Reactive Sematic Execution Environment

Srdjan Komazec

Semantic Technology Institute (STI) Innsbruck, University of Innsbruck
ICT - Technologie Park Innsbruck, Technikerstrasse 21a, 6020 Innsbruck, Austria
`srdjan.komazec@sti2.at`
`http://www.sti-innsbruck.at`

Abstract. The power of Semantic Web Services (SWS) paradigm lies in the automation of the service life cycle thanks to the rich semantic descriptions. Currently available SWS Execution Environments (SEE) exhibit weaknesses in terms of autonomicity and reliability. The complexities and relationships between the typical SEE components deployed in a distributed environment and dependable on the external services introduce the necessity to introduce comprehensive observation and reactiveness mechanisms. In order to make SEE solutions more adaptive and reliable, this Ph.D. thesis proposes an application of the event-triggered reactivity approach based on Abstract State Machines.

Key words: semantic web services, event, reactivity, monitoring, ECA rules, abstract state machines

1 Problem statement

The stack of typical Web Service standards and protocols (like WSDL and SOAP) is supporting Web Service usage only by manual inspection and integration, i.e. existing service-based solutions are proving difficult to scale without a proper degree of automation. Tasks such as service discovery, ranking, mediation, composition, and invocation can be suitably addressed in the context of Semantic Web Services (SWS) which lie in the heart of Semantically Enabled Service-oriented Architectures (SESA) [5], thus providing for the next generation of service-oriented computing based on machine processable semantics.

Currently available SWS Execution Environments (SEE) such as Web Services Execution Environment[1] and Internet Reasoning Services[2] are implementing a significant part of the SESA facilities. However, they are showing high resilience to the internal and external changes. The components of a SEE (such as discovery, ranking and selection, choreography, and invocation) are not capable of coping with unexpected situations (e.g. failure of an external service invocation). Furthermore, the overall system functionality (implemented in a form of a process which orchestrates the components towards the user goal fulfilment) is usually predefined and fixed which hinders any possibility to react to

[1] `http://www.wsmx.org`
[2] `http://technologies.kmi.open.ac.uk/irs`

unusual situations. The complexity of the component behavior and their interrelations (e.g. behavior of an external service can influence its candidacy during the selection phase) require support for comprehensive observation and reactivity in order to enable an environment capable of a reliable and adaptable functionality. Additionally, the reactive behavior can assist in various service related activities, commonly experienced in a service provisioning system, such as enforcement of the Service Level Agreements (SLAs) and validation of functional service compositions.

2 State of the art

In the last few years, much industrial and academic research effort has been made in the area of SOA monitoring. This thesis is especially interested in event-based monitoring where, according to Jonas [7], the monitoring is performed as a set of functions like instrumentation and sensing, data collection and analysis. In the world of run-time monitoring of SOA systems, Web Service composition monitoring is a thoroughly addressed research topic where vast majority of the approaches are directed towards BPEL descriptions (Barbon et al. [1] and Baresi et al. [2]). However, the research community showed only limited interest in SWS monitoring. Vaculin et al. [11] defined an event taxonomy (which will be adapted and extended by the work) and a solution for detection of complex events in the context of the OWL-S choreographies. Nevertheless, the existing approaches are not considering any application of the service capability and behavior models (i.e. service choreography and orchestration) during the monitoring functions. Reliability and adaptability of services have been widely investigated in the Grid computing field where some of the approaches (like Xing et al. [12]) are arguing for an ontology-based information integration representing Grid resources. Oberle [9] presented an ontology-based approach to support development and administration of the middleware-based applications. Although currently absent, formal characterization of the SWS middleware solutions can contribute to the annotation and processing of the complex events produced by them. Schmidt et al. [10] introduced a survey on current trends in the field of event-driven reactivity. Additionally, some interest is shown in the integration of Event-Condition-Action (ECA) rules and Semantic Web. May et al. [8] proposed an ontology based on the ECA paradigm, and Behrends et al. [3] combined ECA rules and process algebras for the formal specification of the Action part. While the majority of the approaches focuses on the formal characterization of the event part, the work in this thesis also envisions precise mathematical grounding of the condition and action parts.

3 Proposed Approach

The presented Ph.D. thesis explores the problem of reactivity in the context of SEE in order to adapt its behavior when confronted with (un)expected events.

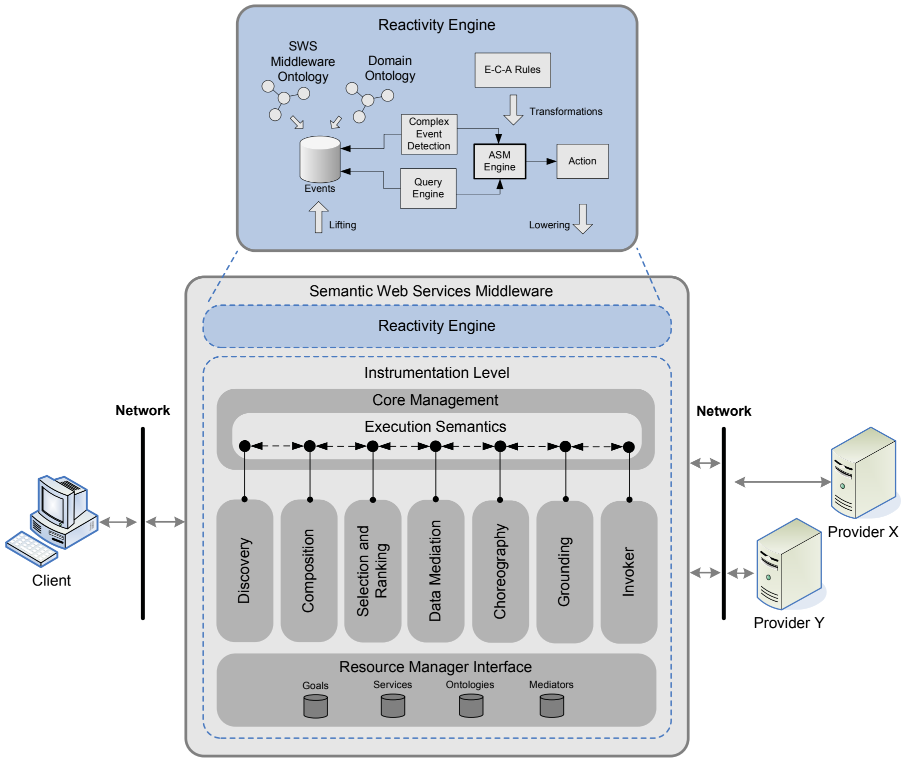

Fig. 1. Architecture of a reactive semantic execution environment

The goal of the thesis is to employ the precise mathematical notation (i.e. Abstract State Machines [4]) in order to develop an autonomic and reliable SEE solution. Figure 1 presents foundations of the work currently conducted:

Formal representation of the notions in SEE. Suitable SWS framework (e.g. Web Service Modeling Ontology - WSMO [6]) is used to formally characterize not only externally used business services but also SEE components and processes. Furthermore, all events of interest have an ontological representation. Event acquisition is performed by the proper instrumentation of the observed components during which suitable lifting mechanisms are used (i.e. transformation of collected low-level data to the semantically represented counterparts). These representations are forming rich knowledge base. Advantages of such a representation in comparison to the commonly used log files are numerous - richer descriptions of the events and their casual relationships, application of powerful reasoning techniques for data analysis and easier sharing of the the data through the standardized languages and tools.

Definition and enactment of the reactive behavior. The primary objective is to employ the well founded mathematical notation in the form of Abstract State Machines (ASMs) in order to model and implement reactive behavior of the system. While the state signature of the ASM describes events of interest as well as possible actions, the transition rules are defining system's reactive behavior by leveraging on service invocations capable of modifying component life cycle and affecting the system behavior. Since the ASM notation is cumbersome for majority of potential users, definition of reactive behavior is based on a set of ECA rules. Each of the respective parts of an ECA triple is specified by the application of the rule-based language (i.e. Web Service Modeling Language[3]). While the *Event* part represents a simple binding between a formal expression variables and observed events, the *Condition* part represents a query over the knowledge base of arbitrary complexity. Special attention is devoted to the *Action* part, which enables service-based management of the available components (thus modifying component life cycle and affecting component behavior). Transformations from ECA rules to ASM are necessary in order to provide a comfortable, natural and well established way to describe reactive behavior of the system. Integration of an event detection mechanism (*Event* part), a query engine (*Condition* part) and the service-based activity descriptions (*Action* part) with the ASM engine creates a novel platform for dealing with reactive behavior.

4 Methodology

The proposed approach is based on the specification, design and development of an extension to the existing SESA implementation. The prototype will cover all the peculiarities of the suggested solution presented in Section 3. It is expected that activities around the thesis work will produce various results which should be suitably evaluated. Ontological characterization of the environment should yield a set of ontologies that will be evaluated according to a selected ontology evaluation technique (e.g. application-based and data-driven methodology). Furthermore, the languages used to specify ECA rules will be compared to the existing ECA rule specifications in the context of expressivity and ease-of-use. The prototyped environment will be evaluated by comparison of its capabilities and behavior with respect to existing ECA-based reactivity solutions. The monitoring probe effect will also be evaluated by comparing critical performance measurements of the platform with and without instrumentation of the SWS middleware components. At the end, results of the thesis will be evaluated by empirical study based on the requirements stemming from the project which will use the proposed solution (the COIN project[4] which envisions service provisioning based on the Software as a Service paradigm).

5 Conclusions and future work

This paper outlined the objectives, proposed the approach and briefly described the methodology and evaluation of the Ph.D. thesis, which in general aims to

[3] `http://www.wsmo.org/wsml`

[4] `http://www.coin-ip.eu`

apply a formally grounded event-triggered reactivity approach based on Abstract State Machines in order to develop an autonomous and reliable SEE. Future efforts regarding the work will concentrate on the most challenging part of the research plan which is specification of the ECA rule structure and associated languages and transformation of the ECA rules to ASM notions. Furthermore, integration of a fast complex event processing engine is planned in order to rise the overall system performance by lifting the load of detecting events of interest from ASM engine.

References

1. F. Barbon, P. Traverso, M. Pistore, and M. Trainotti. Run-Time Monitoring of Instances and Classes of Web Service Compositions. In *ICWS '06: Proceedings of the IEEE International Conference on Web Services*, pages 63–71, Washington, DC, USA, 2006. IEEE Computer Society.
2. L. Baresi, C. Ghezzi, and S. Guinea. Smart monitors for composed services. In *ICSOC '04: Proceedings of the 2nd international conference on Service oriented computing*, pages 193–202, New York, NY, USA, 2004. ACM.
3. E. Behrends, O. Fritzen, W. May, and F. Schenk. Combining eca rules with process algebras for the semantic web. In *RULEML '06: Proceedings of the Second International Conference on Rules and Rule Markup Languages for the Semantic Web*, pages 29–38, Washington, DC, USA, 2006. IEEE Computer Society.
4. E. Borger and R. Stark, editors. *Abstract State Machines: A Method for High-Level System Design and Analysis*. Springer-Verlag, 2003.
5. D. Fensel, M. Kerrigan, and M. Zaremba, editors. *Implementing Semantic Web Services: The SESA Framework*. Springer-Verlag, 2008.
6. D. Fensel, H. Lausen, A. Pollers, J. de Bruijn, M. Stollberg, D. Roman, and J. Domingue, editors. *Enabling Semantic Web Services: The Web Service Modeling Ontology*. Springer-Verlag, 2007.
7. M. Jonas. *Resource-Predictable and Efficient Monitoring of Events*. PhD thesis, Linkping University, Department of Computer and Information Science, 2004.
8. W. May, J. J. Alferes, and R. Amador. *Rules and Rule Markup Languages for the Semantic Web*, chapter Active Rules in the Semantic Web: Dealing with Language Heterogeneity, pages 30–44. Springer Berlin Heidelberg, 2005.
9. D. Oberle, editor. *Semantic Management of Middleware*. Springer Science + Business and Media, 2005.
10. K.-U. Schmidt, D. Anicic, and R. Sthmer. Event-driven Reactivity: A Survey and Requirements Analysis. In *Proceedings of the 3rd international Workshop on Semantic Business Process Management at 5th European Semantic Web Conference*, 2008.
11. R. Vaculin and K. Sycara. Specifying and Monitoring Composite Events for Semantic Web Services. In *Proceedings of the 5th IEEE European Conference on Web Services*, November 2007.
12. W. Xing, O. Corcho, C. Goble, and M. D. Dikaiakos. An ActOn-based Semantic Information Service for EGEE. Technical Report TR-0111, 2007.