An Orchestration as a Service Infrastructure using Grid Technologies and WS-BPEL

A. Höing[1], G. Scherp[2], S. Gudenkauf[2], D. Meister[3], A. Brinkmann[3]

[1] Technische Universität Berlin, Complex and Distributed IT Systems, Einsteinufer 17, 10587 Berlin, Germany
email: andre.hoeing@tu-berlin.de
[2] OFFIS Institute for Information Technology, Technology Cluster EAI, Escherweg 2, 26121 Oldenburg, Germany
email: [stefan.gudenkauf, guido.scherp]@offis.de
[3] University of Paderborn, Paderborn Center for Parallel Computing, Fürstenallee 11, 33100 Paderborn,Germany
email: [dmeister, brinkmann]@uni-paderborn.de

Abstract. The BIS-Grid project, as part of the German D-Grid initiative, investigates service orchestration using Grid service technologies to show how such technologies can be employed for information systems integration, especially when crossing enterprise boundaries. Small and medium enterprises will be enabled to integrate heterogeneous business information systems and to use external resources and services with affordable effort.
In this paper, we discuss our Orchestration as a Service (OaaS) paradigm and present the BIS-Grid OaaS infrastructure. This infrastructure is based upon service extensions to the Grid middleware UNICORE 6 to use an arbitrary WS-BPEL workflow engine and standard WS-BPEL to orchestrate both plain Web services and stateful, WSRF-based Grid services. We report on the evaluation scenarios at our industrial application partners and on the applied service modeling methodology.

1 Introduction

The integration of heterogeneous information systems, referred to as Enterprise Application Integration (EAI), 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. This characteristics of cloud computing are also referred to as *Infrastructure as a Service* (IaaS), *Platform as a Service* (PaaS), and *Software as a Service* (SaaS). As example, Amazon offers its IaaS product Elastic

Original Article published at Springer, see
http://www.springerlink.com/content/jtk1172u34p26t43/?p=437aaa79a7bd4e1ba4c15788de79139a&pi=1

Compute Cloud (EC2)[4], but also others in the context of its Amazon Web Services platform. An IaaS open source implementation using the same Web services interfaces as EC2 is available from the EUCALYPTUS project [9].

Cloud services are designed as on-demand services as they only charge what users actually consume. Therefore, they are especially interesting for small, medium, and start-up enterprises that need highly scalable IT infrastructures and/or do not want to run the respective IT infrastructures on their own. However, such companies also have the need to map their business processes to the technical system level, integrating outsourced cloud services as well as in-house-hosted services. Nowadays, many companies offer consultant services supporting small enterprises to identify their key business processes and to create integrated IT environments by the introduction of in-house orchestration engines. This brings up the idea of *Orchestration as a Service* (OaaS), meaning that the orchestration engine is hosted in a cloud environment, directly to be maintained by the OaaS provider. Considering the security and privacy of the deployed workflows and their data, such an orchestration engine should be designed as a multi-tenant service, decreasing costs since hardware can be shared over several customers.

In the BIS-Grid project, we focus on realizing EAI using Grid service technologies. Our major objective is to proof that Grid technologies are feasible for information systems integration, especially when traversing 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, even across company boundaries. To do so, we propose and regard *Orchestration as a Service* (OaaS) as the primary infrastructure paradigm. This paper is organized as follows. OaaS is discussed in Sec. 2, and the BIS-Grid OaaS infrastructure is presented in Sec. 3, including our OaaS-capable workflow engine and the general security infrastructure. Section 4 presents our application scenarios and describes the applied service modeling methodology. After the presentation of related work in Sec. 5 a conclusion is given in Sec. 6.

2 Orchestration as a Service

Integration is a topic for both industry and research for many years in order to enable the seamless interaction of (heterogeneous) applications. Modern integration solutions adopted the service-oriented architecture (SOA) design paradigm which is tightly coupled with the representation of business logic. This means that all applications are encapsulated by enclosed, loosely coupled, often low-level services which are composed to business processes. Such a business process, often referred to as service orchestration, is often modeled graphically in order to develop executable workflow representations. The Web Services Business Process Execution Language (WS-BPEL) [10], an OASIS standard to orchestrate Web services, is an example of such a representation. Commonly, it is regarded as a

[4] http://aws.amazon.com/ec2.

key technology to build SOA, and to offer service orchestrations itself as Web services. This enables the use of low-level processes to build complex services on a higher level.

The use of loosely coupled services in SOA allows to dynamically switch the location of invoked services in order to utilize services offered by an external provider instead of local services. Thus, SOA can be considered as an enabling technology to outsource IT infrastructure and corresponding services to reduce IT costs. Grid and the upcoming Cloud technologies are examples for realizing such outsourcing scenarios. Based on SOA and the rapid development of Internet technologies several service providers emerged that offer services known as *Cloud*. The general idea behind Cloud is that services can be accessed on demand after a short setup time based on a pay-per-use utility model. As the range of Cloud services is highly diverse a classification in the manner of "* as a Service" is widely adopted at present. Such services can be divided into three categories: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS) [12]. In brief, IaaS represents the service-based access to (virtualized) computing resources as storage and processing power, PaaS represents the service-based access to a software platform that enables the custom development of (scalable) applications that are normally executed on a virtualized infrastructure. SaaS represents the service-based access to a specific software product (e. g., ERP software) or a certain functionality (e. g., creditworthiness check). Furthermore, in the following order, IaaS, PaaS, and SaaS are considered as subsequent abstraction layers to the executing infrastructure. In our opinion such services can also be provided by utilizing Grid technologies which are in general highly related to Cloud technologies [12].

Microsoft BizTalk Server[5] or SAP XI[6] are actual commercial integration platforms that can execute business processes and suitable for SOA integration. In order to run such a platform costs such as licenses, hardware and system administrators have to be considered. Many companies such as small and medium enterprises (SMEs) are not able to finance or operate such an infrastructure although they certainly have needs for integration. Even freely available products like Sun's OpenESB[7] are hard to set-up and maintain in a productive manner. Thus, our approach is to offer such an integration platform as an external service which we call *Orchestration as a Service*(OaaS). Thereby, we regard OaaS as a specialization of PaaS, as process developers are able to develop, deploy and manage custom business processes, and SaaS, as end users can use the functionality of deployed business processes as services. Our OaaS solution is build upon the WS-BPEL-based BIS-Grid engine which is developed in the BIS-Grid project and described in Sec. 3 in detail.

In summary, the described OaaS scenario must meet the requirements *Grid compatibility*, *SOA compatibility* and *WS-BPEL compatibility*. Beside OaaS we considered further deployment scenarios depending on the degree of Grid utiliza-

[5] http://www.microsoft.com/biztalk/en/us/default.aspx
[6] http://www.sap.com/platform/netweaver/pdf/BWP_SB_ExchangeInfrastructure.pdf
[7] https://open-esb.dev.java.net

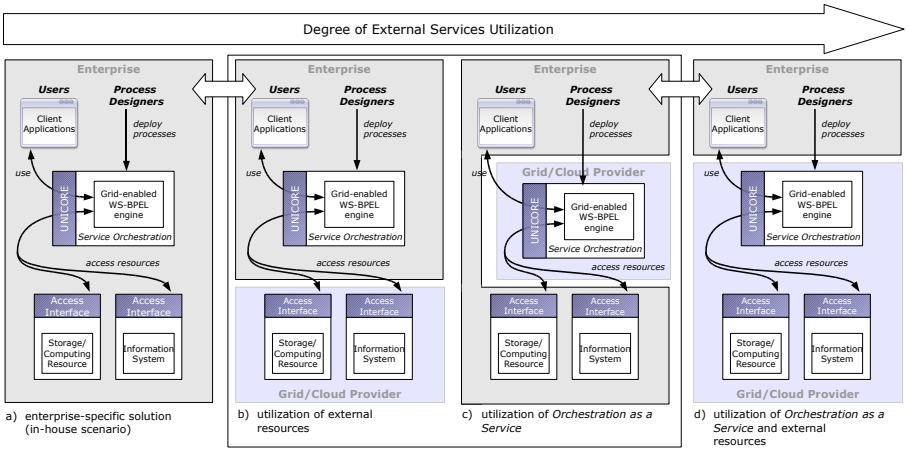

Fig. 1. Degree of Grid utilization

tion of the involved components, namely the BIS-Grid engine and orchestrated services as information systems. This ranges from (a) a pure in-house scenario, in which all components are deployed within a company, up to (d) a complete Grid scenario, in which each component is located in a Grid or Cloud infrastructure, see Fig. 1. The scenarios (b) and (c) are alternatives in which involved components are partially outsourced. In our case the in-house scenario (a) and the OaaS scenario (c) are currently regarded as most realistic since outsourcing information systems to Grid or Cloud providers often raises serious security concerns. So the philosophy is to keep all data services in-house and relevant data for orchestration and integration may leave the company on demand under ensuring certain security standards[8], see Sec. 3.2.

3 BIS-Grid OaaS Infrastructure

This section describes the BIS-Grid engine that is used to realize the technical side of the OaaS infrastructure. First, in Sec. 3.1 we describe the architecture of the orchestration engine, including the components of the engine and how they interact with each other. Second, we describe the security infrastructure for authentication and authorization in Sec. 3.2, also addressing privacy issues for realizing a multi-tenant environment.

3.1 BIS-Grid Engine Architecture

The BIS-Grid engine was designed with regard to its applicability in our OaaS scenario, see Sec. 2. This means the engine meets the following requirements:

[8] Beside the technical and organizational issues discussed in this paper, this also involves legal issues to be covered which are not part of the BIS-Grid project

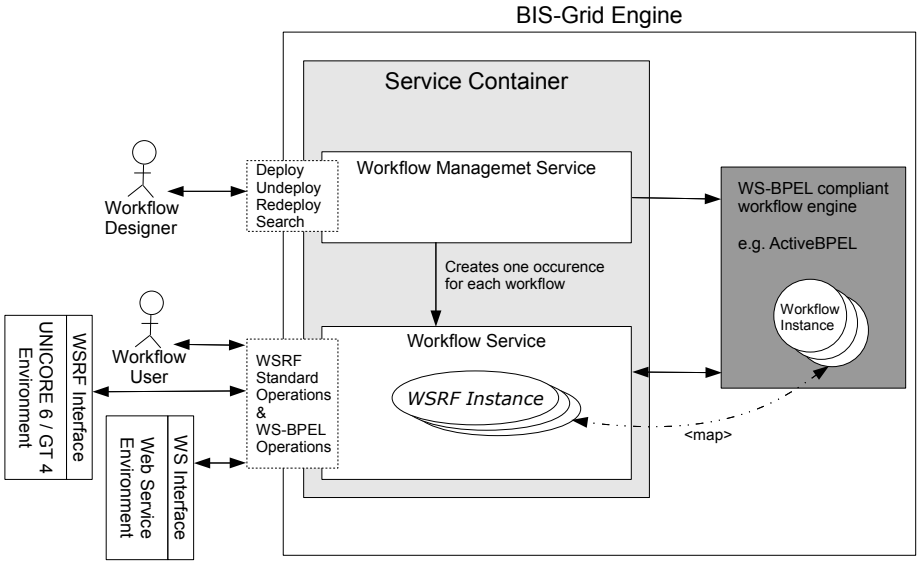

Fig. 2. The BIS-Grid engine architecture

- *Grid compatibility*: The BIS-Grid engine is based on the Grid middleware UNICORE 6.
- *SOA compatibility*: As UNICORE 6 is based on Web services which is a key technology to build SOA.
- *WS-BPEL compatibility*: The BIS-Grid engine utilizes an arbitrary standard WS-BPEL engine to execute workflows.
- *Security*: The BIS-Grid engine supports authorization and authentication supporting a fine-grained role-based access control.

A more detailed view on the BIS-Grid engine's architecture is depicted in Fig. 2. The key concept is to use an arbitrary WS-BPEL engine, in our case ActiveBPEL, for workflow execution that is encapsulated by a *Grid proxy* based on service extensions to UNICORE 6. These service extensions enable interoperability between WS-BPEL and Grid environments by the support of the Web Service Resource Framework (WSRF) and Grid security. WSRF is an OASIS standard developed by the Grid community in order to enable stateless Web services to become stateful. This is essential, for instance, for job submission and data transfer services since they have a state by nature. WSRF-based Web services are also called Grid services, their states are represented by service instances and are stored in so called *resource properties*. Such properties can be accessed or manipulated by corresponding WSRF Web service methods. We discuss Grid security separately in Sec. 3.2.

The BIS-Grid service extensions mentioned above are the *Workflow Management Service* and the *Workflow Service* which are both realized as Grid services.

The Workflow Management Service is initially deployed in the UNICORE 6 service container and provides methods for workflow management (i. e., deployment, redeployment, and undeployment of workflows). When a WS-BPEL workflow is deployed, a specialized version of the generic Workflow Service is created on-demand and hot-deployed to the UNICORE 6 service container. Thus, each deployed workflow in the WS-BPEL engine has a corresponding Workflow Service in UNICORE 6. A Workflow Service provides the same interface as its WS-BPEL workflow counterpart. The state of a workflow execution and additional configurations such as security credentials (see Sec. 3.2) are exposed by the corresponding Grid service instance as resource properties according to the WSRF standard. Since a workflow execution in the WS-BPEL engine is also regarded as an instance, each workflow execution is represented by two instances, one Grid service instance in the UNICORE 6 service container of the BIS-Grid engine, and one workflow instance in the WS-BPEL engine.

The communication between the BIS-Grid service extensions and the WS-BPEL engine depends on the used functionality. For management functions like workflow deployment and undeployment or workflow monitoring an engine-specific adapter is used. This adapter is pluggable and can be exchanged to support other WS-BPEL engines. The communication during workflow execution is based on standard Web service calls (SOAP) whereas outgoing messages (i. e., the invocation of external services within the workflow) must be sent to a HTTP proxy running in the UNICORE 6 container. Most Web service containers should provide a HTTP proxy configuration. If supported, the secure HTTPS protocol can be used, too. As each workflow execution has two instances, it is ensured that outgoing messages that originate from a workflow instance and that are sent to the HTTP(S) proxy, are forwarded to the correct Grid service instance. Then, the Grid service instance performs the external service call. This is especially important due to security concerns as certain credentials may be used for a specific service invocation. Thus, these credentials can be configured both globally at design time for a workflow and locally at runtime for a workflow's Grid service instance. It is in the responsibility of the workflow designer to ensure that ingoing messages (i. e., invocations of a workflow's Web service method), that primarily are send to a corresponding Workflow Service instance, are forwarded to the correct workflow instance. This is based on the WS-BPEL correlation concept. For more information about the described instance mapping please refer to [5, 6].

3.2 Security Infrastructure

Security is one of the most important issues when setting up a Cloud computing environment like our OaaS infrastructure. Security can be achieved on different ways. On the one hand, we could install one BIS-Grid engine for each customer and use IP-based authentication. On the other hand, such a solution would be very expensive. In this section, we propose a security infrastructure that guarantees authentication, role based access control, and information privacy. To lower start-up and maintenance costs, we require a solution with minimal costs

and maintenance overhead. Furthermore, credential delegation is an urgently needed feature. This enables the user to delegate rights to the OaaS environment to invoke external services in his name meanwhile the OaaS provider himself has naturally no access to that service.

Nowadays, Grid security is based on personalized X.509 certificates issued by a Certificate Authority (CA). Everyone participating in the Grid must trust this CA and authenticate himself with this CA or an associated Registration Authority (RA). Rights are not granted on business roles but on membership to a virtual organization. This scenario is not applicable for the business domain. A company, possibly having several hundred employees and high employee fluctuation can not send each new employee to a RA. To reduce maintenance overhead, rights must be bound to business roles and not to organizational membership. Hence, a distributed identity management system with the possibility to grant or revoke role-based permissions in a short time is necessary for OaaS.

We decided to build the security infrastructure up on well-known standards. SAML Assertions [11] are capable to fulfill most of our requirements. Arbitrary attributes, as roles and affiliation, can be included into an assertion. Such assertions are issued by an identity management system. SAML also provides the capability to express fine-grained credential delegation rights, expressing what entity is allowed to process what activity until what timestamp. For describing rights, we use XACML Policies [8] that are also well-known and very flexible and fine-grained if required. Access decisions can be described as rules, that define the applicability of a rule by means of the user, targeted resource, and the desired action. Furthermore, conditions that express dependencies between these can be formulated. Users are identified by the sum of all attributes included in the SAML assertion or/an via additional attributes requested during authorization process. Therefore, the policy designer must be aware of the organizational structures of the enterprises to describe the access rights correctly.

As technical infrastructure for an appropriate distributed identity management system with low maintenance costs and the capability to integrate different identity management systems we suggest the Shibboleth-based system[9]. This solution prevents the users from all complex security configurations. In combination with Grid-Shib[10], the system is able to automatically issue short-lived certificates (SLC) together with a SAML assertion including the user's business roles. Welch et al. described such an architecture in more detail [13]. The SLC is used to establish a SSL connection between the user and the BIS-Grid engine so that transport layer security is guaranteed. The major advantage is the seamless integration of existing identity management systems such as Active Directory or OpenLDAP with the Shibboleth system. These internally hosted and maintained systems can obtain and check credentials as well as supply attributes, so-called campus attributes. SLCs only have short lifetimes, usually one million seconds (circa 11 days), so that all roles and the connected rights are invalid after this period.

[9] http://shibboleth.internet2.edu/
[10] http://gridshib.globus.org/

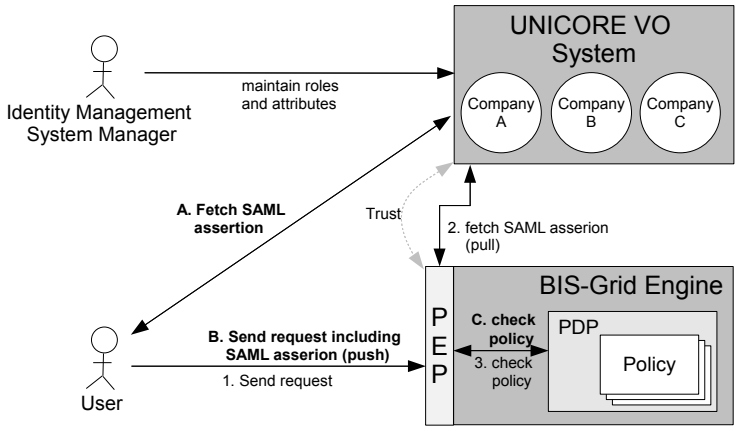

Fig. 3. Interaction between the BIS-Grid engine and UVOS

Because of simplicity of the exemplary evaluation, we decided not to set up a complete Shibboleth environment for our OaaS prototype but we use a similar solution that also provides the urgently needed SAML Attribute Assertions. The UNICORE Virtual Organisations System (UVOS)[11] allows the administration of user identities combined with arbitrary attributes for each identity. Additionally, hierarchical organizations can be mapped to hierarchical organized groups as well as attaching attributes to all members of a group. Groups or sub-groups members and attributes can be managed by different administrators. All UVOS-managed information can be queried by SAML2-compatible applications [3] and UVOS answers with a signed SAML assertion including all attributes (groups affiliations, group attributes, and global attributes). The combination of UVOS and UNICORE 6 are fully integrated and well-tested in the Chemomentum project, wherein UVOS was developed[12].

UNICORE 6 (and for this reason the BIS-Grid engine, too) supports two mechanism for assertion retrieval: push and pull. Push (cp. Fig. 3 A-C) means that the user authenticates himself at the UVOS server and retrieves his signed assertion that he attaches to the request to the BIS-Grid engine. Pull (cp. Fig. 3 1-3) uses the distinguished name of the user's certificate to fetch the assertion itself from the UVOS server. Nevertheless, there are two major disadvantages of UVOS compared to the Shibboleth solution. First, the user does not obtain a SLC for establishing TLS or for signing credential delegation assertions to delegate trust to the BIS-Grid workflow engine. Hence, each user must still own a standard X.509 certificate. Second, there is no integration of local identity management systems. This means that all identity information must be maintained a second time, beside the already existing company-local identity management system, either by the OaaS provider or by the companies themselves.

[11] http://uvos.chemomentum.org/
[12] http://chemomentum.org/

Hot deployment of Workflow Service also demands the hot deployment of security policies. We established such a hot deployment of XACML policies by adding new rules to the Policy Decision Point (PDP) that is part of the BIS-Grid deployment package. However, the possibility to add new policies also brings up the danger of misuse, for example, adding policies that affects other services. We limit the degree of freedom of newly inserted XACML rules in such a way that the rules must be limited to the newly created Workflow Service. Otherwise the deployment of the policy will fail.

After discussing authentication and authorization, we have to consider privacy. If several companies work with the same OaaS infrastructure, it must be ensured that only authorized users can see *what workflows are deployed* or *what workflows are currently running*. Authorized, in this case, does not only mean that the user must have the right to deploy a workflow but the system must also distinguish between affiliations or even departments during information retrieval operations. The same applies to running workflow instances.

We established means to filter information when discovery operations are used that must be accessible from different companies due to architectural issues, for example, creating new Workflow Management Service instances or searching for already created instances. Therefore, we store enriched information about the creator of an instance in the instances itself. As an example, instances of the Workflow Management Service store the creator's distinguished name, his affiliation, and his business role. When someone else searches for all deployed workflows, the BIS-Grid engine will only show deployed workflows matching the same affiliation and business role. Both must be included in the signed SAML assertion the requester presents. A similar filter also guarantees privacy for searching for instances of a Workflow Service. All other information depends on WSRF instances and are protected via the XACML policies.

4 Application Scenarios

We evaluate our OaaS approach in two business scenarios motivated by our industrial project partners, CeWe Color[13] and KIESELSTEIN Group[14]. CeWe Color is the number one services partner for first-class trade brands on the European photographic market supplying stores and internet retailers with photographic products, and KIESELSTEIN Group is one of the global market leaders in the field of wire drawing and draw-peeling for the automotive industry. Both partners have strong needs for enterprise application integration: CeWe Color to integrate enterprise data for unified access for call center agents, and KIESELSTEIN Group to improve access to, and retrieval and maintenance of product and project data. The overall goal of these application scenarios is to investigate the feasibility of EAI based on Grid technologies and the OaaS paradigm by prototypical realization.

[13] http://www.cewecolor.de
[14] http://www.kieselstein-group.com

For CeWe Color, the impact of digital photography affected requirements to business information systems (BIS) and processes, opened up new distribution channels, and facilitated new product lines. Product mass customization and the need to flexibly respond to market development demand BIS that can adapt dynamically. Regarding the call center scenario, information of different sources must be accessed by call center agents to provide feedback to customers, for example about order status, production failures, or accounting data. This access has to be provided unified and with hard constraints to the quality of the demanded services. For KIESELSTEIN Group, the main challenge is to integrate enterprise resource planning (ERP) data and product (CAD/PDM) data that are distributed across different sites. At these sites, BIS store information redundantly, since KIESELSTEIN Group grew together from three different producing factories, each providing their own information systems.

4.1 Workflow Modeling Methodology

Within the business application scenarios, we employed a top-down workflow modeling methodology. Figure 4 presents an overview of this methodology. Within a concrete workflow development process, the individual modeling activities may be applied in different orders. The upper half of the picture shows the creative part of the modeling methodology, and the lower half shows the components during the operational service. Additionally, the business roles are annotated to each component. The arrows depict main dependencies between the components.

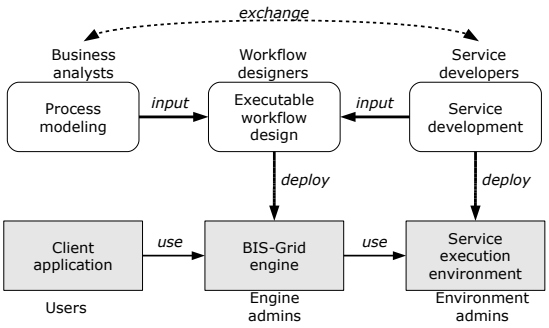

Fig. 4. Overview of the workflow modeling methodology.

The design of a workflow mainly depends on the process model of a business analyst that uses BPMN as an high-level abstraction model to describe the desired business process, and on a service developer to identify existing services that can be reused or to design new services that have to be implemented and deployed. Furthermore, there is also an information exchange between a business analyst and a service developer, for example, to gain a common view of a global data model. To provide executable workflows, the workflow must be deployed on

the BIS-Grid engine and, of course, all used services must be available on their service execution environments (cp. use-dependency in Fig. 4). Finally, users can initiate workflows using an appropriate client application.

For the evaluation of our application scenarios, we used a top-down development approach to identify, model, and deploy business workflows. The individual process steps were as follows.

(1) *Domain analysis.* The respective business domain(s) had to be analyzed to gain a thorough domain understanding. This included, for example, the analysis of the current enterprise architecture, expert interviews, on-site investigations, and requirements analysis.
(2) *Control-flow modeling.* This activity included the following sub-activities:
   (a) The current business processes (as-is state) were described as diagrams using the Business Process Modeling Notation (BPMN).
   (b) From the as-is state a first version of the to-be processes were developed and described using the BPMN, too, representing the basis for the realization of the prototype scenarios.
   (c) Data sources and simple data-flows were annotated in the to-be BPMN diagrams as far as possible using the BPMN (cp. Figure 5 lane 4).
   (d) The to-be BPMN diagrams were iteratively expanded to regard different layers of abstraction directly within the diagrams. Thereby, we focused on the operational layer, the services layer, the business process layer, and the consumer layer. We especially found this activity to be very helpful in order to separate concerns at an early stage of development (cp. SOA reference architecture in [4]). Figure 5 illustrates a such-layered call center process for read-only data retrieval (layers are ordered from the bottom to the top).
(3) *Data structure modeling.* Upon the relevant BIS and databases, the logical structure of the required information was modeled. To do so, we used entity-relationship (ER) diagrams that represent the relevant data structures whereas the BIS/database origin of the structures was annotated.
(4) *Data-flow modeling.* In addition to control-flow modeling, we modeled 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 described 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 described the usage protocol of the services regarded as black-boxes using protocol state machines. Although representing an overhead for services with small signatures, we think that this activity is of great importance 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 were implemented by our partners.
(8) *Service deployment.* The services were deployed by our partners under consideration of the enterprise architecture and the scenario requirements, for example, security requirements.

(9) *Workflow design.* Finally, we implemented WS-BPEL workflows for the modeled business processes.
(10) *Workflow use.* Users can now execute workflows via appropriate client applications. Within the application scenarios, we developed a prototypical user client on basis of the GridSphere Portal Framework[15], and use NetBeans IDE[16] as a workflow modeling tool.

5 Related Work

There are several upcoming and new projects regarding service orchestration in cloud environments, proving the relevance of the Orchestration as a Service paradigm. Unfortunately, these are all new projects and aim at commercial issues and hence the infrastructures are not described neither scientifically nor in detail. Here, we present some of these projects but also regard related work concerning service orchestration in Grid environments.

For example, Microsoft is recently providing the .NET Workflow Service as part of the .Net Services of the Azure Services Platform[17] in order to execute user-defined declarative workflows as lightweight service orchestrations. These services facilitate the idea of an *Internet Service Bus* that addresses the need for cross-enterprise service orchestration, supporting both the Software as a Service paradigm as well as Microsoft's Software-plus-Services strategy.

CSC[18], as another example, recently announced Cloud Orchestration Services and Trusted Cloud Services promising various features such as service level management, remote monitoring, reporting, data transparency, and security while ensuring industry-specific compliance and auditing services. Thereby, Business Process as a Service (BPaaS) is named as one category of Trusted Cloud Services. Unfortunately, there is very little information on the concrete services, their realization, and the respective Service Level Agreements.

As a third example, Cordys also promotes cloud-based service orchestration, called Enterprise Cloud Orchestration[19]. Thereby, they emphasize the still-traditional nature of the SaaS distribution model in contrast to the Cloud idea as a federation of different Clouds, that may range from general-purpose Clouds to specialized Clouds in the future. Fundamentally this requires an orchestration layer in the Cloud to enable enterprises developing new business models and facilitate Application Service Provisioning.

The Chemomentum project already provides workflow extensions for UNICORE 6, consisting of two UNICORE 6 service containers. The first represents a workflow engine that processes workflows on a logical level, the second represents a service orchestrator that transforms so-called Work Assignments into

[15] http://www.gridsphere.org/
[16] http://www.netbeans.org/
[17] http://www.microsoft.com/azure/workflow.mspx
[18] http://www.csc.com/cloud/
[19] http://www.cordys.com/cordyscms_com/enterprise_cloud_orchestration.php

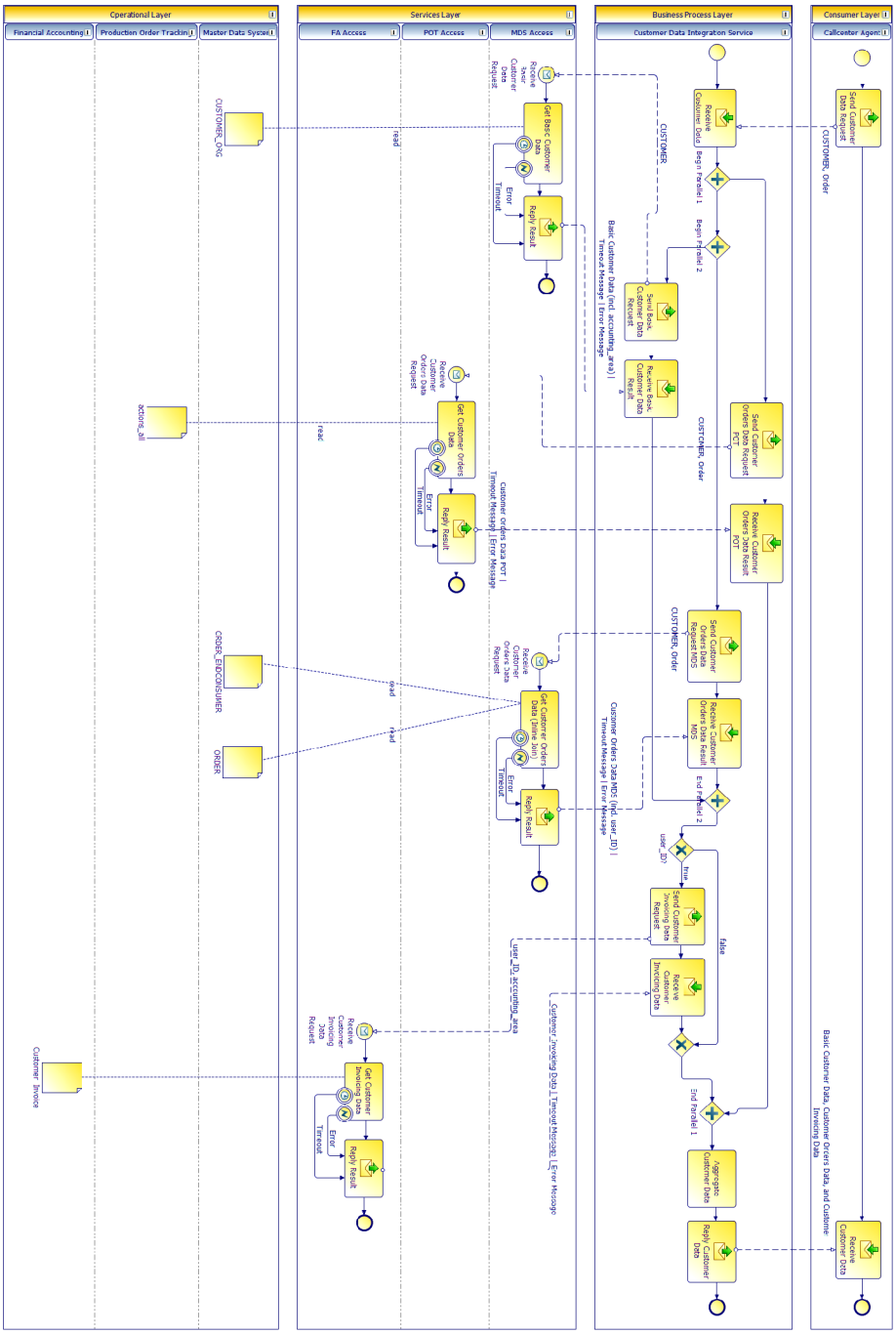

Fig. 5. A layered BPMN describing a call center process for read-only data retrieval as a basis for customer orders feedback.

jobs, given in the Job Submission Description Language (JSDL) [1]. Both, this UNICORE 6 workflow system and the BIS-Grid engine, are implemented as service extensions to the UNICORE 6 service container. However, the UNICORE 6 workflow system does not support the integration of a WS-BPEL workflow engine well-adopted in industry.

In [2], Amnuaykanjanasin and Nupairoj present a solution to orchestrate Globus Toolkit services secured with the Grid Security Infrastructure. For each Grid service, a proxy implementation is generated automatically when the user requests one. To overcome the GSI, Proxy Certificates are requested dynamically from a MyProxy implementation. This architecture aims on scientific workflows without considering role-based access control or providing the workflow itself securely.

Many other paper present the possibility to model and execute workflows in Grid environments but without using the industrial de-facto standard WS-BPEL nor addressing the new cloud computing paradigm (e. g., see [7, 14]).

6 Conclusion

In this paper, we discussed our Orchestration as a Service (OaaS) paradigm as a specialized form of PaaS. Since Oaas decreases the start-up costs for introducing EAI and SOA by outsorcing the operation and maintenance of a service orchestration infrastructure, we regard it as a viable option for small and medium enterprises. We also presented the BIS-Grid OaaS infrastructure, enabling enterprises to deploy workflows based on service-oriented architectures and opening up new possibilities for outsourcing IT to Grid or Cloud providers. Thereby, we regard different degrees of service outsourcing relying on the same technology underlying. The definition of service-level agreements (SLA) that are adequate to OaaS is future work.

Acknowledgement

The underlying work for this paper would not have been possible without the support of and close cooperation with our project partners. We especially thank our colleagues Herbert Nase, Manfred Neugebauer, and Christoph Rüger for their engagement.

References

1. Job Submission Description Language (JSDL) Specification, Version 1.0. http://www.gridforum.org/documents/GFD.56.pdf, November 2005.
2. Pichet Amnuaykanjanasin and Natawut Nupairoj. The bpel orchestrating framework for secured grid services. *Information Technology: Coding and Computing, International Conference on*, 1:348–353, 2005.

3. Krzysztof Benedyczak. UNICORE Virtual Organisations Service Overview. Technical report, Interdisciplinary Centre for Mathematical and Computational Modelling Warsaw University, Poland, 2007.
4. 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.
5. André Brinkmann, Stefan Gudenkauf, Wilhelm Hasselbring, André Höing, Odej Kao, Holger Karl, Holger Nitsche, and Guido Scherp. Employing WS-BPEL Design Patterns for Grid Service Orchestration using a Standard WS-BPEL Engine and a Grid Middleware. In Marian Bubak, Michal Turala, and Wiatr Kazimierz, editors, *CGW'08 Proceedings*, pages 103–110, Cracow, Poland, 2009. ACC CYFRONET AGH.
6. Stefan Gudenkauf, André Höing, and Guido Scherp. Catalogue of WS-BPEL Design Patterns. Technical report, 05 2008.
7. Andreas Hoheisel. User tools and languages for graph-based Grid workflows: Research Articles. *Concurr. Comput. : Pract. Exper.*, 18(10):1101–1113, 2006.
8. Tim Moses. eXtensible Access Control Markup Language (XACML) Version 2.0. `http://docs.oasis-open.org/xacml/2.0/access_control-xacml-2.0-core-spec-os.pdf`, Februrary 2005.
9. Daniel Nurmi, Rich Wolski, Chris Grzegorczyk, Graziano Obertelli, Sunil Soman, Lamia Youse, and Dmitrii Zagorodnov. The Eucalyptus Open-source Cloud-computing System. In *Proceedings of 9th IEEE International Symposium on Cluster Computing and the Grid*, 2009.
10. OASIS WSBPEL Technical Committee. Web Services Business Process Execution Language (WSBPEL) Primer. http://www.oasis-open.org/committees/download.php/23974/wsbpel-v2.0-primer.pdf, May 2007.
11. Nick Ragouzis, John Hughes, Rob Philpott, Eve Maler, Paul Madsen, and Tom Scavo. Security Assertion Markup Language (SAML) V2.0 Technical Overview. `http://www.oasis-open.org/committees/download.php/22553/sstc-saml-tech-overview-2%200-draft-13.pdf`, February 2007. Working Draft.
12. Luis M. Vaquero, Luis Rodero-Merino, Juan Caceres, and Maik Lindner. A break in the clouds: towards a cloud definition. *SIGCOMM Comput. Commun. Rev.*, 39(1):50–55, 2009.
13. V. Welch, T. Barton, K. Keahey, and F. Siebenlist. Attributes, Anonymity, and Access: Shibboleth and Globus Integration to Facilitate Grid Colloboration. In *Proceedings of the 4th Annual PKI R&D Workshop*, 2005.
14. Jia Yu and Rajkumar Buyya. A novel architecture for realizing grid workflow using tuple spaces. In *GRID '04: Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing*, pages 119–128, Washington, DC, USA, 2004. IEEE Computer Society.