computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Journal of Computing

The BIS-Grid Engine: An Orchestration as a Service Infrastructure

André Höing[1)], Guido Scherp[2)], Stefan Gudenkauf[2)]

[1)] Technische Universität Berlin, Einsteinufer 17, 10587 Berlin, andre.hoeing@tu-berlin.de, http://www.cit.tu-berlin.de
[2)] OFFIS, Escherweg 2, 26121 Oldenburg, {guido.scherp, stefan.gudenkauf}@offis.de, http://www.offis.de

Abstract: *The need for information system integration is typical for many companies including small and medium-sized enterprises (SMEs). But especially for SMEs, the costs to run a full-fledged integration platform in-house are beyond the available IT budget. This article describes the concept of Orchestration as a Service (OaaS), a specialization of the Platform as a Service (PaaS) paradigm in the Cloud (computing) world. The goal of this paradigm is to provide a workflow-based integration platform as a (Cloud) service focusing on so-called service orchestrations. We present the BIS-Grid Engine as a core middleware for an OaaS infrastructure including a discussion about how our solution addresses security requirements that are a key issue in Cloud technologies.*

Keywords: *Cloud, Grid, Orchestration, Service Oriented Architecture, Security, Orchestration as a Service, Platform as a Service*

1. INTRODUCTION

Over the last few years the Cloud computing paradigm gained more and more momentum. Its main characteristics are immediate scalability, resources on demand provisioning and usage optimization, virtualization, and a pay-per-use model. Today, there is a consensus that categorizes Cloud computing services into the layers Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS) (cp. [1, 2]). Web services turned out to be the key technology to access services at each layer.

IaaS is the basic layer to allocate and use virtualized compute and storage resources. Users are completely free in designing their own computing infrastructure. For example, providing customized operation system images on virtual resources to process data from storage services. Currently, Amazon EC2 and S3 Cloud Services [17] are the most famous examples for IaaS. PaaS, in contrast, offers a *platform* where users can deploy and run some kind of self-written software. Thereby, the programming language depends on the platform which is typically designed for scalability. One example for PaaS is the Google Apps Engine [22]. Finally, SaaS represents the provision of a single application to customers for on-demand use as a service. Users have usually no or only very restricted possibilities for customization. Normally, such a software service can be accessed via standard web browsers. Salesforce.com, for example, offers a customer relationship management (CRM) software as a service based on a pay-per-use model. In summary, IaaS, PaaS, and SaaS are considered as subsequent abstraction layers with an increasing degree of abstraction to the executing infrastructure [2]. Thereby, mechanisms to ensure security and data privacy are crucial at each layer in a Cloud environment.

Cloud computing has several properties that make it very attractive for SMEs and start-up enterprises. Most prominently, these are low entry costs, high scalability, and the possibility to outsource IT systems and services into Cloud infrastructures that can guarantee a quality of services that the original enterprises could not provide on their own. Hence, enterprises can concentrate on their core business without dealing with complex IT infrastructures directly.

Besides running the various information systems and services, their integration, referred to as Enterprise Application Integration (EAI), is also a critical issue for many enterprises. EAI provides means to map business processes to the technical system level. One very popular concept to achieve this is service orchestration in service-oriented architectures (SOA). Web service technologies are commonly used to create SOA since they enable service orchestration and hide the underlying technical infrastructure.

Many small and medium enterprises (SMEs) employ consultant services to identify their key business processes and to build integrated IT

environments by the deployment of in-house integration platforms. But often, the maintenance and operation of such platforms is cumbersome. This brings up the idea of Orchestration as a Service (OaaS), where an integration platform in the form of a service orchestration engine is hosted and maintained by a so-called Cloud provider. To optimize resource utilization such an orchestration engine should be designed as a multi-tenant architecture, meaning that different users deploy und run workflows on shared machines. In this context, security and privacy of both the deployed workflows and their data have to be considered as key issues.

Generally, Cloud computing offers are based on Web service technologies, depending on the provider. Furthermore, Grid computing middleware – closely related to IaaS in Cloud computing and mostly used in scientific environments – and service orchestration middleware are nowadays as well based on Web services. The idea to combine these in order to create an OaaS infrastructure seems promising. In the BIS-Grid project, our major objective is to proof that Grid and Cloud 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.

This article presents two major contributions: At first, we describe our idea of a Cloud orchestration service and list the requirements we identified especially with respect to security. Secondly, we introduce the BIS-Grid Engine as a standard-based core middleware for such an OaaS infrastructure. We focus on the architectural design and security standards we adapted from the Grid domain to show the appropriateness of this implementation.

2. ORCHESTRATION AS A SERVICE

Integration is a continuous challenge for both industry and research in order to enable the seamless interaction of (heterogeneous) IT systems and services. Modern integration solutions adopted the service-oriented architecture (SOA) design paradigm in which the workflow-oriented representation of business logic is an inherent part. This means that all applications are encapsulated by enclosed, loosely coupled, often low-level services which are composed as service orchestrations to create (executable) business processes.

The Web Services Business Process Execution Language (WS-BPEL) [3] is an OASIS standard that is able to orchestrate Web services as well as to offer each orchestration itself as a Web service. This enables the use of lower-level services and processes to build complex services on a higher level, mapping a value added chain partially or in whole to the technical system level. In the course of time, WS-BPEL turned out as the *de-facto* standard for service orchestration with broad support from industry.

The use of loosely coupled services in SOA allows to dynamically switch the location of invoked services, for example to use services offered by an external provider instead of local services. Thus, SOA can be considered as an enabling technology to outsource IT systems and corresponding services to reduce IT costs. Most of today's Cloud services adopted the SOA paradigm that can be seen as the basic requirement for Clouds.

Microsoft BizTalk Server and SAP XI are two examples of commercial integration platforms applicable to SOA integration, using WS-BPEL. In order to run such platforms, 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 operate such infrastructures although they certainly have needs for integration. Even freely available products like Sun's OpenESB are hard to set-up and maintain in a productive manner. Hence, the trusted orchestration of services is a major concern in Cloud computing.

Within the BIS-Grid project, one goal is to combine WS-BPEL, the de facto standard for service orchestration, and Grid technologies with its comprehensive security mechanisms to provide a secure integration platform as *Orchestration as a Service* (OaaS), according to the "*-as a Service" paradigm. Thereby, we define OaaS as a specialization of PaaS, since process developers are able to develop, deploy and manage custom business processes which can be considered as a kind of self-written software.

To successfully provide an orchestration engine as a commercial Cloud service, the implementation must address several requirements of the business domain to prove that OaaS represents a cost-saving and secure option for IT system integration. Hence, besides the functionality to **execute service orchestrations** denoted as workflows, there are several non-functional requirements that must be taken into account when developing an implementation for OaaS.

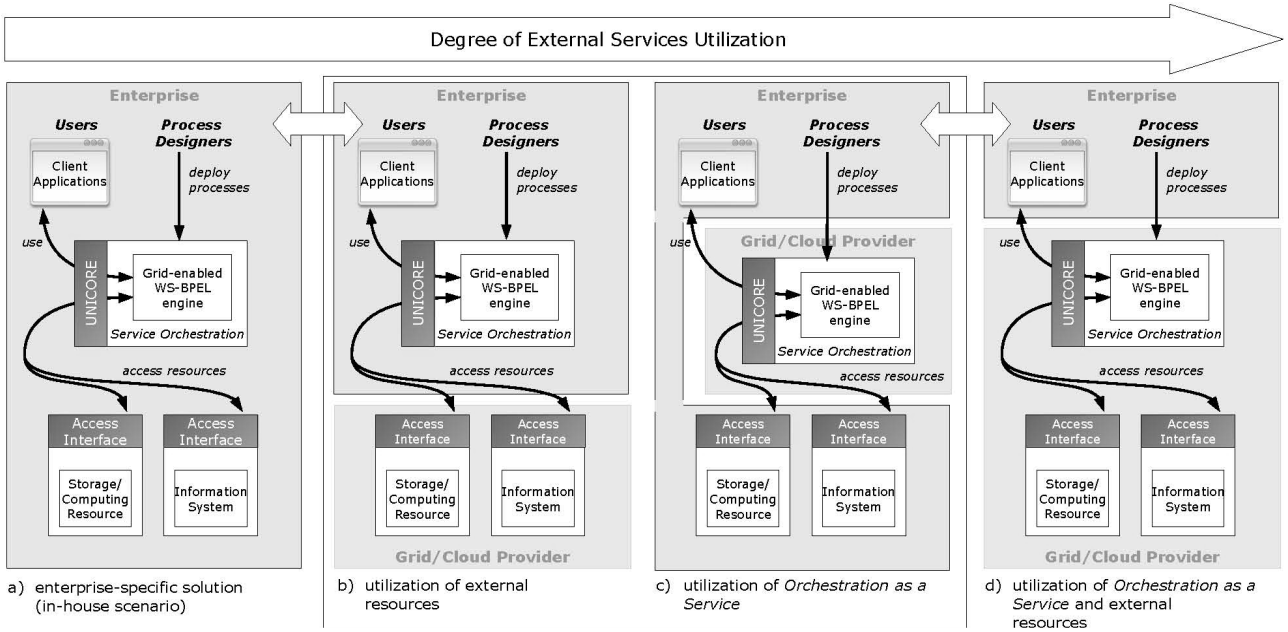


Fig. 1 – Application scenarios for workflow execution

The major requirement is **security**, comprising access control and activity monitoring as well as data privacy. Typically, a workflow combines services that are provided by several information systems, and business-crucial or confidential information is passed through the workflow engine. This information must be secured from unauthorized access. Therefore, business-applicable authentication and authorization mechanisms that support **role-based** and **fine-grained access control** mechanisms are essential. Moreover, a detailed **monitoring** of user activities on accessed information systems is necessary, regarding that workflows may act on behalf of users and a consistently access log is required due to legal issues. Because Cloud computing and information system integration middleware are based on SOA, we require that OaaS infrastructures must be seamlessly capable of being integrated with **SOA**. Interfaces should be provided as services including the deployed workflows.

For efficient resource utilization, we require that OaaS infrastructures must be designed as **multi-tenant** infrastructures, meaning that workflows of several companies or users must be executable on the same resource without affecting each other and still guaranteeing data privacy and other relevant qualities of services. Furthermore, workflow **deployment must be simple**, and the execution of workflows must be **scalable and high-performing**. Finally, we require conformance to the **WS-BPEL** specification, since WS-BPEL is the de facto industry standard for service orchestrations.

Beside OaaS we considered further deployment scenarios depending on the degree of Grid/Cloud utilization of the involved components, see Figure 1. This ranges from (a) a classical in-house scenario, in which all components are deployed within one organization, up to (d) a complete Grid/Cloud scenario, in which all components except for the client applications are outsourced to some Grid or Cloud infrastructure. The scenario (b) is an alternative to the OaaS scenario depicted in (c), in which conventional components such as storage and computing resources and the actual information systems are outsourced partially or in whole.

The evaluation of the OaaS approach is based on two scenarios with our industrial partners, both SMEs. Currently, in these scenarios we regard the in-house scenario (a) and the OaaS scenario (c). Outsourcing information systems to a Grid or Cloud provider is dismissed because of serious security concerns. The overall philosophy is to keep all data services in-house and relevant data for orchestration and integration may leave the company on demand under ensuring security, privacy and a trustworthy relationship to the service provider.

3. ARCHITECTURE

The BIS-Grid Engine is our core middleware for an OaaS infrastructure, based on widely-used standards that are mostly adopted from the Grid domain. The engine is designed to address most of the above identified requirements.

Grid middleware usually provides access to compute resources via stateful Web services, also called Grid services, based on the Web Services Resource Framework (WSRF) [4]. Grid middleware also provides comprehensive security mechanisms to enable authentication and authorization. UNICORE 6 (www.unicore.eu), for example, supports X.509 certificates for authentication and provides an authorization mechanism that is based on the distinguished name of the user certificate with only a

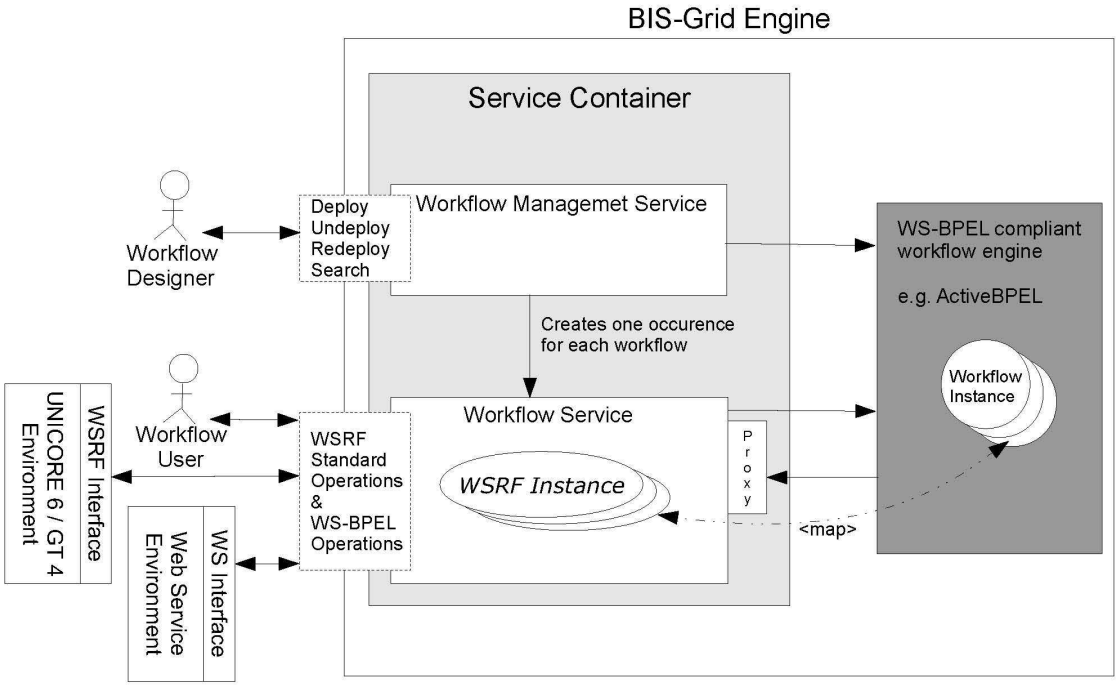


Fig. 2 – BIS-Grid Engine Architecture

very restricted role mapping. States of WSRF-based Web services are represented by service instances and stored in so called *Resource Properties*. Such properties can be accessed or manipulated by corresponding WSRF methods.

The BIS-Grid Engine is based on the UNICORE 6 middleware and provides service extensions for workflow deployment and execution. Thus, our service extensions make use of the security mechanisms of UNICORE 6. Although this provides an adequate basis to enable secure workflow execution over the internet, more effort was needed to fulfill the OaaS requirements we discuss in Sec. 4. Figure 2 gives an overview of the architecture of the BIS-Grid Engine. The key concept is to use an arbitrary WS-BPEL engine for workflow execution encapsulated by service extensions to UNICORE 6. All communication must pass these services resulting in a secured WS-BPEL engine that can only be accessed with the UNICORE 6 layer. The WS-BPEL engine is loosely coupled via standard HTTP connections and can easily be exchanged if the WS-BPEL standard changes or a newer version of the engine is available. In general, the BIS-Grid Engine enables the interoperability between WS-BPEL and external services through the support of (stateful) WSRF Web services and advanced security mechanisms that are discussed separately in Sec. 4.

The UNICORE 6 service extensions that are mentioned above are the *Workflow Management Service* and the *Workflow Service* that are both realized as stateful services. The Workflow Management Service is statically deployed and provides methods for workflow management (i.e. deployment, redeployment, and undeployment of workflows). For deployment, a deployment package with all necessary information is send to a service instance. During the deployment process, a specialized version of the generic Workflow Service is created on-demand and hot-deployed into the service container. Thus, each deployed workflow in the WS-BPEL engine has a corresponding Workflow Service, and therewith each workflow is represented through an independent and separate stateful service. Thereby, the Workflow Service provides the same interface as its WS-BPEL workflow counterpart. The progress of workflow execution and additional configurations such as security credentials (see Sec. 4) are exposed by the corresponding Workflow Service *instance* as resource properties according to the WSRF standard. Since each workflow execution in the WS-BPEL engine is also regarded as an instance, each workflow execution is represented by two instances, one Workflow Service instance in the UNICORE 6 service container and one workflow instance in the WS-BPEL engine. Both instances are mapped to each other automatically based on exchanged messages [13, 14].

Message exchange for workflow execution is based solely on SOAP messages and must pass the services in both directions (ingoing and outgoing). Outgoing messages that are initiated by the WS-BPEL engine (i.e. the invocation of external services) must be sent to a HTTP proxy that is running in the UNICORE 6 container. It is ensured that outgoing messages are forwarded to the correct Workflow Service instance, which stores the information that is necessary for service invocation.

Then, the Workflow Service instance performs the external service call. This is especially important due to security concerns since certain credentials may be used for specific service invocations. These credentials can be configured both globally at design time for one single workflow and locally at runtime for one single Workflow Service instance.

For workflow management functions like workflow (un)deployment or workflow monitoring an engine-specific adapter is used. Each WS-BPEL engine has its own engine adapter. Currently, the ActiveBPEL workflow engine is supported. The implementation of an adapter for the Java Business Integration (JBI) Platform OpenESB that includes the BPEL SE workflow engine is in progress.

4. SECURITY

Security is one of the most important issues within Cloud environments. Communication security, access control, and data privacy have to be considered, especially when message are exchanged over the internet. Grid security mechanisms can be regarded as a basis for secure OaaS because of their focus on modern security standards.

Grid security is based on personalized X.509 certificates issued by a Certificate Authority (CA) or an associated Registration Authority (RA). Each person that intends to participate in a Grid must trust this CA. Access permissions are granted as the membership in a so-called virtual organization (VO) that is maintained in separate VO management systems. This scenario is neither applicable for Clouds nor the business domain. A company, possibly having several hundred employees and high employee fluctuation, cannot send each new employee *personally* to a RA in order to get them a personalized certificate. Furthermore, rights should be bound to business roles from existing identity management systems without introducing an additional system for VO management. These facts require a distributed identity management system that recognizes local identity management systems with the possibility to grant or revoke role-based permissions in a short time for an OaaS scenario.

SAML [5] is a standard that is capable to encode arbitrary attributes, such as roles and affiliations, into a so-called assertion. Such assertions can be issued by an identity management system. SAML also provides the capability to express fine-grained credential delegation rights, for example, to express that an entity is allowed to process an activity on a resource in a well defined time frame. This standard is appropriate to transport enriched user information from an identity management system to the OaaS infrastructure.

For describing rights, the BIS-Grid Engine uses XACML Policies [6] that are very flexible and allows fine-grained access control. Access decisions are described as rules that define the applicability of a rule by means of the user, targeted resource, and the desired action. Furthermore, conditions can be added that express dependencies between these elements. The result of a rule is to either deny or grant access to the resource. Users are identified by the sum of all attributes included in a SAML assertion or/and via additional attributes requested during the authorization process. Hence, the policy designer must be aware of the organizational structures and roles of the enterprise to describe the access rights correctly.

As a recommendation for an appropriate distributed identity management system with low maintenance costs and the capability to integrate different identity management systems such as Active Directory or OpenLDAP we suggest Shibboleth [23]. In combination with Grid-Shib the system is able to automatically issue short-lived certificates (SLC) together with a SAML assertion that includes the user's business roles. Welch et al. describes such an architecture in more detail [7] and a Short Lived Credential Service (SLCS) that issues SLC is presented by SWITCH [16].

The major advantage in this security architecture is the seamless integration of existing identity management systems with the Shibboleth system, also providing Grid compatibility based on SLC with GridShib. An internally hosted and maintained identity management system can obtain and check credentials as well as supply attributes, so-called campus attributes. An SLC usually has a short lifetime of at most one million seconds (approx. 11 days), so that all roles and the connected rights are invalid after this period. It is technically also possible to reduce the lifetime of an SLC, for example, to one day. This guarantees that roles and permissions are only valid for a short time and must be renewed every workday, which is especially important since a mechanism to revoke invalid SLCs is missing in this architecture.

Fig. 3 depicts the authentication and authorization process of the BIS-Grid Engine. The certificate is used to establish an SSL connection between the user and the BIS-Grid Engine so that transport layer security as well as user authentication is guaranteed. All requests must pass the Policy Enforcement Point (PEP) before reaching the actual service. The PEP asks the Policy Decision Point (PDP) for a decision that is based on the inserted XACML rules. The attributes that are included in the attached SAML assertion are used to authorize the user. Only when access is granted, the request reaches the service instance.

Hot-deployment of the Workflow Service also

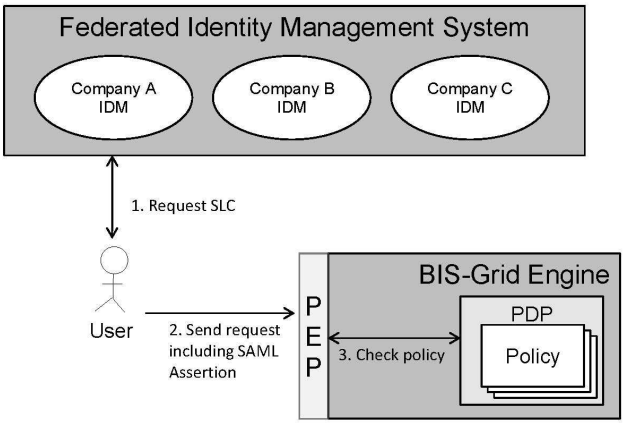


Fig. 3 – Security Architecture Overview

demands hot-deployment of security policies. We established such a hot-deployment mechanism for XACML policies by adding new rules to the PDP. However, the possibility to add new policies also brings up the threat of misuse, for example, by adding policies that affect other services. Hence, we limit newly inserted XACML rules so that the according rule set can only affect the corresponding Workflow Service.

After discussing authentication and authorization, we have to consider privacy, too. 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. The system has also to distinguish between affiliations or even departments during information retrieval operations for deployed workflows or for running workflow instances. XACML policies are not applicable for this issue since they restrict service method invocations but not the content of an invocation result. We established means to filter information according to the authentication attributes when discovery operations are used. 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 queries for deployed workflows, the BIS-Grid Engine will only show workflows matching the same affiliation and business role. Both must be included in the signed SAML assertion the user presents with his SLC. A similar filter also guarantees privacy for searching for instances of a Workflow Service.

In addition we have to regard how to configure security for invoking Web, Grid, and Cloud services from the BIS-Grid Engine. At the moment, we support the invocation of standard Web services without advanced security, UNICORE 6 services with certificate-based authentication and SAML-based authorization, as well as Globus Toolkit 4 (GT4) services that are secured via proxy certificates and Web services security standards like WS-Trust or WS-Secure Conversation. For each used service, security credentials can be configured separately. We can use Credential Delegation via Proxy Certificates (GT4) or SAML assertions (UNICORE), invoke services with a BIS-Grid Engine certificate or another previously provided certificate, or just username and password. Since most Cloud services use proprietary security mechanisms, possibly some customizing will be necessary, if a new Cloud service should be supported.

To sum up, the BIS-Grid Engine provides high-level security on all layers as required in Sec. 2. Besides the technical and organizational issues discussed in this article, this topic also involves legal issues to be addressed which are not part of the BIS-Grid project.

Due to simplicity of the exemplary evaluation, we decided not to set up a complete Shibboleth environment for our OaaS prototype. We use a similar solution that also supports SAML assertions but lacks in using a federated identity management system. The *UNICORE Virtual Organisations System* (UVOS) allows the administration of user identities combined with arbitrary attributes for each identity. Additionally, hierarchical organizations can be mapped to hierarchically organized groups and attributes can be attached to groups and therewith to all members of a group. Groups or sub-group members and attributes can be managed by different administrators. All UVOS-managed information can be queried by SAML2-compatible applications [8]. Responding to such a query, UVOS answers with a signed SAML assertion including all attributes (group affiliations, group attributes, and global attributes). The combination of UVOS and UNICORE 6 is fully integrated and well-tested in the Chemomentum project, in which UVOS was developed originally.

5. RELATED WORK

There are several upcoming and new projects regarding service orchestration in Cloud environments, proving the relevance of the OaaS paradigm. Unfortunately, these are all new projects and aim at commercial issues and hence the infrastructures are not described in detail. In the following, we present some of these commercial projects but also regard related work concerning service orchestration in Grid environments.

Microsoft recently started to provide the .NET

Workflow Service as part of the .Net Services of the Azure Services Platform [18] 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 SaaS paradigm as well as Microsoft's *Software-plus-Services* strategy. The provided services can also be regarded as OaaS, thus highlighting our efforts to be relevant in commercial contexts.

CSC [19] 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 their Trusted Cloud Services brand. Unfortunately, there is very little information about the concrete services, their realization, and the offered service level agreements.

Cordys [20] also promotes Cloud-based service orchestration, called *Enterprise Cloud Orchestration*. 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 [21] project provides workflow extensions for UNICORE 6, consisting of two UNICORE 6 service containers. The first container represents a workflow engine that processes workflows on a logical level. The second container represents a service orchestrator that transforms so-called Work Assignments into jobs, given in the Job Submission Description Language (JSDL) [12]. 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 [9] Amnuaykanjanasin and Nupairoj present a solution to orchestrate Globus Toolkit services secured with the Grid Security Infrastructure (GSI). A proxy implementation is generated automatically for each Grid service when the user requests one. To support 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 [10, 11]).

6. CONCLUSION

We discussed our Orchestration as a Service paradigm (Oaas) as a specialized form of Platform as a Service (PaaS). OaaS decreases the start-up costs to apply information system integration by outsourcing the operation and maintenance of a workflow engine to an OaaS provider. Such a provider possesses both the know-how and the resources to provide orchestration services with the corresponding quality of services with lower costs than small or medium enterprises can provide on their own. Additionally, we presented four scenarios with different degrees of outsourcing services to a Grid or Cloud infrastructure.

We also presented the BIS-Grid Engine as an example of a technical core system of an OaaS infrastructure that considers most of the necessary requirements especially with respect to security. Building on well-accepted standards from industry and scientific computing (namely Web service technologies, WS-BPEL for service orchestration, and security standards such as X.509 certificates, Short Lived Certificates, SAML, and XACML for authentication and authorization), we constructed a multi-tenant, OaaS-capable workflow engine that supports role-based access control.

An open issue is to investigate the scalability of the system. In Clouds, architectures must scale for several thousand users and workflows at the same time. In [15], we present some ideas about load balancing with an early version of the presented engine. In general, the loose coupling of the WSRF Proxy of the architecture and the WS-BPEL engine offers various possibilities for load balancing approaches. However, concrete evaluation is future work.

The BIS-Grid Engine and the engine's documentation can be found on Sourceforge at http://bis-grid.sourceforge.net. The project deliverables (partly in German) are available on http://bisgrid.de.

7. ACKNOLEDGEMENT

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.

8. REFERENCES

[1] L. Youseff, M. Butrico, D.S. Da Silva. Toward a Unified Ontology of Cloud Computing.

Proceedings of Grid Computing Environments Workshop (GCE '08), 2008

[2] Luis Vaquero, Luis Rodero-Merino, Juan Caceres, Maik Lindner. A Break in the Clouds: Towards a Cloud Definition, *SIGCOMM Computi,. Commun. Rev. 39* (2009). pp 50-55.

[3] OASIS WS-BPEL Technical Committee. Web Services Business Process Execution Language (WS-BPEL) Primer. http://www.oasis-open.org/committees/download.php/23974/wsbpel-v2.0-primer.pdf

[4] OASIS WSRF Technical Committee. Web Services Resource Framework – Primer v1.2. http://docs.oasis-open.org/wsrf/wsrf-primer-1.2-primer-cd-02.pdf

[5] N. Ragouzis, J. Hughes, R. Philpott , E. Maler, P. Madsen, T. 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, Working Draft. 2007

[6] Tim Moses. eXtensible Access Control Markup Language (XACML) Version 2.0. 2005

[7] V. Welch, T. Barton, K. Keahey, F. Siebenlist. Attributes, Anonymity, and Access: Shibboleth and Globus Integration to Facilitate Grid Colloboration, *Proceedings of the 4th Annual PKI R&D Workshop*.2005

[8] K. Benedyczak. *UNICORE Virtual Organisations Service Overview*. TechReport. 2007

[9] Pichet Amnuaykanjanasin, Natawut Nupairoj. The BPEL Orchestrating Framework for *Proceedings of* "Secured Grid Services. *International Conference on Information Technology: Coding and Computing"*, Los Alamitos, USA 2005, pp. 348-353

[10] Andreas Hoheisel. User tools and languages for graph-based Grid workflows: Research Articles. *Concurr. Comput. : Pract. Exper. 18* (10) (2006). P. 1101-1113.

[11] Jia Yu, Rajkumar Buyya. A Novel Architecture for Realizing Grid Workflow using Tuple Spaces. *Proceedings of the 5th IEEE/ACM International Workshop on Grid Computing*. Washington D.C., USA 2004, pp 119-128

[12] Job Submission Description Language (JSDL) Specification, Version 1.0 http://www.gridforum.org/documents/GFD.56.pdf. 2005

[13] A. Brinkmann, S. Gudenkauf, W. Hasselbring A. Höing, O. Kao, H. Karl, H. Nitsche, G. Scherp. Employing WS-BPEL Design Patterns for Grid Service Orchestration using a Standard WS-BPEL Engine and a Grid Middleware *In "The 8th Cracow Grid Workshop"*, Cracow, Poland 2009 pp. 103-110

[14] S. Gudenkauf, A. Höing, G. Scherp. Catalogue of WS-BPEL Design Pattern. Techreport. https://bi.offis.de/bisgrid/tiki-download_file.php?fileId=269 2008

[15] S. Gudenkauf, W. Hasselbring, F. Heine, A. Höing, G. Scherp. O. Kao, Bis-Grid: Business Workflows for the Grid *In "The 7th Cracow Grid Workshop"* Cracow, Poland 2008 pp. 86-93

[16] Short Lived Credential Service (SLCS). http://www.switch.ch/grid/slcs/ [23.10.2009]

[17] Amazon Web Services LLC, Amazon Web Services. http://aws.amazon.com

[18] Microsoft Azure Homepage http://www.microsoft.com/azure

[19] Computer Science Cooperation, Cloud Orchestration Services, http://www.csc.com/cloud/ds/27357-cloud_orchestration_services

[20] Cordys Hompage, Enterprise Cloud Orchestration, http://www.cordys.com/cordyscms_com/enterprise_cloud_orchestration.php

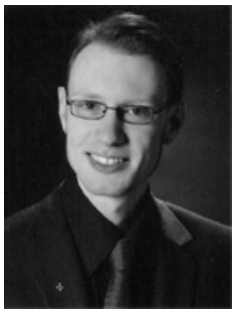
[21] Chemomentum: Grid-Services based Environment for enabling Innovative Research, http://chemomentum.org

[22] Google App Engine Homepage, code.google.com/appengine

[23] S. Cantor, Tom Scavo. Shibboleth Architecture. Techreport. http://open-systems.ufl.edu/files/draft-mace-shibboleth-tech-overview-latest.pdf

André Höing *is a research assistant at the Technische Universität Berlin. He is a member of the workgroup "Complex and Distributed IT-Systems" since 2007. Before this he was employed at the Paderborn Center for Parallel Computing for about 6 months. He got this Diploma degree in computer science from the University of Paderborn in 2006.*
He is currently working in the field of distributed service computing and service orchestration with WS-BPEL. Furthermore, he is interested in Cloud Computing technologies.

Guido Scherp *is a research assistant at the OFFIS Institute for Information Technlogy in Oldenburg, Germany. He received his diploma degree in computer science from the University of Oldenburg in 2005. His research interests include Grid and Cloud computing, scientific and business workflows, and model driven development.*



Stefan Gudenkauf *is a research assistant at the OFFIS Institute for Information Technology in Oldenburg, Germany. He received his diploma degree in computer science from the University of Oldenburg in 2006. His research interests include Business workflows and coordination models, parallel programming and model driven development.*