Workflow Modeling for WS-BPEL-based Service Orchestration in SMEs

S. Gudenkauf, G. Scherp, W. Hasselbring
OFFIS Institute for Information Technology
Technology Cluster EAI
Escherweg 2, 26121 Oldenburg, Germany
email: [stefan.gudenkauf, guido.scherp]@offis.de,
wha@informatik.uni-kiel.de

A. Höing, O. Kao
Technische Universität Berlin
Complex and Distributed IT Systems
Einsteinufer 17, 10587 Berlin, Germany
email: [andre.hoeing, odej.kao]@tu-berlin.de

Abstract: The BIS-Grid project[1], a project in the context of the German D-Grid initiative, investigates Grid and Cloud service orchestration for information systems integration, especially when crossing enterprise boundaries. Small and medium enterprises shall be enabled to integrate heterogeneous business information systems and to use external resources and services with affordable effort. In this paper, we present the workflow modeling method that we developed within the BIS-Grid project.

1 Introduction

The integration of heterogeneous information systems is crucial in order to map business processes to the technical system level. To do so, integration is often achieved by service orchestration in service-oriented architectures (SOA). Web services are commonly used to create SOA since they enable service orchestration and hide the underlying technical infrastructure. SOA and Web service technologies are also the basic technologies for the newly emerging Cloud computing paradigm. Cloud computing provides easy access to IT infrastructures, computing platforms, or complete applications. These characteristics are also referred to as Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS). In the BIS-Grid project[2], we focus on the integration of enterprise applications (Enterprise Application Integration, EAI) using Grid and Cloud service technologies. Our major objective is to proof that these technologies are feasible for information systems integration for small and medium enterprises (SMEs), especially

[1] This work is supported by the German Federal Ministry of Education and Research (BMBF) under grant No. 01IG07005 as part of the D-Grid initiative.

[2] http://www.bisgrid.de

when traversing enterprise boundaries. SMEs shall be enabled to integrate heterogeneous business information systems and to use external resources and services with affordable effort. To do so, we proposed Orchestration as a Service (OaaS) as the primary infrastructure paradigm – a specialization of PaaS where a service orchestration engine is hosted in a Cloud environment, directly to be maintained by the OaaS provider – and provided an adequate Oaas infrastructure, see [GHH+09, HSG+09].

However, before even regarding the utilization of externally provided Grid and Cloud services, SOA can be regarded as the main requirement for SMEs, both from an organizational and from a technical perspective (cf. [BGJ+09]). While the organizational perspective requires SMEs to structurally align with an underlying SOA infrastructure and to define the different layers of service provision, the technical perspective requires to formalize the business processes, to separate the parts of the processes that can or shall be automatized from those that are subject to human activity, and to define the basic services, for example. Within the BIS-Grid project, we evaluate the OaaS approach and our OaaS infrastructure in two business scenarios motivated by our industrial project partners, a services partner for first-class trade brands on the European photographic market supplying stores and internet retailers with photographic products, and a global market leader in the field of wire drawing and draw-peeling for the automotive industry. Both have strong needs for service orchestration, the first one to integrate enterprise data for unified access for call center agents, and the latter to improve access to, and retrieval and maintenance of product and project data across different sites of the company group. However, the industrial partners are SMEs that initially did not have extensive practice and experience in workflow technologies and SOA as well as Grid and Cloud Computing. While the lack of expertise in setting up and maintaining a service-oriented computing infrastructure was compensated by the OaaS scenario in which the technical partners of the project act as service providers for an orchestration engine, process orientation and workflow modeling still remained as requirements for the SMEs. For the pragmatically definition of workflows for such SMEs, a method was needed that is simple to follow, general, and effective. This method is presented in the next section.

2 Workflow Modeling Method

Within our application scenarios, we developed and employed a top-down workflow modeling method that is small enough to target service orchestration in prototype SOA scenarios. Figure 1 presents an overview of this method. The upper half of the figure shows the creative activities of the method, and the lower shows the involved infrastructure components. Associated business roles are annotated to both the activities and the components, and the arrows informally depict the main dependencies between the respective activities and components. The design of a workflow mainly depends on a process model that is delivered by *business analysts* in the form of Business Process Modeling Notation diagrams (http://www.bpmn.org/) as a high-level abstraction to describe the business process, and on services that are provided by *service developers* – either by identifying existing ones or by genuine implementation. There is an information exchange between the individual roles, for example, to gain a common view of the global data model. To provide an executable workflow, the workflow must be deployed on an adequate workflow engine (here, the BIS-Grid Workflow Engine) and, of course, all used services must be made available at their respective service execution environments.

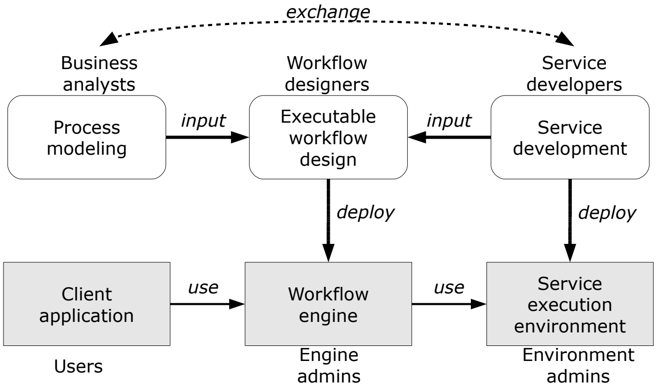


Figure 1: Overview of the workflow modeling method.

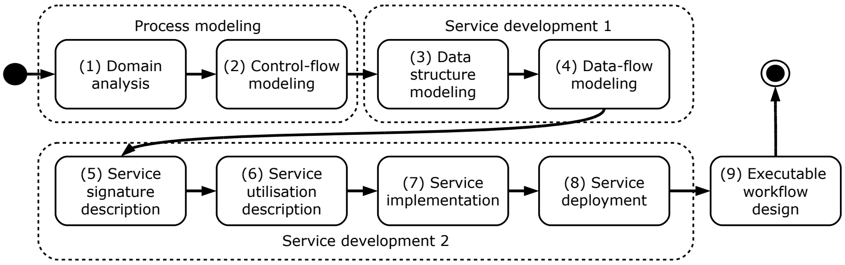

Process modeling
Service development 1
(1) Domain analysis
(2) Control-flow modeling
(3) Data structure modeling
(4) Data-flow modeling
(5) Service signature description
(6) Service utilisation description
(7) Service implementation
(8) Service deployment
(9) Executable workflow design
Service development 2

Figure 2: Applied workflow development process.

Figure 3 presents an UML object diagram that gives a more specific overview of the individual artifacts of the method and the conceptual background. It represents a method model that focuses on the *RepresentationOf* (μ) and *DecomposedIn* (δ) relations between the objects, while the objects themselves represent systems. While the μ relations identify a given system as the *model* of another system that is called system under study (*sus*), the δ relation simply identifies a system as a *part* of a *composite* system. We find that such $\mu\delta$ graphs provide a good overview on the elements of a method, ranging from the conceptual problem decomposition behind the method (i.e., the abstract systems, AS), to the digital systems (DS) to be developed, to the physical systems (PS) – i.e., diagrams and formal models – that are used to alleviate the development of the digital systems by focusing on specific aspects of the problem. Please note that the *RepresentationOf* (μ) relation is not necessarily transitive [Fav04a], and that automated transformations between systems are not in the focus. Also, the roles of the respective systems are only depicted exemplarily (blue). For more information on this kind of modeling-in-the-large, see, for example, [Fav04a, Fav05, Fav04b].

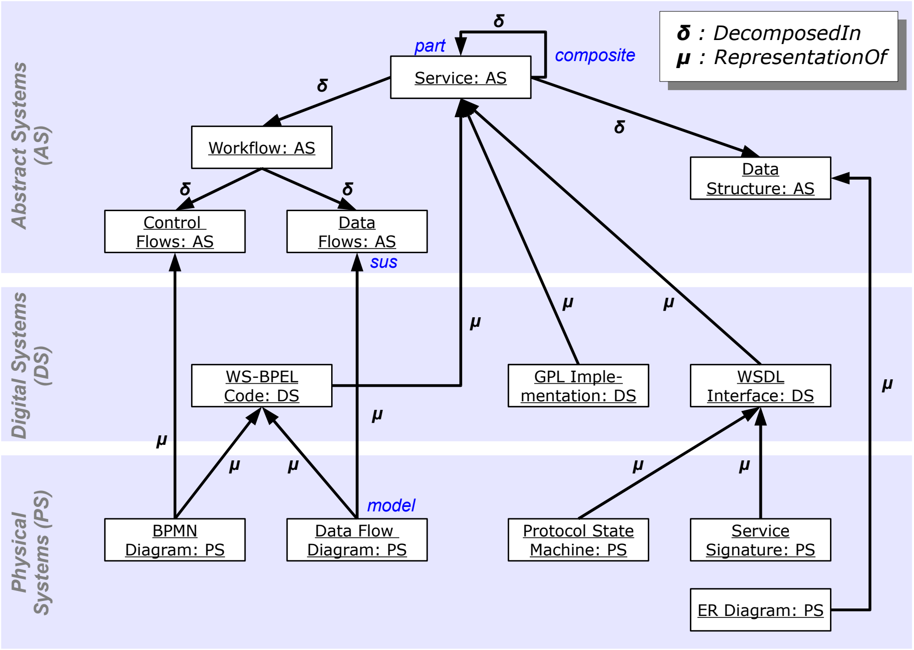


Figure 3: Model of the workflow modeling method.

The fundamental distinction is that a computer program consists of computation that has to be coordinated (cf. [GC92]). Thereby, the fundamental representation of a computer program is a service, which itself can be decomposed in services. We consider a workflow as the concrete embodiment of the coordination structure of the program, and deployed services as the concrete embodiment of computation, since they are then regarded as black boxes. An executable service can consist of, for example, the WS-BPEL process description, primitive service implementations given in general programming language (GPL) code, and the WSDL interface descriptions. In order to alleviate the development of an executable service, various diagrams and models are employed that represent parts of a service while focusing on specific aspects of the service: (1) BPMN diagrams focus on the control flows of the workflow. (2) Data flow diagrams (DFD) focus on the data flows of the workflow. (3) Protocol state machines describe the utilization of services to be orchestrated. (4) Signature descriptions describe the interfaces of services with multiple in and out-parameters. (5) Entity relationship diagrams describe the data structures to be used within the services. Based on this model, the individual process steps to identify, model, and deploy business workflows are as follows (see Fig 2).

(1) *Domain analysis*. The respective business domains have to be analyzed to gain a thorough domain understanding. This includes, for example, the analysis of the

current enterprise architecture, expert interviews, on-site investigations, and requirements analysis.

(2) *Control-flow modeling*. This activity is composed of the following sub-activities:

   (a) The current business processes (as-is state) is described using the Business Process Modeling Notation (BPMN).

   (b) From the as-is state a first version of the to-be processes are developed and described using the BPMN, too.

   (c) Data sources and simple data-flows were annotated in the to-be BPMN diagrams as far as possible using the BPMN (see the fourth lane from the top in Fig. 4).

   (d) The to-be BPMN diagrams are iteratively expanded to regard different layers of abstraction. Thereby, we consider the operational layer, the services layer, the business process layer, and the consumer layer. We especially find this activity to be very helpful in order to separate concerns at an early stage of development (cp. SOA reference architecture in [BLJ$^+$08]). Figure 4 illustrates a such-layered call center process from the first application scenario for read-only data retrieval (layers are ordered from the bottom to the top).

(3) *Data structure modeling*. Upon the relevant information systems and databases, the logical structure of the required information is modeled. To do so, we use entity-relationship (ER) diagrams that represent the relevant data structures whereas the information system/database origin of the structures is annotated.

(4) *Data-flow modeling*. In addition to control-flow modeling, we model the data-flows of business processes using data flow diagrams (DFDs).

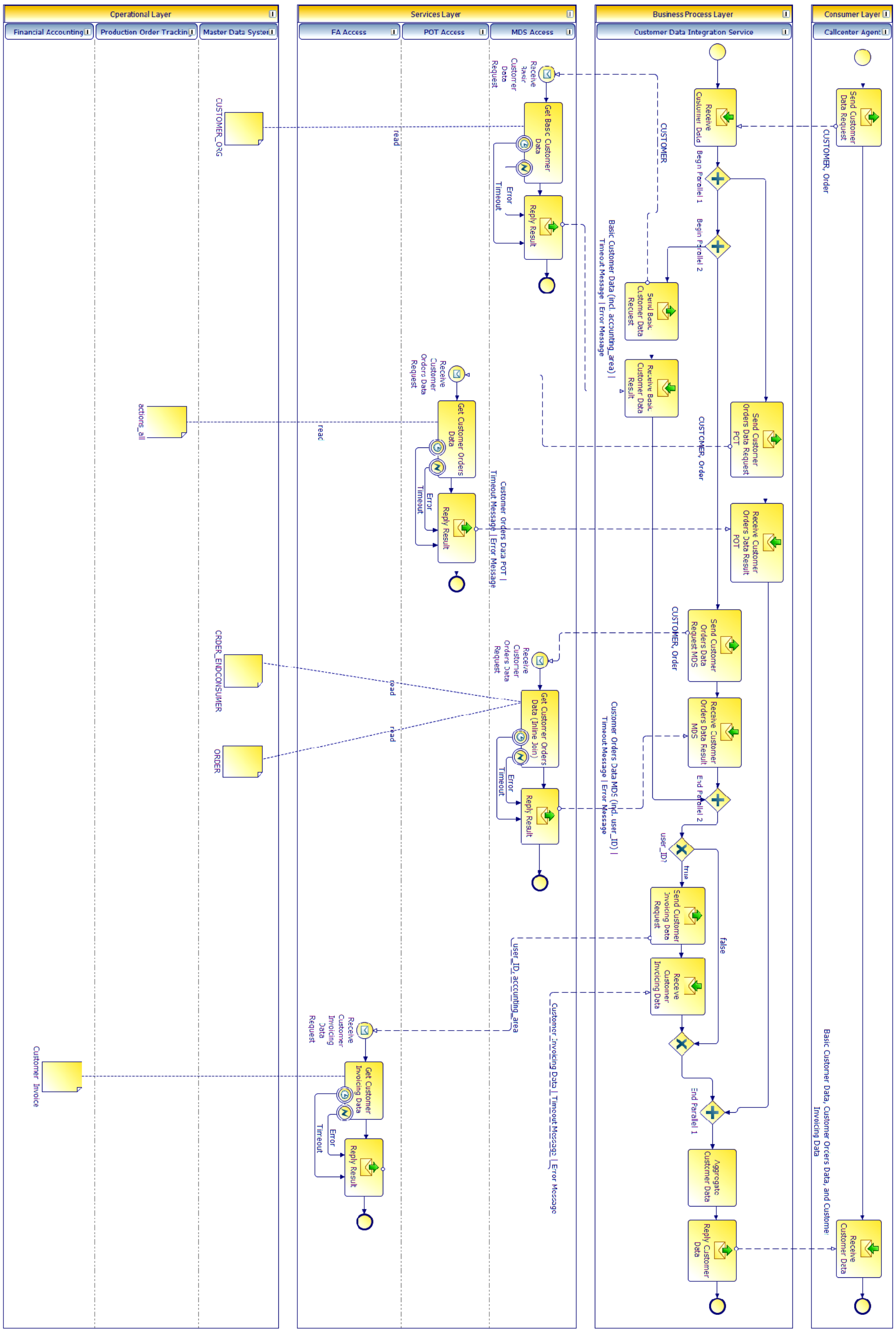
(5) *Service signature description*. Based upon the results of the previous activities, we textually describe the signatures of the services of the respective business processes as a basis for service interface definition.

(6) *Service utilization description*. In addition to signature description, we describe the usage protocol of the services regarded as black-boxes using protocol state machines. Although this activity represents an overhead for services with small signatures, we think that this activity is of great value for services that provide several operations and where the operations have strong service lifecycle dependencies.

(7) *Service implementation*. Starting with WSDL interface design, the services are implemented.

(8) *Service deployment*. The services are deployed under consideration of the enterprise architecture and the scenario requirements, for example, security requirements.

(9) *Workflow design*. Finally, we implement WS-BPEL workflows for the modeled business processes.



Operational Layer
Financial Accounting
Production Order Tracking
Master Data System
Services Layer
FA Access
POT Access
MDS Access
Business Process Layer
Customer Data Integration Service
Consumer Layer
Callcenter Agent
Send Customer Data Request
CUSTOMER, Order
Receive Customer Data
Begin Parallel 1
Begin Parallel 2
CUSTOMER
Send Basic Customer Data Request
Receive Basic Customer Data Result
Basic Customer Data (incl. accounting_area) | Timeout Message | Error Message
Receive Basic Customer Data Request
Get Basic Customer Data
Error
Timeout
Reply Result
CUSTOMER_ORG
read
Send Customer Orders Data Request POT
CUSTOMER, Order
Receive Customer Orders Data Result POT
Receive Customer Orders Data Request
Get Customer Orders Data
Customer Orders Data POT | Timeout Message | Error Message
actions_all
read
Send Customer Orders Data Request MDS
Receive Customer Orders Data Result MDS
Customer Orders Data MDS (incl. user_ID) | Timeout Message | Error Message
Receive Customer Orders Data Request
Get Customer Orders Data (Inline Join)
ORDER_ENDCONSUMER
ORDER
read
read
End Parallel 2
user_ID
true
false
Send Customer Invoicing Data Request
Receive Customer Invoicing Data
user_ID, accounting_area
Customer_Invoicing Data | Timeout Message | Error Message
Receive Customer Invoicing Data Request
Get Customer Invoicing Data
Customer_Invoice
End Parallel 1
Aggregate Customer Data
Reply Customer Data
Basic Customer Data, Customer Orders Data, and Customer Invoicing Data
Receive Customer Data

Figure 4: Call center process for read-only data retrieval as a basis for customer orders feedback.

3 Related Work

The Quasar Enterprise (QE) Method is a coherent method for realizing service-oriented architectures [EHH+08]. It provides detailed guidelines for designing and providing services which is driven by the identification of business goals and requirements. There are also many other approaches for SOA, for example, [Mel08, KBS04, AGA+08]. We regard them to be targeted at a much broader SOA realization that our approach, but possibly not as broad and comprehensive than QE.

Currently, each step in our method relies on manual execution. Particularly with regard to the concrete service development steps (e. g. WS-BPEL process and Web service implementation), there already exist approaches for (semi-)automatical transformation. In [OvdAD06], for example, a method is described to translate BPMN diagrams to WS-BPEL processes. On the technical side, model driven technologies [RH09] can be used to perform such transformation steps [YZZ+07].

In contrast to full-fledged SOA methods, the method presented in this paper is much more narrow, and specifically focuses on soft-wired service orchestration in SMEs, where the targeted SMEs are supposed to have limited resources for and little experience with service orchestration in order to undertake a complete SOA introduction in one single commitment. From this point of view, the presented method can be seen as a simple method for a first step in SOA introduction to be applied in a prototype service orchestration scenario.

4 Conclusion

In this paper, we presented the workflow modeling method that we developed within the BIS-Grid project. Although the project is aimed specifically at Grid and Cloud service orchestration that can cross enterprise boundaries, the method targets WS-BPEL service orchestration in general.

References

[AGA+08] A. Arsanjani, S. Ghosh, A. Allam, T. Abdollah, S. Gariapathy, and K. Holley. SOMA: A method for developing service-oriented solutions. *IBM Syst. J.*, 47(3):377–396, 2008.

[BGJ+09] Melanie Baier, Gernot Gräfe, Melanie Jekal, Florian Röhr, and Tanja Vörckel. Marktanalyse für kommerzielles Grid-Providing . Technical report, Universität Paderborn and Siemens AG - Siemens IT Solutions and Services, July 2009.

[BLJ+08] Norbert Bieberstein, Robert G. Laird, Keith Jones, Tilak Mitra, and Jason Weisser. *Executing SOA: A Practical Guide for the Service-Oriented Architect*, chapter A Methodology for Service Modeling and Design, pages 57–81. developerWorks Series. IBM Press, May 2008. 2008 Dimensions 7x9-1/4 240 Edition: 1st. 0-13-235374-1 ISBN-13: 978-0-13-235374-8.

[EHH+08] Gregor Engels, Andreas Hess, Bernhard Humm, Oliver Juwig, Marc Lohmann, Jan-Peter Richter, Markus Voß, and Johannes Willkomm. *Quasar Enterprise: Anwendungslandschaften serviceorientiert gestalten.* dpunkt, Heidelberg, 2008.

[Fav04a] Jean-Marie Favre. Foundations of Model (Driven) (Reverse) Engineering : Models - Episode I: Stories of The Fidus Papyrus and of The Solarus. In *Language Engineering for Model-Driven Software Development*, 2004.

[Fav04b] Jean-Marie Favre. Towards a Basic Theory to Model Model Driven Engineering. In *In Workshop on Software Model Engineering, WISME 2004, joint event with UML2004*, 2004.

[Fav05] Jean-Marie Favre. Foundations of Meta-Pyramids: Languages vs. Metamodels – Episode II: Story of Thotus the Baboon. In Jean Bezivin and Reiko Heckel, editors, *Language Engineering for Model-Driven Software Development*, number 04101 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany, 2005. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany.

[GC92] David Gelernter and Nicholas Carriero. Coordination Languages and their Significance. *Communications of the ACM*, 35:97–107, 1992.

[GHH+09] Stefan Gudenkauf, Wilhelm Hasselbring, André Höing, Guido Scherp, and Odej Kao. Workflow Service Extensions for UNICORE 6 - Utilising a Standard WS-BPEL Engine for Grid Service Orchestration. *Lecture Notes in Computer Science: Euro-Par 2008 Workshops - Parallel Processing*, 5415:103–112, 2009.

[HSG+09] André Höing, Guido Scherp, Stefan Gudenkauf, Dirk Meister, and André Brinkmann. An Orchestration as a Service Infrastructure Using Grid Technologies and WS-BPEL. In *ICSOC/ServiceWave*, pages 301–315, 2009.

[KBS04] Dirk Krafzig, Karl Banke, and Dirk Slama. *Enterprise SOA: Service-Oriented Architecture Best Practices (The Coad Series).* Prentice Hall PTR, Upper Saddle River, NJ, USA, 2004.

[Mel08] Ingo Melzer, editor. *Service-orientierte Architekturen mit Web Services: Konzepte – Standards – Praxis.* Spektrum, Heidelberg, 3. edition, 2008.

[OvdAD06] C. Ouyang, WMP. van der Aalst, and AHM. ter Hofstede M. Dumas. Translating BPMN to BPEL. Technical report, 2006.

[RH09] Ralf Reussner and Wilhelm Hasselbring, editors. *Handbuch der Software-Architektur.* dpunkt, Heidelberg, 2. edition, 2009.

[YZZ+07] Xiaofeng Yu, Yan Zhang, Tian Zhang, Linzhang Wang, Jianhua Zhao, Guoliang Zheng, and Xuandong Li. Towards a Model Driven Approach to Automatic BPEL Generation. In *ECMDA-FA*, pages 204–218, 2007.