SEMANTIC-BASED DELIVERY OF E-SERVICES APPLYING THE SAAS APPROACH
Poster Paper

Agata Filipowska, Monika Kaczmarek and Tomasz Kaczmarek
Department of Information Systems, Faculty of Informatics and Electronic Economy
Poznan University of Economics, Al. Niepodleglosci 10, 61-875, Poznan, Poland

Keywords: Service composition, Service platforms, Software as a service.

Abstract: The cloud computing paradigm aims at sharing various resources that encompass infrastructure, software, application and business processes. This paradigm may affect the way companies develop and provide services and offers inter alia increased reusability, costs reduction and decreased time to market. However, its fulfilment requires a certain change in the approach to create applications that support business processes. This paper provides a vision of a platform enabling users to build applications out of services available in the cloud.

1 INTRODUCTION AND MOTIVATION

The cloud computing paradigm aims at sharing various resources that encompass infrastructure, software, application and business processes (Zhang & Zhou, 2009; Zhang et al., 2008). In the cloud, as indicated in (Armbrust et al., 2009), especially the software application resources i.e. services, are provisioned and consumed using the software as a service (SaaS) model. It affects the way companies develop and provision their services.

More and more often, companies use services distributed in the cloud and offered by various providers on various machines. To build a solution applying SOA with services available in the cloud, one should also use mechanisms such as discovery and composition of services. The vision of efficient service discovery and composition in the SaaS settings is very appealing offering increased reusability, costs reduction and decreased time to market. However, its fulfilment requires, in our opinion, a certain change in the approach to create applications that support business processes. In addition, in order to allow for collaboration between partners offering their services in the cloud and allow business to build their offer in a more dynamic way, an appropriate tool support, e.g., in a form of a platform, is needed.

Developing a software platform supporting the SOA architecture with functionalities such as discovery and composition of services, which would enable using complex services, requires addressing a number of challenges. As the issue of service discovery is well investigated, we focus on the composition mechanism and the platform itself.
The aim of this paper is to discuss the requirements for the composition mechanism in the cloud computing setting and sketch a model of a platform supporting this process. In order to fulfil this aim, the paper is structured as follows. First, we define requirements for the semantic-based composition being a core element of the platform. The following section describes an exemplary framework for implementing these requirements. The paper concludes with final remarks.

2 COMPOSITION–REQUIREMENTS

Although there is a large number of services available in the cloud, alone they have a limited utility from users' perspective, as they often fail to meet users' high level requirements. Therefore, the need to create and then provision composite services appears (Grossmann et al., 2011; Küster et al., 2005; Dustdar and Schreiner, 2005; Peer, 2005). A number of various approaches to static and dynamic (automated) service composition has been developed e.g., (Hoffman et al, 2008, Rao et al., 2006; Pistore et al., 2005; Benatallah et al., 2002). Each of these approaches has to face the environment, where the services are deployed. This environment usually has the following characteristics:

- There is potentially a large number (not comprehensible for a human developer) of services available that may become the building blocks for complex applications;
- Services may be both external to an enterprise and internal ones, which might result in certain security concerns or requirements towards the composed workflow; some services might be poorly documented;
- Services differ not only with respect to their functionality, but non-functional parameters, which has to be taken into account to deliver a sound composed service;
- Services may be described using different ontologies; even if they could be merged or mapped, the resulting ontology might be large.

Given the environment adhering to the assumptions outlined above, the requirements towards the composition algorithm may be formulated.

Req.1 Composition MUST be automated and therefore, it SHOULD operate on the formal description of a service.

Req. 2. Composition MUST deal with the formal description of the relevant domain (e.g., a background ontology). In addition to formal services' description there may be some additional facts to take into account during composition, which are expressed in ontologies used for service description or added during ontology mapping.

Req. 3. Composition MUST be scalable with respect to the complexity of the ontology (ontologies) used for service description. It is in general expected that the ontologies to be used for describing real world services can have hundreds or even thousands of concepts. If this would impact the composition algorithm heavily, it could render it useless.

Req. 4. Composition MUST be scalable with respect to the number of services.

Req. 5. Composition MUST be able to deal with a number of services that have the same semantic characteristics (the same pre/postconditions) but different non-functional parameters, or no non-functional parameters at all.

Req. 6. Composition MUST deal with non-functional requirements towards the effect of the composition.

Except from requirements towards composition, there are certain requirements that the composition algorithm poses against the environment, or more precisely, the service descriptions that it operates on:

1. Services have to be single, atomic, stateless operations or, if they are not stateless, this should be expressed accordingly on the semantic level.
2. Services have to be described in a standard format (or translatable standards).
3. Services have to be described in a single formalism, or clearly announce the formalism, which enables translation; in general different approaches to semantic-based composition assume certain logical formalism, to which the service descriptions have to adhere.
4. Services have to be described using the same ontology (or ontologies that can be mapped or merged).
5. Service descriptions have to include average / declared / promised non-functional parameters values, including pricing model.
6. Service repository has to be accessible. This requirement is implicit in the whole composition.
7. Services have to be searchable by semantic description.
8. Services have to be searchable by non-functional parameters.

From the all composition approaches, the dynamic, ontology-based approaches are the ones, that fulfil most of the requirements defined above.

Except from the composition, additional work might be required to bridge the technological gap between different Web services on a technical level. Another issue of concern might be the security context. In the cloud environment each service might require credentials, which have to be provided by the application, but are not taken into account during the composition - they have to be added during the manual plumbing. It is also assumed that appropriate business mechanisms are available within the platform which controls the execution of the composed service: billing for service usage, and monitoring of non-functional parameters during service execution. Payment and Service Level Agreement negotiation, signing and fulfilment need to be set in place and they are handled externally to the composition environment, and are not considered to be part of a business process that composition is used for. The error handling, recovery and reliability mechanisms for the whole business process solution have to be in place and are not handled usually by the composition algorithms themselves, nor by the business processes that are composed.

3 CONCEPTUAL FRAMEWORK FOR DELIVERY OF E-SERVICES IN THE SAAS PARADIGM

The platform supporting enhanced delivery of services using the software as a service approach, should address challenges described above. Therefore, numerous requirements including functional or non-functional should be met.

After a careful study and benefiting from our experience from research projects dealing with SOA and service composition, we defined the following requirements for the platform:

1. Exposing services to platform users enabling for services discovery, enactment, monitoring, etc. This includes exposing services using SaaS model.
2. Discovery of services taking into account semantic descriptions of services and allowing for finding services matching functional and quality requirements defined by users.
3. Semantic-based composition of services taking into account functional and non-functional features of services. These services may be also exposed by platform-external providers and available in the cloud.
4. Reasoning based on the user-provided ontology, that is to be used mainly while composition and discovery of services.
5. Support for ontology development and maintenance enabling for ontology evolution and linking to new ontologies and knowledge bases being made available in the LOD cloud.
6. Billing and SLA management to assure proper service delivery and payment.
7. Enactment of services and composed services.
8. Execution of services and composite services.
9. Monitoring of execution the services and composite services.

In addition, the platform should conform to the current quality standards for software development. Among of non-functional requirements worth underlining are:

1. Accordance with the SOA principles to ensure extensibility of the platform.
2. Development using available and open-source components under a proper license model.
3. Scalability, especially with regard to the composition and discovery of services.
4. Security and privacy model. The platform should provide a mechanism for user authentication and should be able to generate digital certificates with that information.

These requirements led to proposing the platform framework that is discussed in the section below.

3.1 Platform Proposal

The central architectural component of the platform to address previously described challenges is the service bus (SB) that enables communication between architecture components. Its main feature is provision of the communications layer being a message-oriented middleware infrastructure capable of supporting a variety of industry-standard access mechanisms and protocols. Other components will communicate over the bus by sending and receiving messages. The SB should also handle external Web service invocations and incoming messages as all services (including also composite services) will be exposed in the SaaS model.

Other important components include:

- Composition component ensuring services composition based on ontological descriptions,
- Discovery component that retrieves services fulfilling criteria specified as service goals (in functional and nonfunctional terms),
- Reasoner that provides logic-based inference engine capable of reasoning with ontologies describing services and user goals,
- Monitoring service that while monitoring the execution of services provides data regarding achieved results and quality parameters achieved, that may be further used while billing users for usage of composite services and providing overview or changes to values of NFPs of services,
- Ontology evolution component supporting ontology storage and change management,
- Execution components including Execution Engine (EE) and Service Execution Engine (SEE). SEE responsibility is to enable execution and interoperability of services, while EE is to deal with semantic descriptions of services (and proper enactment of atomic services in composite services).

Finally, also an Advanced User Interface for platform administration and maintenance should be provided to enable a user to initially define his profile, and provide access to platform functionalities such as ontology management, services discovery and composition.

3.2 Implementing the Platform

Such a platform may be developed from a scratch or adopting already existing open source components. We indicated the following components fulfilling the requirements and available under a licence that would enable for their application within a research project. These components include:

Service Bus Implementation: Petals ESB (Petals ESB, 2011): being a distributed open source implementation of Java Business Integration (JBI) specification.

Composition Components: composition components are usually not publicly available as separate components but offered as a part of integrated Web Service platforms, e.g. WSMX (Web Services Execution Environment, 2011).

Execution Engines: Apache ODE (Apache, 2011) – a WS-BPEL compliant web services orchestration engine that needs to be semantically extended in order to support execution of Semantic Web services as done e.g. in (SUPER, 2011).

Web Services Execution Infrastructure: the message exchange between services may be ensured using e.g. Apache Axis. In addition, one also needs a tool such as Virtuoso Universal Server assuring data storage and providing web application server.

Ontology Storage: it depends on the ontology specification language e.g. for OWL-S or RDF, SESAME may be used.

Advanced User Interface: The user interface must be designed to support user interaction with the platform, such as specification of a composition goal, update of an ontology, etc. An example of such advanced user interface is proposed by the WSMO Studio, where a user may edit an ontology, design a process, describe a service in terms of desired functionality, launch composition or enactment.

4 SUMMARY

In order to realise the full potential of service oriented applications and deliver the e-services in a SaaS model, it is essential that the platform for application developers and business analysts is available, that allows for service discovery, composition, execution and monitoring. The goal of this paper is to present the general considerations and requirements towards such platform, and currently existing solutions that form a base for such platform. Much has been done to address various requirements and challenges that are posed by the service oriented environment. There exists solutions for virtually all aspects of the postulated platform, they are however not coherent, or overlap only to some extent. Judging by the involvement of the software industry, the solution (or a number of them) will arise on the border between research projects and industry platform, where state-of-the-art research results will be included in strong industry frameworks and will gradually re-shape the way applications are build, and business processes are carried out.

REFERENCES

Apache ODE, http://ode.apache.org/, last accessed 08.04.2011.

Armbrust, M., Fox, A., Griffith, R., Joseph, A.: R. Katz, Konwinski, Lee, A. G., Patterson, Rabkin, D. A., Stoica, I., Zaharia, M.: Above the Clouds: A Berkeley View of Cloud computing. *Technical Report. University of California at Berkley, USA, 2009.*

Benatallah, B., Dumas, M., Sheng, Q. Z., Ngu, A. H. H. Declarative Composition and Peer-to-Peer Provisioning of Dynamic Web Services. *In Proc. 18 th International Conference on Data Engineering, 2002.*

Dustdar, S. and Schreiner, W. ,'A survey on web services composition', *Int. J. Web and Grid Services, Vol. 1, No. 1, pp.1–30, 2005.*

Grossmann, G., Thiagarajan, R., Schrefl, M., Stumpter, M. Conceptual Modeling Approaches for Dynamic Web Service Composition. In: The Evolution of Conceptual Modeling, *LNCS 6520, pp. 180-204, 2011.*

Hoffmann, J., Ingo Weber, James Scicluna, Tomasz Kaczmarek, and Anupriya Ankolekar. Combining scalability and expressivity in the automatic composition of semantic web services. *In ICWE-08: Proceedings of the 8th International Conference on Web Engineering 2008, July 2008.*

Küster, U.; Stern, M. & König-Ries, B. A Classification of Issues and Approaches in Automatic Service Composition Proc. of the First International Workshop on Engineering Service Compositions *(WESC'05), 2005, 25-33.*

Peer, J. Web Service Composition as AI Planning - A Survey. Univ. of St. Gallen, Switzerland, 2005.

Petals Enterprise Service Bus, http://petals.ow2.org/, last accessed 08.04.2011.

Pistore, M., Traverso, P., and Bertoli, P. Automated Composition of Web Services by Planning in Asynchronous Domains. *In 15th Intl. Conference on Automated Planning and Scheduling, 2005, pp. 2-11.*

Rao, J.; Dimitrov, D.; Hofmann, P. & Sadeh, N. A Mixed Initiative Approach to Semantic Web Service Discovery and Composition: SAP's Guided Procedures Framework Proceedings of the 2006 I*EEE International Conference on Web Services (ICWS 2006), Chicago, USA, September 18 - 22, 2006.*

SUPER project, http://www.ip-super.org, last accessed 08.04.2011.

WSMX, http://www.wsmx.org/, last accessed 08.04.2011.

Zhang, L., Zhou, Q.: CCOA: Cloud Computing Open Architecture. *In: IEEE International Conference on Web Services, pp. 607–616, 2009.*

Zhang, L. J., Chang, C. K., Feig, E., Grossman, R., Panel, K.: Business Cloud: Bring the Power of SOA and Cloud Computing. *In: IEEE International Conference on Service Computing, 2008.*