (DFKI),
Saarland University, Saarbruecken (Germany)
oliver.thomas@iwi.dfki.de, katrina.leyking@iwi.dfki.de

Florian Dreifus
SAP Inc., Walldorf (Germany)
florian.dreifus@sap.com

Abstract

This article explains the importance of business process management for service orientation and illustrates how process models can be used for the design and realization of service-oriented architectures. We will introduce a multi-level concept consisting of a design, a configuration and an execution level. The approach presented here, illustrated along the standards of EPC, BPMN, BPEL and WSDL, bridges the existing research gap between conceptual modeling and service-oriented IT support. The requirements analysis from an online-mail order company in the consumer electronics sector serves as a use case. Results show that up to now, organizational aspects have been neglected in the SOA discussion.

1. Introduction

Business processes are subject to constant changes today due to the increasing importance of flexible value-adding business networks. These process changes define new requirements for the supporting software systems and have a fundamental effect on the design of the underlying architectures. The concept of service-oriented architecture (SOA) [13; 21] takes on this challenge. SOA strives at an IT infrastructure, which automates enterprise processes by orchestrating service invocations. Thus, an SOA can react flexibly to the ever changing requirements in business environments. Service-oriented architectures follow a business process-oriented view and because of this, are often considered as process-oriented software architectures.

Knowledge about business processes and their operational and organizational structure is a critical success factor for the design and realization of an SOA. The process design must follow a comprehensive approach, which comprises planning and controlling, i.e. managing operational processes [1]. Modeling has proved helpful in supporting such a systematic approach in process design [16; 18]. Modeling methods, such as the Unified Modeling Language (UML) [3], the architecture of integrated information systems (ARIS) [29] or PROMET [26], serve as operationalized approaches for the construction of models. These methods are implemented in software tools for business process modeling, such as Microsoft VISIO, IBM Rational or ARIS Toolset, which support the design, analysis and simulation of business process models [9].

For the successful design and realization of service-oriented architectures, companies must face the central challenge of considering both conceptual, as well as technological aspects. Whereas business process modeling is established both in research and industry for documentation and business reengineering purposes in particular, the modeling of process sequences is, from a technical point of view, a comparatively young field and highly dynamic. In this context, difficulties arise due to the fact that the conceptual, as well as the technical modeling of processes are usually not coupled with each other.

The problem in information systems design lies in the very early and very late phases of modeling. The early modeling phase comprises the specification of a system from a high-level requirement and user perspective. The late modeling phase defines the components and their interaction from a technical view and is often closely connected with the development environment used for the implementation. Due to the existing gap between the conceptual and the technical model, a consistent transfer of conceptual requirements

1530-1605/08 $25.00 © 2008 IEEE

into supportive IT systems cannot be guaranteed. Hence, the problem is mainly caused by a lack of continuity between the conceptual and the technical level of modeling ("top-down problem"). This difficulty is worsened by the fact that most changes are carried out ad hoc on the technical systems without affecting the documentation of the conceptual processes leading to a line-up in which the actual processes supported by IT systems in the company and the documented, conceptual processes drift apart. The problem here is, once again, a lack of continuity in modeling, here however from the technical to the conceptual level ("bottom-up problem"). The lack of such a mutual continuity in modeling is addressed by the approach introduced here. In the context of the service-oriented structuring of information systems currently being discussed, this article will focus on the model-based orchestration of existing services based on a conceptual model.

The article is organized in the following manner: Chapter 2 presents related approaches, which tackle parts of the problem areas described, but however do not provide a coherent solution. In Chapter 3 we propose a framework, which provides a continuous methodology for designing an SOA based on a 3-level modeling concept. This generic approach is instantiated in Chapter 4 by an E-Commerce scenario. Each step of the methodology is illustrated on the case of an online-mail order company, deriving an SOA design from the business process models involved. In conclusion, Chapter 5 summarizes the critical success factors for the model-based design of an SOA.

2. Related Work

As is characteristic for topics related to information systems sciences, the approach presented here links interdisciplinary fields between service-oriented software architectures, model-based software engineering and business process management (BPM). Although relevant research has been pursued in each of these areas, a comprehensive approach is still missing.

Web Services in General: while the concept of a service is not new to software engineering, it has gained new momentum by the commoditization of internet technology which brings web services with it. W3C [2] and [15] define Web Services as software applications that can interact with each other over XML-based interfaces bound to URIs using XML-based messaging and internet protocols. Thus, web services are considered as a means for exposing functionalities of information systems and delivering them through web technology standards [7]. Applying the benefits of service-oriented computing to enterprises yields the notion of enterprise services, i.e. web services, which execute business transactions within and across enterprise systems [17]. To realize this cross-organizational vision, a multitude of industry initiatives have been started for developing XML-based standards on how to describe, discover, and compose web services [5; 6; 12]. The notion of web services is further advanced by Chesbrough [11] by being merged with other related, yet isolated fields to a service discipline. One of the author's challenges is to explore how information on the capabilities of artifacts is to be managed and (re-) composed to create value. However, this aspect of (business) value creation is still lacking in practical methods of SOA engineering.

Web Services and SOA: despite a long history of service-orientation for distributed computing [7], the term SOA is mostly used in the ongoing scientific discussion synonymously for service-oriented software architectures based on web service technology. Nevertheless and despite a lenient usage by some authors, an SOA is more than simply a pool of web services capable of interacting with each other [21]. Rather, web service technology is seen as a vehicle to facilitate the basic principles of SOA – the interoperability of functions offered by different organizational units in electronic business scenarios [19]. Beyond the questions about the design, description and execution of services, there is quite some work being done on the middleware infrastructure of an SOA, called Enterprise Service Bus [10]. This core component of an SOA is responsible for the execution and interaction of web services along specific processes. Although there is some related work on web service integration as a means of enterprise application integration (EAI), the existing concepts remain reserved to IT specialists instead of business managers who know best about their processes.

Web Services and BPM: given the decisive characteristic of composition, the design of an SOA must not only cover the specification of web services, but also their flexible composition to business processes [22; 33]. The field of BPM striving for process-driven application integration has recognized this chance. Zhao [34] and Moitra [25] envision the advancement of web services as a universal computing platform, which provides enterprise services to be flexible composed to value-creating business processes. Thus, earlier ideas expressed by Verner [31] concerning a closed loop from process design, to process development and process deployment, feeding information back to (re-)design are becoming reality. However, existing work such as the order management scenario by Zimmermann [35] is limited to a purely technical perspective on processes neglecting the

decisive aspects of business relevancy. Also, the instrument of process simulation – enabled by codified service processes – remains on an IT level, as described by Tewoldeberhan [30].

Model-Driven Design of SOA: the idea of model-driven SOA design is not completely unheard-of. Baresi et al. [8] use static and dynamic UML models to decide about the aptitude of certain middleware platforms for business information platforms. Their approach shows some convergence between modeling and SOA, but concentrates solely on the validation of SOA and not on the specific needs in designing an SOA. Another approach [14] examines how the concept of "Programming in the Large" can be practically applied to SOA development based on process models. They use BPMN as a process programming language, which can be mapped to the Business Process Execution Language (BPEL). As both process languages are led by IT-relevant aspects, they are hardly relevant for business analysts designing an SOA in terms of business needs. On the level of technical integration, the synergy effects between process modeling and SOA have also been recognized by approaches related to the concept of Model Driven Architecture. Pfadenhauer et al. [28; 27] have already proposed a model-driven service architecture, but did not integrate pure business requirements formulated by non-technical managers or domain experts. In contrast to this prevailing technical understanding, Wenhui [32] confirms that the true potential of a process-driven SOA lies in innovative business models facilitated by web service infrastructure.

As a summary, there is a link missing between the moderate innovations an SOA brings to IT integration and the unprecedented potentials to be leveraged for business integration in all existing approaches. None of the related contributions examined acknowledges the fact that business process modeling is established in companies by means of semi-formal diagrams that are easy to understand and handle by non-technicians. Our approach is to use this treasure of existing models created in long business process (re-)engineering (BPE/BPR) projects, representing both as-is and to-be business processes.

3. Research Methodology

Conceptual process models contain the potential business-relevant context for each of the tasks, functions or activities defined in the process organization. These can be participating organizational units, costs or legal restraints. Technical process descriptions however, contain the detailed information needed for the IT support of these processes including for example, the services involved, required input and output data or the interface information needed for the integration of external partners. The basic idea of the integration of conceptual and technical process descriptions is to incorporate the process logic shared by both levels of modeling into a third level of process description utilizing it for the integration and synchronization of the conceptual and technical description level. This triple layer design can be seen from the perspective of the underlying processes, the constructed models and the languages used (Figure 1).

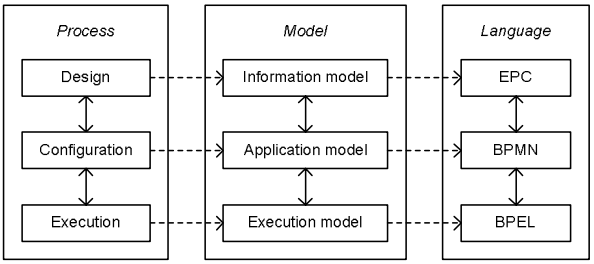

Figure 1: Framework for the model-based design of service-oriented architectures

Based on the underlying processes one can differentiate between a design level, a configuration level and an execution level. On the design level, information models can be used to deal with the complexity connected with the conceptual description and requirements analysis. They can be constructed with a semi-formal graphical language such as the event-driven process chain (EPC) [20]. On the configuration level, the complexity connected with the adaptation of information systems can be reduced with application models, in order to configure information systems model-based. The Business Process Modeling Notation (BPMN) [4] is a language suited to this. Thus, models can be constructed containing the process logic of the information model level, as well as technical details of the execution model level by using attributes. On the process execution level, models contain the information needed for their technical execution, which can then be expressed by languages such as the Business Process Execution Language (BPEL) [6].

The vertical connections between the levels in Figure 1 emphasize the importance of the bi-directional coupling of information models with execution models via application models. The horizontal links symbolize loose associations between elements of the same proximity to information technology in different fields of the framework. While the area on the left contains the major phases of information system engineering,

the middle area shows model types used by each of these phases. By providing suitable languages to explicate these abstractions on the right hand side, the framework is completed. In the following, the integration of conceptual and technical process models will be illustrated based on a case study. The languages used here are EPC, as well as the standards BPMN, BPEL and WSDL.

4. Process Models for the Design and Implementation of SOA

4.1. Case scenario

The entert@inment GmbH is an online-mail order company, which focuses on consumer electronic products. Products are ordered either per telephone or over the Internet. The ERP software used for the support of the company's business processes was created almost completely by the company itself, with the exception of the CRM system used in the call center. High growth rates of 70 to 80 % have caused many changes in the company and its IT infrastructure in the past years. At the beginning of 2006, the management initiated the modeling project "Implementation of a Process Documentation" (Length: 6 months, Effort: 1 person year). The objectives of the project were to collect, model and evaluate existing business processes in selected areas, in order to provide a basis for process optimization, as well as a continuous BPM. The activities were supposed to lead to an increase in effectivity and efficiency. In accordance with the concept in Section 2, the project's intention was not primarily aimed at supplementary implementation goals, but rather strategic and design goals. On the strategic level, transparency in the processes and responsibilities on the aggregated level were to be achieved through the systematic and concise representation of core processes. Based on this, an optimization of the processes within the framework of coordinated projects was encouraged on the design level. In this context, the project was to achieve transparency in processes and responsibilities on a detailed level. A requirements analysis carried out within the framework of the documentation project showed the following improvement potentials:

1. The company was struggling with a large number of unserious incoming orders. These so-called "junk orders" contained misstatements in terms of names and addresses or unusually large order sizes. Because regulating such junk orders is extremely time consuming, the automated comparison of orders with an address database was seen as a possible solution.
2. Up to now, the creditworthiness of a customer was checked only via an electronic inquiry to the customer's bank when the customer selected the corresponding method of payment. To allow improved credit assessments in the future, especially for orders on account, internally available evaluation rules and data were to be enriched with externally available data. The combination of statistical methods, such as external credit information databases, updated regularly within certain periods, with dynamic methods, such as the SCHUFA information, which makes up-to-date data available, was planned. The credit assessment process was to implement with the highest possible flexibility, in order to react quickly to changes in calculation rules or upcoming compliance requirements. Credit assessment was to additionally be supplemented by a Q-Bit-Identity Check (SCHUFA service, which confirms the majority of a customer during the order process).
3. Stock was still checked manually in the company. Furthermore, an automated availability check for products to be delivered later by the manufacturer was not yet implemented, resulting in inexact delivery prognoses for the customer. The increase in customer dissatisfaction lead to complaint calls and increased call center costs. To deal with this problem, the management called for the integration of real-time data in delivery prognoses.

Due to the improvement potential recognized, the management added a new objective to the previous strategic and creative intentions followed up to this point: the flexible automation of processes for the reduction of processing times and increase in process quality through the continual design of an SOA. The SOA concept is ideally suited to the realization of the described process changes for the following reasons:

1. The additional functionalities can be realized by a number of independent, loosely coupled services. These have, in part, already been realized and can be used via a standardized interface (e.g. SCDI interface or XML Gateway from the SCHUFA as a "service provider").
2. The newly planned IT infrastructure should be consequently geared to business processes, which will be assured by the orchestration of services.
3. Services can be realized in different programming languages and on different system platforms. Furthermore, they facilitate the seamless integration of already existing functionalities from older systems.

4.2. Process Design

The EPC model shown in Figure 2 illustrates the situation after the completion of the modeling project at the entert@inment GmbH. The model illustrates the potentials listed beforehand for the definition and execution of checking functions for customer orders. The check refers to order data, customer creditworthiness and product availability. Negative results such as: "Order data is incorrect" or "Customer is not creditworthy" lead to the rejection of the customer order through the function "Reject customer order". These functions should be automated by services within the framework of the SOA initiative of the entert@inment GmbH. To identify the respective services for the support of the individual process steps, the conceptual activities were also described by input and output data. This extension was implemented in Figure 2, on the left, with organizational units, information objects and input and output performance [29]. Correspondingly, one also differentiates between organization flows, data flows and performance flows respectively. In modeling tools, these objects can be enriched by attributes whose value (based on a certain order) is then processed in the verify functions. One example is the customer data (for example: age, address), which enters the function "Define customer order" over the information object "Customer".

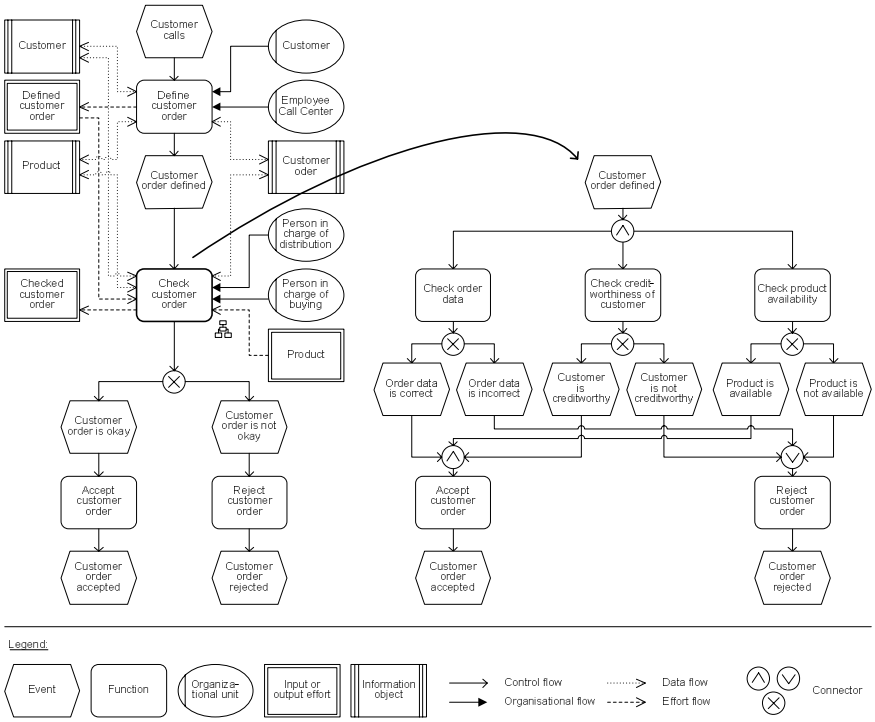

Figure 2: EPC model for sales order checks

4.3. Process Configuration

In this section the transformation of the conceptual process model into a conceptual-technical application model will be described. The Business Process Modeling Notation (BPMN) was used here as a modeling language. It was developed by Stephen A. White from IBM and accepted as a specification by the OMG in February of 2006 [4]. The aim of its designers was to achieve high acceptance by business experts through the easily understandable graphical notation.

The starting point of the transformation is the existing process logic in an EPC model. This is carried over to the BPMN model using suitable language constructs from the BPMN. With it, the basic structure for the sequence of a process is defined. While this basic structure is complemented in the EPC through the annotation of organizationally relevant aspects in the form of further model elements, a supplementation of its basic process-logical framework in the BPMN model takes place using technical details for the process execution. Conceptual information about a process, thus remains in the information model, while its basic process-logical framework is the starting point for enrichment with execution-relevant technical information in the application model.

In the following, we will describe how the EPC model in Figure 2 can be transformed into a BPMN model. Figure 3 shows the resulting BPMN model.

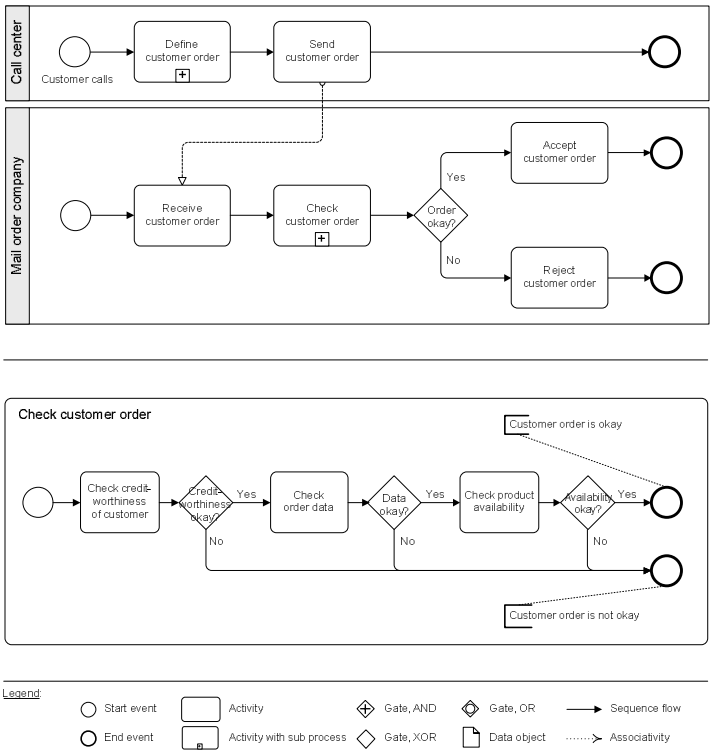

Figure 3: BPMN model for sales order checks

The process described in the EPC model contains steps that can be allocated to two different organizational units and potentially, different business partners: the call center and the mail order company. In the BPMN, partner processes can each be allocated to their own pool for the clarification of this fact. This pool can then be subdivided up into lanes in accordance with the business partner's internal organization structure. The representation of an interaction between different pools is, in comparison to lanes, only possible via a message flow.

The process described in the EPC starts with a customer's call to the call center. The customer order is defined. This is represented in the BPMN model by a corresponding start event and an activity within the call center pool. Then the customer order is checked in the EPC model. This process is represented in BPMN by a message flow from the pool of the call center to the pool of the mail order company. One should note that the control flow in the EPC model describes the logical course of a process from a conceptual view, while the sequence flow in the BPMN model defines the order of the activities to be executed. To allow the mail order company to process the customer order defined in the call center, another message flow must be established between the two business partners in the BPMN model, in conjunction with the respective activities "Send customer order" and "Receive customer order".

Further customer order processing now takes place in the pool referred to as "Mail order company". After a customer order is received using the activity "Receive customer order", it is checked by the next activity in the sequence flow. Because the activity "Check customer order" consists of several steps, it can be detailed as a sub-process, similar to the EPC model. In contrast to the EPC, the BPMN differentiates between three types of sub-processes: "embedded", "independent" and "reference". Because the intended check only makes sense in the context of customer order processing, an embedded sub-process is taken for a fine granular description. The embedded process is emblematized by a "+" in the activity symbol. A decision concerning the further course of the process is now dependent on the result from the customer order check. This circumstance is illustrated by a data gate of the type XOR, which decides on the progression of the process according to the data made available by the previous activity. While the condition is noted in the gate itself, possible alternatives are noted on the outgoing sequence flow edges. Analogue to the EPC,

the process is either continued with the activity "Accept customer order" or "Reject customer order" and then ends.

The sub-process contained in the activity "Check customer order" is shown in the lower part of Figure 3; an embedding takes place directly through the enlargement of the relevant activity symbol of the superior process, which is not represented in the figure due to shortage of space. In comparison to the EPC, the sub-process generates the same results – a customer order is accepted when all three conditions are fulfilled. Because the order check described here is a fully automatic process, which in addition, is not time-critical because the call center was given the order after telephone contact with the customer, the internal flow logic has been modified so that the verifications occur sequentially and not – as in the EPC model – parallel. This allows a more simple design and generation of the executable BPEL process respectively. Through the internal configuration of the gates in the sequence flow of the sub-process, tokens reach the end of the sub-process, which leads to order acceptance or rejection.

After the EPC is transformed into a BPMN model, the model must be improved further in the sense of its technical executability. If there are no services available on the granularity level of the activities defined in the model already, then they can be improved further with sub-processes until a sufficient decomposition for the execution with web services is achieved. For the execution with web services, the activities in the BPMN diagram can be enhanced with required attributes, such as the types of incoming and outgoing messages. These are not visible in the graphic model and at the same time, form a bridge between the configuration on the application model level and the execution level.

All things considered, the transformation of an information model into an application model, illustrated here with the languages EPC and BPMN, is a creative process, since the concepts on the conceptual level such as strategies, resources, organizational units, products, etc. must be represented on the technical level, affecting elements such as services, data, interfaces, transactions etc. The reuse of the knowledge necessary for this and the securing of the systematic application of established solutions and best practices could occur over reference models and patterns during the model construction, as well as on the information and application model level.

4.4. Process Execution

As discussed above, the BPMN serves both as a graphic representation of the control flow and a textual attribution of the process elements. To transform this semi-formal intermediate result into a code-based, executable BPEL model, the BPMN specification provides design rules, which generate a formal BPEL process model from the BPMN process model on the execution level. The Business Process Execution Language (BPEL) is already considered by many as the de-facto-standard for business process implementation based on web services. It is an XML-based language whose specification was initiated by IBM, BEA and Microsoft and is currently being developed further by the OASIS standardization initiative in the Version 2.0. BPEL builds on the Web Service Description Language (WSDL) by combining the web services described in WSDL to a process [6].

Based on XML, the Web Service Description Language has been published in the Version 1.1 in 2001 by the World Wide Web Consortium (W3C) in participation with SAP, Microsoft and IBM. It is used to describe the available data, data types, functions and communication protocols of a web service. Through the platform, protocol and programming language-independent description of web services, WSDL abstracts from the underlying IT infrastructure.

A BPEL process is generally seen as a number of service calls subject to a logical and chronological order. Single services are orchestrated or composed to become a process [7]. Thus, a BPEL process model is the IT counterpart to the conceptual business process model. The goal is to invoke the previously selected services, according to the business process based on the BPEL process and thus, execute the process in business reality. In doing so, the transformation of the BPMN application system model into an executable BPEL process represents the most technical task. While both of the previous modeling steps still took place on a graphical level and were carried out by technically oriented business analysts, as well as system architects, the coding of the graphic application model into a textual execution model is an engineering task.

Figure 4: BPEL process and WSDL description

The transformation rules defined in the BPMN specification are the starting point for this. They assign each BPMN element and attribute to a corresponding BPEL representation. The invisible attributes in conjunction with the graphically represented process sequence are part of the required data basis. In certain situations, it may be necessary to adjust the BPMN process design according to execution constraints required by BPEL in compliance with the conceptual determining factors. As described above, within the case study of the entert@inment GmbH it was advisable to change from the parallel procession of check activities to a consecutive course. In addition, a BPEL process created in this manner will exhibit non-conceptual information gaps, which are caused by the semi-formal definition of the BPMN and get in the way of process executability. The task of the IT developer is to fill these gaps and to detail vaguely formulated information respectively (for example: specify partner links or express data manipulation).

The deterministic rule base allows the IT-supported automation of the BPMN/BPEL transformation. Accordingly, IT-supported BPEL tools can decisively increase the efficiency of the procedure in implementation projects, in which executable process description languages such as BPEL are used. Almost all of the large software manufacturers offer such products or are currently working on BPEL support. On the one hand, graphic modeling tools, such as SemTalk for Microsoft Visio or Intalio, enable the transformation of a BPMN model into BPEL process code; On the other hand, integration platforms such as the BizTalk Server (Microsoft), embedded in the development environment Visual Studio, allow the import, creation, processing and export of service orchestrations. These tools provide process environments and support the management of process instances. An example for such a "BPEL engine" in the open-source environment is the ActiveBPEL (Active Endpoints, Inc.).

Figure 4 represents the process "Create customer order" specified in the BPMN model on the company side in BPEL code and the service description `CreateSalesOrder` described in WSDL. The schematically represented BPEL process expresses the control flow of the selected example and consists mainly of a sequence of service calls, which are split by the switch construct. This switch is equivalent to the exclusive BPMN gateway. It divides the control flow up into two alternatives, which are dependent on the

result of the previous order verification. This is returned from the preceding service call in the variables `OrderData`, `CustomerRating` and `Availability`. If one of these checking results is negative, the customer order is rejected by calling the service operation `RejectCustomerOrder`. If the check is positive three times then the customer order is accepted by calling `CreateSalesOrder`.

To show the transformation process with a practical service example and illustrate the potentials of ERP systems, the service "Create Sales Order" was selected. This is part of the process component "Sales Order Process", which is published as an ERP element within the framework of the Enterprise Service Workplaces on the SAP Developer Network Homepage (URL http://www.sdn.sap.com/). The WSDL description of this `CreateSales Order` service is shown on the right-hand side in Figure 4 and comprises the definition of the necessary XML-based messages (`messages`) and the combination of the service from `Porttype` and `Operation`. The latter is defined over the `Input` and `Outputmessages`, as well as a `default message`.

It becomes apparent in the discussed context that the sub-process of the BPMN model was integrated into the BPEL process. The alternative of implementing the activity "Check customer order" by a so-called composite service, i.e. an independently callable BPEL process conflicts with the flexible adaptability of the check activities to be executed in this sub-process. Instead, any possible check service can be inserted, exchanged or deleted in the process logic implemented in the BPEL process.

5. Conclusion and Outlook

Many IT solution providers see a service-oriented approach as the architecture paradigm of the future. For instance, the analysts Gartner have forecasted that until 2010 approximately 65% of the large companies will base over 35% of their application portfolio on an SOA architecture [23]. Euphoria aside, it must be emphasized that service orientation does not represent an isolated software product, but rather a long-term, strategic adjustment of the company-wide IT architecture. The respective change processes are characterized by creativity and are highly complex due to their diversity and dependency on human judgment. The generalized procedure for the model-based design of SOA introduced here can be used to deal with this complexity. The following success factors are critical for its implementation and thus, for a model-based SOA implementation project:

1. *Process knowledge:* The discussion about process-oriented organization has considerably influenced the development and structure of information systems. In order to realize efficient, IT supported inter-organizational process integration, the software architecture must be extended to "orchestrate" the relevant corporate activities. With the goal of customer-oriented business process design, the focus must be put on bringing together the various core competences distributed in the network according to the current requirements. The interdisciplinary knowledge about company processes thus, forms the foundation for the identification and development of supportive services, as well as for the orchestration of these.
2. *Process documentation:* In addition to the knowledge about the processes, their documentation is also important, especially with regards to the approach taken in this article. In addition to the textual description of a company's reality, graphical process models should also be available. This applies to not only the planning and realization of an SOA. A new study from Gartner Research has shown that the documentation of conceptual processes with their processing times and respective responsibilities leads to an increase in productivity of more than 12% [24].
3. *Process quality:* An IT developer will however, use a process model created by a business department for the implementation of an SOA only if the quality of the resulting technical models – and with it the software to be developed – is noticeably improved from his point of view. The quality of the conceptual process models has therefore a lasting and significant effect on the model-based design of an SOA.
4. *Process communication:* Process models are used to mediate between business and domain experts with conceptual knowledge and those with methodological or technical knowledge, such as IT consultants or engineers. This applies for SOA projects in two ways. On the one hand, process-oriented execution languages such as BPEL come to the fore due to standardization initiatives. On the other hand, the importance of conceptual process description languages such as EPC increases due to the targeted orchestration of services.

In order to fully support the integration of business process models, technical service models and actual software code, future SOA development environments will need to provide one integrated repository shared by all levels of modeling. The mapping of objects between information, application and execution models is stored there, so that work actually done on one of the three levels can be propagated and synchronized with the others in real-time. This vision is will require a

common process ontology specifying generic modeling artifacts and, in the end, a standardized vocabulary bridging business requirements and IT constraints.

6. References

[1] Becker, J.; Kugeler, M.; Rosemann, M. (eds.): *Process Management: A Guide for the Design of Business Processes.* Berlin: Springer, 2003

[2] Booth, D. et al. (eds.): *Web Services Architecture: W3C Working Group Note 11 February 2004.* W3C, 2004

[3] OMG (ed.): *Unified Modeling Language: Superstructure, version 2.0, formal/05–07–04.* Needham: OMG, 2005

[4] OMG (ed.): *Business Process Modeling Notation Specification: Final Adopted Specification dtc/06–02–01.* Needham: OMG, 2006

[5] Clement, L. et al. (eds.): *UDDI Version 3.0.2: UDDI Spec Technical Committee Draft, Dated 20041019.* Billerica: OASIS, 2006

[6] Alves, A. et al. (eds.): *Web Services Business Process Execution Language Version 2.0: Committee Draft, 17th May, 2006.* Billerica: OASIS, 2006

[7] Alonso, G. et al.: *Web services: concepts, architectures and applications.* Berlin: Springer, 2004

[8] Baresi, L. et al.: Modeling and validation of service-oriented architectures: application vs. style. In: *Proc. of the 9th European software engineering conference held jointly with 11th ACM SIGSOFT intern. Symposium on Foundations of software engineering.* Helsinki: ACM, 2003, pp. 68–77

[9] Blechar, M. J.; Sinur, J.: *Magic Quadrant for Business Process Analysis Tools, 2006.* Stamford: Gartner, 2006

[10] Chappell, D. A.: *Enterprise Service Bus.* Beijing: O'Reilly, 2004

[11] Chesbrough, H.; Spohrer, J.: A research manifesto for services science. *CACM* 49 (2006) 7, pp. 35–40

[12] Christensen, E. et al.: *Web Services Description Language 1.1: W3C Note 15 March 2001.* W3C, 2001

[13] Dostal, W. et al.: *Service-orientierte Architekturen mit Web Services: Konzepte – Standards – Praxis.* Heidelberg: Elsevier, 2005 (in German)

[14] Emig, C.; Weisser, J.; Abeck, S.: Development of SOA-Based Software Systems – an Evolutionary Programming Approach. In: *Proc. of the Advanced Int'l Conference on Telecommunications and Int'l Conference on Internet and Web Applications and Services.* IEEE, 2006, pp. 182–188

[15] Ferris, C.; Farrell, J.: What are Web services? *CACM* 46 (2003) 6, p. 31

[16] Frankel, D. S.: *Model driven architecture: applying MDA to enterprise computing.* Indianapolis: Wiley, 2003

[17] Fremantle, P.; Weerawarana, S.; Khalaf, R.: Enterprise services. *CACM* 45 (2002) 10, pp. 77–82

[18] Hailpern, B.; Tarr, P.: Model-driven development: The good, the bad, and the ugly. *IBM Systems Journal* 45 (2006) 3, pp. 451–461

[19] Hündling, J.; Weske, M.: Web Services: Foundation and Composition. *Electronic Markets* 13 (2003) 2, pp. 108–119

[20] Keller, G.; Nüttgens, M.; Scheer, A.-W.: Semantische Prozeßmodellierung auf der Grundlage Ereignisgesteuerter Prozeßketten (EPK). In: Scheer, A.-W. (ed.): *Veröffentlichungen des Instituts für Wirtschaftsinformatik*, No. 89, Saarland University, 1992 (in German)

[21] Krafzig, D.; Banke, K.; Slama, D.: *Enterprise SOA: service-oriented architecture best practices.* Upper Saddle River: Prentice Hall, 2006

[22] Leymann, F.; Roller, D.; Schmidt, M. T.: Web services and business process management. *IBM Systems Journal* 41 (2002) 2, pp. 198–211

[23] Malinverno, P.: *Service-Oriented Architecture Craves Governance.* Stamford: Gartner, 2006

[24] Melenovsky, M. J.: *Business Process Management's Success Hinges on Business-Led Initiatives.* Stamford: Gartner, 2005

[25] Moitra, D.; Ganesh, J.: Web services and flexible business processes: towards the adaptive enterprise. *Information and Management* 42 (2005) 7, pp. 921–933

[26] Österle, H.; Blessing, D.: Ansätze des Business Engineering. *HMD – Praxis der Wirtschaftsinformatik* 41 (2005) 241, pp. 7–17 (in German)

[27] Pfadenhauer, K.; Dustdar, S.; Kittl, B.: Challenges and Solutions for Model Driven Web Service Composition. In: *Proc. of the 14th IEEE Intern. Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise.* IEEE, 2005, pp. 126–134

[28] Pfadenhauer, K.; Dustdar, S.; Kittl, B.: Comparison of Two Distinctive Model Driven Web Service Orchestration Proposals. In: *Proc. of the 7th IEEE Intern. Conference on E-Commerce Technology Workshops.* IEEE, 2005, pp. 29–36

[29] Scheer, A.-W.: *ARIS – business process modeling.* 2nd ed. Berlin: Springer, 1999

[30] Tewoldeberhan, T. W.; Verbraeck, A.; Msanjila, S.: Simulating process orchestrations in business networks: a case using BPEL4WS. In: *Proc. of the 7th intern. conference on Electronic commerce.* Xi'an: ACM, 2005, pp. 471–477

[31] Verner, L.: BPM: The Promise and the Challenge. *Queue* 2 (2004) 1, pp. 82–91

[32] Wenhui, S. et al.: Develop a telecommunication service system using service-oriented architecture. In: *Proc. of the IEEE Intern. Conference on e-Business Engineering.* IEEE, 2006, pp. 674–677

[33] Zeng, L. et al.: Flexible Composition of Enterprise Web Services. *Electronic Markets* 13 (2003) 2, pp. 141–152

[34] Zhao, J. L.; Cheng, H. K.: Editorial: web services and process management: a union of convenience or a new area of research? *Decision Support Systems* 40 (2005) 1, pp. 1–8

[35] Zimmermann, O. et al.: Service-oriented architecture and business process choreography in an order management scenario: rationale, concepts, lessons learned. In: *Companion to the 20th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications.* San Diego: ACM, 2005, pp. 301–312