Reasoning Tasks and Mediation on Choreography and Orchestration in WSMO

Michael Stollberg

University of Innsbruck, Digital Enterprise Research Institute, Technikerstrasse 21a, 6020 Innsbruck, Austria
michael.stollberg@deri.org
http://www.deri.org

Abstract. The Web Service Modelling Ontology (WSMO) semantically describes the core elements of Semantic Web Services, aiming at a framework for unambiguous formal descriptions on which inference mechanisms shall enable automated discovery, composition, execution and invocation of Web Services. Choreography and Orchestration are defined as sub-classes of service interface for describing how the functionality of a Web Service can be consumed and how it is achieved by aggregating other Web Services. Therefore, a basic model for formally describing the dynamics of service interface descriptions has been defined. This paper discusses the usability of the mentioned model by identifying the main reasoning tasks for choreography and orchestration descriptions as well as the requirements on mediation facilities in order to handle possible occurring mismatches in WSMO service interfaces.

1 Introduction

Realizing the promise of Semantic Web Services requires usage of ontologies as the underlying data model throughout the complete service usage process. This ranges from ontology keywords referenced in non-functional properties, over capability and interface definitions of services that are used in the pre-execution phase, up to the interchange of ontology data during service execution. In consequence, WSMO defines ontologies as a top-level element for Semantic Web Services; all other WSMO elements apply ontologies for defining the data aspects.

The concept of service interfaces in WSMO is concerned with service usage for consuming the functionality of a service, as well as interaction with other services in order to achieve the service functionality. In relation to the capability as the functional description of a service, service interfaces are understood as decompositions of a service capability that describe how the service functionality can be consumed and how it is achieved. Although being very different from a conceptual point of view, the notions of choreography and orchestration are defined as sub-classes of service interfaces in WSMO; the reason is that WSMO defines a formal model that can serve as a common basis for describing the dynamics of Web Service usage and Web Service interactions, respectively.

The main question arising with respect to the usability of the WSMO approach and model for service interface descriptions is the identification of the essential reasoning tasks for enabling semantically driven techniques for using and handling Web Services automatically. Associated to this, the requirements for appropriate mediation facilities that allow resolution of possibly occurring mismatches within service interface definitions abound. This paper addresses these two aspects. Therefore, we recall the basic formal model for describing service interfaces in WSMO, and then address the reasoning tasks and requirements on mediation facilities for choreography and orchestration.

The paper is structured as follows: Section 2 resumes the basic model for service interfaces defined in WSMO; Section 3 addresses the support provided for choreography descriptions and identifies the essential reasoning tasks and requirements for appropriate mediation facilities; Section 4 addresses the same issues within orchestration; Section 5 discusses related work, and Section 6 concludes the paper.

2 Basic Model for Service Interface Definitions

The following summarizes the common basic model for service interface descriptions in WSMO, as outlined in [13]. The reason for defining a common model for service interfaces is that the following requirements are the same for all service interfaces:

- to provide representation means for the dynamics of the information interchange that takes place when a service is used and interacts with other services
- to support ontologies as the underlying data model along with an appropriate communication technology for information interchange
- to rely on a sound formal model that defines the semantics of service interface specifications in order to allow operations on them.

The formal model for WSMO service interface descriptions relies on Abstract State Machines (ASM for short). ASMs are a high-level, abstract technique for validating complex systems or programs and provide a highly expressive, flexible, and formally sound means for representing dynamics. The core principles of ASMs are that they are state-based, they represent a state by a formal algebra, and they model state changes by guarded transition rules that change the values of functions and relations defined by the signature of the algebra [3]. The reason for choosing ASMs as the underlying formalism for defining service interface definitions in WSMO is the generality and expressiveness provided on the one hand, and, on the other, they allow to overcome the "Frame Problem" [15]: this refers to that a state in a dynamic system is defined by the current information existing at a certain point in time; at a state change, the changes on all information items have to be defined. In contrast to other formalisms, guarded transitions in an ASM fire in parallel such that each condition of the transitions is checked over the current state. Thus, the ASM model overcomes the frame

problem as only information that is changed at a state transition needs to be defined within guarded transitions [7].

The ASM-based model for service interfaces defined in WSMO provides the formal basis for specifying ontology data interchange within service interfaces. In accordance to the ASM framework, this model consists of three notions that we formalize in the following definition:

- a **Vocabulary** Ω that defines the information space of a service interface on basis of ontologies. This is defined as the ontological schema of the information interchanged in a service interface by denoting the used concepts, relations, and functions of ontologies. The communicative usage of this ontological schema information is indicated by sub-information spaces for ontologies instances: Ω_{in} denotes the vocabulary of information received by the service interface; Ω_{out} the vocabulary of information that is provided by the service interface; Ω_{shared} denotes the vocabulary of information both received and provided by the service interface; Ω_{static} defines the vocabulary of ontology notions that cannot be changed by the service interface, and $\Omega_{controlled}$ denotes those that can only be changed by the service.
- **States** $\omega(\Omega)$ that denote a status of the information space within the dynamics of a service interface that is defined by the attribute values of the ontology instances of Ω. A state denotes a stable status within the dynamics of a service interface that is existent as long as attribute values of instances are not changed, thus includes all communicative activities that do not change information in $\omega(\Omega)$.
- **Guarded Transitions** T that specify the dynamics of a service interface. The general structure of T is: if *condition*(ω) then *update*, whereby the condition reflects changes in Ω, while the update part defines the changes on information performed in the transition to the subsequent state ω'. At a state change from ω to ω', all T are executed whose condition is satisfied.

In principle, we understand this model to define an evolving ontology on the information space that progresses during service usage. Thereby, Ω contains the concepts, relations, functions, and axioms as ontology schema on basis of a domain ontology. A state $\omega(\Omega)$ is stable status of the information space of a service interface, defined by the concrete attribute values of ontology instances; the communicative activities to be performed on instance data are denoted by the sub-information spaces of Ω. The set of guarded transitions T defines state changes with regard to the evolution of the information space throughout the usage of the service interface. This provides a formal model for ontology data interchange in service interfaces that satisfies the requirements stated above.

This model allows to describe the dynamics of service interfaces with respect to the ontology data changed by and interchanged during service consumption or service interaction. The differences in choreography and orchestration descriptions are represented by different Guarded Transitions: while the *IF*-part refers to a state in both choreography and orchestration, the *THEN*-part in choreography defines communicative activities to be performed, while the *THEN*-part in

an orchestration denotes communicative activities in terms of an operation *op*. The operation dictates the target Web Service $\mathcal{S}_x$ to be used and the required *updates* over the information space. As for choreography, the updates are defined in terms of the changes in the attribute values of some (or all) instance data.

3 Reasoning and Mediation on Choreography Definitions

The W3C Web Service Glossary defines choreography to be concerned with "the interactions of services with their users. Any user of a Web service, automated or otherwise, is a client of that service. These users may, in turn, be other Web Services, applications or human beings." [8].

With regard to this definition, the following explains how aspects related to choreography can be described within the formal model for WSMO service interface definitions, and defines service compatibility determination as the main reasoning task on choreography definitions along with the emerging needs for mediation on choreography definitions.

3.1 Requirements for Choreography Descriptions

With respect to the W3C Glossary definition we identify two aspects that are related to choreography:

1. the interface description of a Web Service that describes the interaction behavior of the Web Service for consuming its functionality. We denote this as the *choreography interface of a Web Service*, indicated as $ci(S)$.
2. the interaction protocol that describes, from a global perspective, the communicative interaction of several Web Services and clients via their respective choreography interfaces. With respect to the W3C definition as well as to terminology conventions within Web Services research (see [12]), we denote this as a *choreography*, indicated as $\mathcal{C}(S_1, ...S_n) = interaction(ci(S_1), ..., ci(S_n))$.

In compliance to [2], the aspects that need to be described within choreography interface are the *external visible behavior* defined as those aspects of the business process of a Web Service where interaction with the client is required, and the *communication protocol* expected by the service so that a client can consume its functionality, as well as the *semantics of the information* to be interchanged. A choreography needs to define the *business process of the interaction from a global perspective* along with the *communication protocol between choreography participants* and the *semantics of the information* to be interchanged between choreography participants.

We can use the model for WSMO service interfaces for describing both, choreography interfaces of individual Web Services as well as choreographies as interaction models. For the former, we define the vocabulary $\Omega_{ci(S)}$ as the ontological information space of the choreography interface of a Web Service, and the guarded transitions $T_{ci(S)}$ for defining the communication protocol expected for

client-service interaction along with the ontological information interchanged; the external visible behavior is implicitly defined by the states $\omega_0(\Omega_{ci(S)})$ - $\omega_n(\Omega_{ci(S)})$ that can be reached by respective $T_{ci(S)}$. For describing a choreography, we describe the interaction from a global perspective: thereby, $\Omega_{C(S_1,...,S_n)}$ defines the ontological information space used for interaction, the guarded transitions $T_{C(S_1,...,S_n)}$ define the communication protocol and semantics of the information interchanged, and the states of the choreography business process are defined implicitly as above.

Thus, the model for WSMO service interface definitions seems to be appropriate for formally describing aspects concerned with choreography. The WSMO model does not restrict this to a specific technology, but leaves the choice open to implementations. However, the aim of this paper is not to present mappings into executable technologies, but to identify the essential reasoning tasks on basis of formal choreography descriptions.

3.2 Service Compatibility as Reasoning Task

We understand the determination of existence of a valid choreography for the interaction of several Web Services as the main reasoning task on choreography descriptions. This means, given $ci(S_1), \ldots, ci(S_n)$ for Web Services $S_1, \ldots, S_n$ that shall interoperate, whereby each $ci(S)$ is considered as static (i.e. not changeable with respect to the Web Service functionality), we have to determine whether there exists a choreography $\mathcal{C}((S_1), \ldots, (S_n)) = interaction(ci(S_1), ..., ci(S_n))$ as a valid interaction protocol for the Web Services. On the basis of previous work on this issue by Martens [10], we refer to such a choreography to be existent if *service compatibility* holds for $ci(S_1), ..., ci(S_n)$, denoted by $\mathcal{SC}((S_1), \ldots, (S_n))$ in the following.

To determine this, we need to proof that there exists a $\mathcal{C}((S_1), \ldots, (S_n))$ for which the following holds: (1) the service interface descriptions need to use homogeneous ontologies, (2) the information to be interchanged (i.e. the content of communicative acts) needs to be compatible; we refer to this as *information compatibility*, denoted as $\mathcal{SC}_{info}(C)$, and (3) the communication protocol of the choreography has to be sound, meaning that it has at least one start state and can reach a termination state without any additional input; we refer to this as *communication compatibility*, denoted as $\mathcal{SC}_{comm}(C)$.

While the first aspects refers to determination of homogeneous information spaces relevant for all WSMO element descriptions, we can determine information compatibility on basis of vocabulary definitions in choreography interfaces. Considering this for two Web Services, information compatibility is given if all information required as input by S_1 is output of S_2 and vice versa. The following gives the formal definition for information compatibility along with a generalization this for several Web Services:

$$\mathcal{SC}_{info}((S_1), (S_2)) \leftarrow (S_1(\Omega_{in} \cup \Omega_{shared}) = \tag{1}$$

$$S_2(\Omega_{out} \cup \Omega_{shared})) \wedge (S_1(\Omega_{out} \cup \Omega_{shared}) = S_2(\Omega_{in} \cup \Omega_{shared})). \tag{2}$$

$$\mathcal{SC}_{info}(C) \leftarrow \forall S_x \in \mathcal{C}\ \exists S_y \in \mathcal{C}.(\mathcal{SC}_{info}((S_x),(S_y))). \quad (3)$$

For communication compatibility, we have to determine the existence of a choreography as a sound interaction model for the choreography interfaces of the Web Services that are ought to interact. Considering this for two Web Services, we have to proof three properties on $\mathcal{C}((S_1),\ldots,(S_n))$ with respect to the definition of soundness: the first one is that there is an initial state $\emptyset$ for $\mathcal{C}((S_1),(S_2))$ that has compatible guarded transitions in $ci(S_1), ci(S_2)$. The second one is compatibility of guarded transitions in each state $\omega_x(\mathcal{C}((S_1),(S_2)))$. This is given if for the *IF*-part of all guarded transitions in a state $\omega_x(ci(S_1))$ holds that there exists a guarded transition in the equivalent state $\omega_x(ci(S_2))$ whose *THEN*-part fulfils the condition, or vice versa. This means that if the *IF*-part of $\mathcal{T}(\omega_x(ci(S_1)))$ defines a condition on a concept $\mathcal{C} \in \Omega_{in\cup shared}(S_1)$, then the *THEN*-part of some $\mathcal{T}(\omega_x(ci(S_2)))$ needs to define a communicative action on a concept $\mathcal{C}' \in \Omega_{out\cup shared}(S_2)$, and vice versa. The third property is existence of a proper termination state for $\mathcal{C}((S_1),(S_2))$, such that for some state $\omega_x(\mathcal{C}((S_1),(S_2)))$ there does not exist any guarded transition in the equivalent states $\omega_x(ci(S_1))$, neither in $\omega_x(ci(S_2))$. Generalizing this for choreographies with multiple participating Web Services, we define communication compatibility as follows:

$$\mathcal{SC}_{comm}(\mathcal{C}) \leftarrow \forall \mathcal{S}_x \in \mathcal{C}\ \exists \mathcal{S}_y \in \mathcal{C}.(\emptyset(\mathcal{S}_x,\mathcal{S}_y) \wedge GT_{\omega_1-\omega_m}(\mathcal{S}_x,\mathcal{S}_y) \wedge \omega_t(\mathcal{S}_x,\mathcal{S}_y)). \quad (4)$$

$$\emptyset(\mathcal{S}_1,\mathcal{S}_2) \leftarrow \emptyset(S_1) = \emptyset(S_2) \wedge GT_{\emptyset}((S_1),(S_2)). \quad (5)$$

$$GT_{\omega}(\mathcal{S}_1,\mathcal{S}_2) \leftarrow a \in \mathcal{IF}(T_{\omega}(S_1)) \wedge b \in \mathcal{THEN}(T_{\omega}(S_2)) \wedge \mathcal{SC}_{info}(a,b). \quad (6)$$

$$\omega_t(\mathcal{S}_1,\mathcal{S}_2) \leftarrow \neg\exists T(\omega_t) \wedge \omega_t(\mathcal{S}_1) = \omega_t(\mathcal{S}_2). \quad (7)$$

On this basis, we can determine existence of a valid choreography for Web Services that shall interact if service compatibility holds on the choreography interfaces of the Web Services: $valid(\mathcal{C}((S_1),\ldots,(S_n))) \leftarrow \mathcal{SC}_{info}(ci(S_1),\ldots,ci(S_n)) \wedge \mathcal{SC}_{comm}(ci(S_1),\ldots,ci(S_n))$.

3.3 Mediation Requirements

The above examinations reveal the requirements for mediation on choreography descriptions: if service compatibility is not given for Web Services that shall interact, then a WW Mediator has to establish this between the given choreography interfaces. Therefore, we identify three aspects of mediation:

1. providing a homogeneous information space for a choreography by resolving terminological mismatches between the choreography interfaces of participating Web Services.
2. handle missing information with regard to information compatibility. This means if some information is missing in $\Omega_{in}, \Omega_{out}$, or Ω_{shared} in $(ci(S))$ of a Web Service participating in a choreography, this needs to be added by additional interaction with the owner of the Web Service

3. establish a communication compatibility if this is not given a priori. Therefore, protocol related aspects need to be handled (i.e. if $ci(S_x)$ wants to send A and B in state ω_n but $ci(S_y)$ expects A in ω_n and B in ω_{n+1}), as well as business process aspects (i.e. if common states between participating Web Services can only be reached by modifying the business process of the choreography).

The first aspect relates to data level mediation, wherefore a WW Mediator can use respective OO Mediators [14]. The third aspects correlates to the process mediation approach followed in WSMX [6].

4 Service Orchestration

The W3C Glossary defines an orchestration as "the sequence and conditions in which one Web service invokes other Web services in order to realize some useful function. That is, an orchestration is the pattern of interactions that a Web service agent must follow in order to achieve its goal." [8]. In accordance to this, an orchestration in WSMO is defined as a service interface that describes how a Web Services aggregates other Web Services into its functionality.

The following explains orchestration definitions on basis of the common model for WSMO service interfaces, and denotes the support for determining validity of service orchestrations as the main reasoning tasks along with emerging mediation needs.

4.1 Orchestration Descriptions

With respect to the afore examinations, an orchestration defines a decomposition of the capability of a Web Services and how these subtasks can be achieved by using other Web Services. Thereby, only those subtasks of a Web Service functionality are defined in the orchestration $\mathcal{O}(S)$ that are realized by other services; the service functionality can be realized either without using any other Web Service, only by aggregating other Web Services, or as hybrid realizations that combine the former realization types.

An orchestration $\mathcal{O}(S) = aggregation((S_1), ..., (S_n))$ defines the control and data flow for aggregating Web Services $(S_1), ..., (S_n)$ so that the functionality of S is achieved; thereby, S consumes the aggregated Web Services via their respective choreography interface $ci(S_1), ..., ci(S_n)$. We assume that a specific functionality required in a state $\omega_O(S)$ may only be a part of the functionality provided by an aggregated service. Hence, we introduce the concept of an *operation* for describing the interaction between services for consuming partial functionalities within an orchestration. An operation $op(service, update)$ denotes the interacting services and the communicative activities performed; the latter is expressed in terms of a choreography description. In consequence, an orchestration description consists of the vocabulary $\Omega_{O(S)}$ that denotes the information space of the orchestration from the perspective of the orchestrating Web Service S, guarded

transition $T_{O(S)}$ of the form IF *condition*($\omega_{O(S)}$) THEN $op(S_{target}, update)$, and the states $\omega_x(O(S))$ that can be reached by $T_{O(S)}$.

Let's consider an example for clarification: a Web Service 'VTA' orchestrates a flight booking service S_1 and a hotel booking service S_2: the first state in $\mathcal{O}_{VTA}$ is to retrieve information for suitable flights from S_1, the second one retrieval of available hotel rooms from S_2 with respect to the flight dates; in the third state, VTA books the flight and the hotel accordingly. Here, we consume S_1, S_2 completely, but not subsequently. This is what we want to and can be expressed in an orchestration definition on basis of the common model for WSMO service interface descriptions as outlined above. This does not support the dynamic *creation* of an orchestration, i.e. discovering the services to be aggregated and determine the business process of the orchestration, but it allows to *validate* orchestrations.

4.2 Validation of Orchestrations

Validity of an orchestration $\mathcal{O}(S) = aggregation((S_1), ..., (S_n))$ is given if the interactions with the aggregated services can be performed successfully. This means that the operations defined in $\mathcal{O}(S)$ need to be compliant with the choreography interfaces $ci(S_1), ..., ci(S_n)$ of the aggregated Web Services, so that the functionalities expected by the orchestration can be provided.

For determining validity of an orchestration, we assume the following: (1) the choreography interface of a Web Service is first static, i.e. it can not be changed, and (2) the functionality of a Web Service consumable via its choreography interface needs to be consumed completely, i.e. we can not just utilize a part of a service functionality. In consequence, we need to determine for all services aggregated in an orchestration whether this service is consumed *correctly* and *completely*. Thus, for each service aggregated in $\mathcal{O}(S)$, the set of all operations $op(S_{target}, update)$ with a specific service provides a valid client for the choreography interface $ci(S_{target})$. This can be determined by testing service compatibility as described above between $(op(S_{target}, update), ci(S_{target}))$.

Hence, we can apply the same approach for determining validity of an orchestration and for existence of a valid choreography between Web Services. In consequence, also the requirements for WW Mediators for resolving mismatches within an orchestration are the same that we have determined for WW Mediators in choreographies above.

5 Related Work

The main merit of the common model for describing service interfaces in WSMO is the extensive support for ontologies as the underlying data model. This is a basic requirement for enabling Semantic Web Services, which is not supported by any existing language related to Web Service choreography or orchestration description: WSDL [4] allows to describe consumption interfaces of Web Services along with operations for XML data interchange; WS-CDL [9] provides

a language for describing choreographies as global interaction protocols without alignment to choreography interface description languages; BPEL4WS [17] provides an orchestration language in alignment with WSDL, thus only supports XML data interchange. Although the process model of OWL-S follows the same intention of describing both the client-service interaction for consuming a Web Service and the aggregation of Web Services [16], the distinction between choreography and orchestration is not explicitly defined. Also, the process representation language seems not to de adequate with respect to sophisticated reasoning support and is not based on any formal model (although semantics have been provided using Petri-Nets [11]).

Regarding service compatibility as the essential reasoning tasks on service interface descriptions, an initial approach has been presented in [5]. Therein, a formal model on basis of process algebras is defined for WSCI, an extension of WSDL that allows defining the communication process of a Web Service interface. Following the same approach, [10] defines a formal approach for service compatibility on basis of Petri Nets, that we apply and extend with respect to the WSMO description model for service interfaces.

Approaches for dynamic composition as presented in [18], [1], we regard these as compatible efforts to enable the ultimate goal of Semantic Web Services: given a goal to be satisfied, existing Web Services are dynamically composed in a way that they can solve the goal as a higher-level functionality. Nevertheless, the need for the essential reasoning tasks on choreography and orchestration descriptions are also relevant for such composition techniques in order to ensure consistency and interoperability of service compositions.

6 Conclusions

In this paper we have discussed the usability of the basic model for describing service interfaces in WSMO, and determined the essential reasoning tasks for choreography and orchestration.

The WSMO approach for formal service interface descriptions can be understood as an ontology that evolves with respect to the information interchange in service consumption or interaction. Such exhaustive support for ontologies as the underlying data model for service usage is a main drawback of existing description languages for choreography and orchestration. As a specialization of existing terminology, we have shown that choreography is concerned with the communication in interactions, while orchestration extends choreography descriptions with notions for control and data flow. The main reasoning task on both types of service interfaces is service compatibility what allows determining a priori whether Web Services are interoperable; the realization of this is supported by the WSMO model for service interface descriptions.

Concluding, we remark that we consider a choreography interface description as mandatory for Web Services in order to be consumable, while an orchestration is optional in the sense that not every Web Services necessarily needs to define an orchestration in order to achieve its functionality.

References

1. Albert, P.; Henocque, L.; Kleiner, M.: *A Constrained Object Model for Configuration Based Workflow Composition.* Submitted to the 1st International Workshop on Web Service Choreography and Orchestration for Business Process Management, to be held at the BPM 2005, Nancy, France, September 2005.
2. Austin, D.; Barbir, A.; Peters, E.; Ross-Talbot, S.: *Web Services Choreography Requirements.* W3C Working Draft 11 March 2004.
3. Boerger, E. and Staerk, R.F: *Abstract State Machines. A Method for High-Level System Design and Analysis* . Berlin, Heidelberg: Springer 2003.
4. Booth, D.; Liu C. K. (Eds): Web Services Description Language (WSDL) Version 2.0 Part 0: Primer. W3C Working Draft 21 December 2004.
5. Brogi, A., Canal, C., Pimentel, E., Vallecillo, A.: Formalizing Web Service Choreographies. In Proceedings of First International Workshop on Web Services and Formal Methods, Pisa, Italy, February 2004.
6. Cimpian, E. and Mocan, A.: WSMX Process Mediation Based on Choreographies. In the 1st International Workshop on Web Service Choreography and Orchestration for Business Process Management at 3rd International Conference on Business Process Management (BPM 2005), Nancy, France, September 2005.
7. Eck, P.A.T. van, Engelfriet, J., Fensel, D., Harmelen, F. van, Venema, Y. and Willems, M: *A Survey of Languages for Specifying Dynamics: A Knowledge Engineering Perspective.* In IEEE Transactions of Knowledge and Data Engineering, 13(3) 2001; pp. 462-496.
8. Haas, H., Brown, A.: *Web Services Glossary.* W3C Working Group Note 11 February 2004.
9. Kavantzas, N.; Burdett, D.; Ritzinger, G.; Fletcher, T.; Lafon, Y.: *Web Services Choreography Description Language Version 1.0.* W3C Working Draft 17 December 2004.
10. Martens, A.: *On Compatibility of Web Services.* Petri Net Newletter (65), 2003; 12-20.
11. Narayanan, S., McIlraith A., S.: *Simulation, Verification and Automated Composition of Web Services.* In proceedings of the Eleventh International World Wide Web Conference (WWW-11), May, 2002.
12. Preist, C.: *A Conceptual Architecture for Semantic Web Services.* In Proceedings of the 3rd International Semantic Web Conference (ISWC 2004), 2004, pp. 395 - 409.
13. Roman, D., Scicluna, J., Feier, C.: (eds.): Ontology-based Choreography and Orchestration of WSMO Services. WSMO Working Draft D14, 1 March 2005.
14. Scharffe, F. (Ed.): *WSMO Mediators.* WSMO Working Draft D29, 11 March 2005.
15. Shanahan, M.: *The Frame Problem.* In Stanford Encyclopedia of Philosophy, 2004; available at: http://plato.stanford.edu/entries/frame-problem/.
16. The OWL-S Coalition: *OWL-S 1.1 Release*, November 2004; www.daml.org/services/owl-s/1.1/.
17. Thatte, S. (ed.): *Business Process Execution Language for Web Services Version 1.1*, Specification 05 May 2003.
18. Traverso, P., Pistore M.: *Automated Composition of Semantic Web Services into Executable Processes.* In Proceedings of the 3rd International Semantic Web Conference (ISWC 2004), 2004.