WSMX 1.0: A Further Step toward a Complete Semantic Execution Environment

Federico Michele Facca, Srdjan Komazec, and Ioan Toma

Semantic Technology Institute (STI) – Innsbruck
ICT Technologiepark, Technikerstrasse 21a, 6020 Innsbruck, Austria
firstname.lastname@sti2.at

Abstract. The Web Service Execution Environment (WSMX) project is a continuously ongoing effort that aims at delivering a middleware covering all the Semantic Web Services life cycle. WSMX represents the reference implementation of the Semantically enabled Service Oriented Architecture (SESA) [1]. In this demonstration we aim to present the latest achievements that include: Web Service monitoring, Web Service ranking and Web Service grounding.

1 Introduction

The Web Service Execution Environment (WSMX) [1] is an open-source development and execution environment which implements a significant part of the SESA facilities. It is a platform characterized by strong component decoupling, goal-driven Web service usage and direct support for mediation facilities. It is a reference implementation of the Web Services Modeling Ontology (WSMO) [2] which is a conceptual model used to describe various aspects related to Semantic Web Services (SWS). WSMO is based on four major fundamental elements: Ontologies, Web services, Goals and Mediators. WSMX can achieve the goal of the user by dynamically discovering and selecting a matching Web service (where both Web services and goals are described in terms of ontologies) relying on data and process mediation facilities in various phases of the SWS life cycle. In this demo we show some of the most interesting innovations introduced in WSMX 1.0:

- monitoring for Semantic Web Services and the SESA environment will enable fault handling and quality of service monitoring;
- ranking engine to rank discovered services according to user preferences;
- service grounding engine based on W3C SAWSDL recommendation[1].

This paper is structured as follows: Section 2 presents the overall architecture of WSMX. In Section 3 we present our latest achievements toward Web Services and SESA monitoring. Section 4 presents the ranking component. In Section 5 we give an overview of our Web Service grounding solution. Finally in Section 6 we draw conclusions and describe the demonstration plan.

[1] http://www.w3.org/TR/sawsdl/

L. Aroyo et al. (Eds.): ESWC 2009, LNCS 5554, pp. 826–830, 2009.

2 WSMX Architecture

In order to provide for the complete Semantic Web Service life cycle support, the WSMX platform consists of several components with clear functional responsibilities [1], as depicted in Figure 1. The *WSMX Manager Core* component is the central part of WSMX, managing all other components and interactions between them. The Core is responsible for a number of the tasks such as the business logic of the system, the events engine, the internal workflow engine, and the loading of distributed components. The *Resource Manager* offers an interface for WSMX persistent storage facilities for WSMO artifacts. The *Discovery* component is concerned with finding Web service descriptions that match the goal specified by the service requester. In case there is more than one service capable of fulfilling the requester's goal, the *Ranking* component ranks them based on non-functional properties like price, guaranteed response time, etc. The *Data Mediator* component has the role of reconciling the data heterogeneity problems that can appear during discovery, composition, selection or invocation of Web Services based on mapping rules retrieved from the resource manager. Mapping rules have been created in advance at design time. The *Communication Manager* component is responsible for dealing with the various protocols for sending and receiving messages to and from WSMX. It represents the entry point to the system for external entities that want to consume WSMX functionality. The *Choreography* component defines the communication patterns in terms of messages exchanged in order to interact with the Web service. The *Grounding* component enables lifting from SOAP messages to WSMO ontologies and lowering from WSMO ontologies to SOAP messages. The *Invoker* component handles communication between WSMX and external Web services.

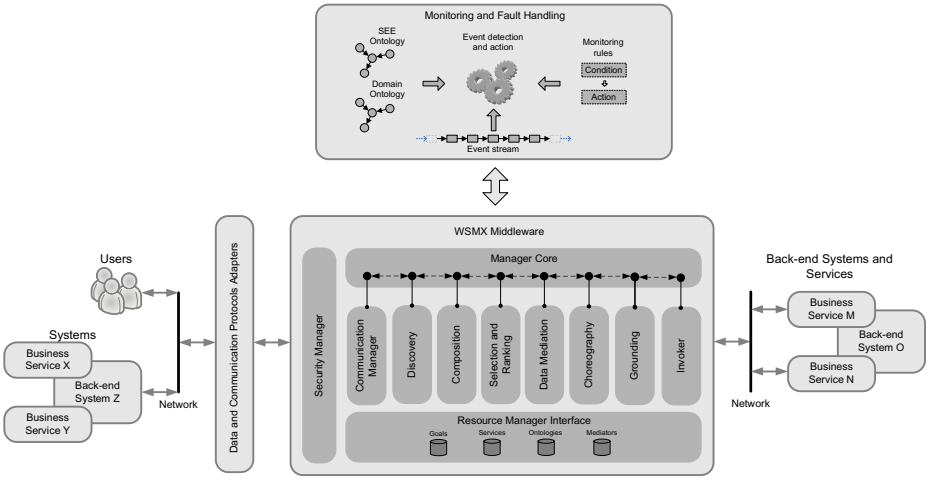

Fig. 1. WSMX architecture

3 Web Service Monitoring

Semantic Web Services monitoring is a relevant research issue. In particular, Vaculin et al. [3] presented a solution based on OWL-S as the foundational SWS ontology. The authors are covering only choreography execution while our solution tends to tackle the overall SWS lifecycle. Additionally, Pedrinaci et al. [4] introduced a set of ontologies for business process analysis and gave outlines for the semantic process mining and monitoring in the form of sequence which consist of 5 steps: observe, evaluate, detect, diagnose, and resolve.

Figure 1 gives a brief architectural overview of the WSMX monitoring and fault-handling subsystem. As suggested by the figure, all the WSMX middleware components are appropriately instrumented, thus enabling detection and extraction of low-level events and generation of the semantically-enhanced event descriptions. These descriptions are defined in the context of the domain ontology which formalizes notions of the problem domain, but also the SEE ontology which specifies structure and relationships inside WSMX. Based on the event streams produced by the monitored objects, as well as observational goals expressed in the ECA compliant rule-based monitoring language (inspired by the work of Behrends et al. [5] and built on top of the WSML-Rule), the solution will employ event-detection and reasoning techniques in order to ensure the reactiveness of the system in a timely fashion. Depending on the ECA rules definition the subsystem is capable of dealing with a variety of issues such as fault-handling, verification of service compositions and monitoring and enforcement of the Service Level Agreements.

4 Web Service Ranking

Web Service Ranking is the process which generates an ordered list of services out of the candidate services set according to ranking criteria specified by the user. The candidate services set is constructed by an a priori discovery process. Various non-functional properties can be specified by the user and obtained from the goal description as ranking criteria. On the service side the requested non-functional property values are either directly specified in the service description or are provided (computed or collected) by a monitoring tool (see Section 3). Non-Functional Properties (NFPs) specified in goal and service descriptions are expressed using the WSML language, by means of logical rules using terms from NFP ontologies [6].

The core of the ranking process is the evaluation of the logical rules used to model NFPs of services. Additional data is required during the rule evaluation process. This data represents mainly user preferences and includes: (1) which NFPs user is interested in, (2) the level of importance of each of these NFPs, (3) how the list of services should be ordered (i.e. ascending or descending) and (4) concrete instances extracted from the goal description.

In a nutshell the multi-criteria ranking algorithm works as follows. First a set of tuples containing nonfunctional properties and their associated importance is

extracted out of the goal description. If no importance is specified the default value is consider to be 0.5 which specifies a moderate interest in the nonfunctional property. The importance is a numeric value ranging from 0 to 1, where 1 encodes the fact that the user is extremely interested in the nonfunctional property and 0 encodes the fact that the nonfunctional property is not of interest. Later on, instance data from the goal is extracted and a knowledge base is created. The last step in extracting relevant information for the ranking process is to identify how the results should be ordered i.e. ascending or descending. Once the preprocessing steps are done, each service is checked to see if the requested nonfunctional properties specified in the goal are available in the service description. In case of a positive answer the corresponding WSML logic rules are extracted and evaluated using a reasoning engine. A quadruple structure is built containing the service, the nonfunctional property, the computed value and its importance. An aggregated score is computed for each service by summing the normalized values of NFPs weighted by importance values. Finally the score values are sorted according to the ordering sense extracted from the goal and the final list of services is returned.

5 Web Service Grounding

Grounding is a fundamental step in the Semantic Web Service life cycle that enables the communication between the semantic layer and and the back-end services. Several ongoing initiatives like SAWSDL [7], and microWSMO attend to tackle this problem. In WSMX, to be able to provide a flexible solution to the grounding problem, we designed and implemented an extensible grounding framework. The framework, according to the different types of service to be invoked (e.g. WSDL, RESTful, ...), is capable of selecting the correct grounding engine. The grounding engine is determined by evaluating the grounding endpoints in the WSMO Web Service descriptions. The current version, since WSMX platform concentrates mainly on WSDL services, includes only a grounding engine based on SAWSDL. To enable decoupling between remote services and WSMX, we implemented a mechanism to replace remote WSDL descriptions with local SAWSDL ones during the execution. This enables the use of the SAWSDL approach even if remote services are not annotated by creating a local annotated copy of them. The engine interprets the SAWSDL descriptions and, according to the XML Schema element to be sent or received, performs the required lowering (WMSO→XML) or lifting (XML→WSMO). Current lifting and lowering solution are based on XSLT scripts or on Java code (this allows for seamless integration of grounding code used in the previous WSMX releases). The introduction of a specific grounding component is a relevant step toward the evolution of WSMX in order to make it more flexible and dynamically configurable. Furthermore it easily enables the publication of WSMO goals as standard Web Services to enable easier B2B integration through the platform.

6 Demonstration Plan and Conclusion

In this paper we presented a set of new functionalities introduced in the new WSMX release. The new functionalities will be demonstrated at ESWC. In particular the demonstration is based on a scenario for the COIN project, which is dealing with automatic composition of product retailer services and shipment services: a company building aircraft needs to integrate components for the cockpit consisting of the various parts such as displays, sensors and actuators, data buses and processors. The company has specific preferences regarding the purchasing and shipping phase so as to minimize the overall price and to minimize the shipping time. The scenario aims to show how the new integrated components empower WSMX.

The demo will present to the visitor a Web interface to create a goal and associated ranking preferences. After the submission of the goal the visitor will be able to follow the behavior of the platform internals through the console. For a better understanding the demo will be supported by an animated architecture diagram.

Acknowledgments

The work presented in this paper is partially founded by FP7 COIN EU Project (216256). We would like to thank Jacek Kopecký for his contributions to the evolution of the grounding component.

References

1. Fensel, D., Kerrigan, M., Zaremba, M. (eds.): Implementing Semantic Web Services: The SESA Framework, 1st edn. Springer, Heidelberg (2008)
2. Fensel, D., Lausen, H., Polleres, A., de Bruijn, J., Stollberg, M., Roman, D., Domingue, J.: Enabling Semantic Web Services: The Web Service Modeling Ontology. Springer, Heidelberg (2006)
3. Vaculin, R., Sycara, K.: Specifying and Monitoring Composite Events for Semantic Web Services. In: Proceedings of the 5th IEEE European Conference on Web Services (November 2007)
4. Pedrinaci, C., Domingue, J.: Towards an Ontology for Process Monitoring and Mining. In: Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007), vol. 251, CEUR-WS (2007)
5. Behrends, E., Fritzen, O., May, W., Schenk, F.: Combining ECA Rules with Process Algebras for the Semantic Web. In: RULEML 2006: Proceedings of the Second International Conference on Rules and Rule Markup Languages for the Semantic Web, Washington, DC, USA, pp. 29–38. IEEE Computer Society, Los Alamitos (2006)
6. Toma, I., Roman, D., Fensel, D., Sapkota, B., Gomez, J.M.: A multi-criteria service ranking approach based on non-functional properties rules evaluation. In: Krämer, B.J., Lin, K.-J., Narasimhan, P. (eds.) ICSOC 2007. LNCS, vol. 4749, pp. 435–441. Springer, Heidelberg (2007)
7. Kopecký, J., Vitvar, T., Bournez, C., Farrell, J.: Sawsdl: Semantic annotations for wsdl and xml schema. IEEE Internet Computing 11(6), 60–67 (2007)