.

A Survey on Services Composition Languages and Models

Antonio Bucchiarone [1]

Istituto di Scienza e Tecnologie dell'Informazione
"A. Faedo" (ISTI - CNR)
Area della Ricerca CNR di Pisa, 56100 Pisa, Italy
antonio.bucchiarone@isti.cnr.it

Stefania Gnesi

Istituto di Scienza e Tecnologie dell'Informazione
"A. Faedo" (ISTI - CNR)
Area della Ricerca CNR di Pisa, 56100 Pisa, Italy
stefania.gnesi@isti.cnr.it

Abstract

Composition of services has received much interest to support business-to-business (B2B) or enterprise applications integration. On one side, the business world has developed a number of XML-based standards to formalize the specification of Web services, their composition and execution. On the other side, the Semantic Web community focuses on reasoning about web resources by explicitly declaring their preconditions and effects with terms precisely defined in ontologies. So far, both approaches have been developed rather independently from each other. In this paper the major languages, namely BPEL4WS, BPML, WSCI, WS-CDL and DAML-S, are compared with reference to the requirements identified and finally the trend of Services composition is discussed.

Key words: Service-Oriented computing, Business-to-Business, Web Services, Web Services Composition, Semantic Web.

1 Introduction

Service Oriented Computing (SOC) is an emerging paradigm for distributed computing and e-business processing that finds its origins in object-oriented and component computing. Services are computational entities that are autonomous and heterogeneous (e.g. running on different platforms or owned

[1] IMT Graduate School, Via San Micheletto, 3 - 55100 Lucca , Italy

by different organizations). Services are described using appropriate service description languages, published and discovered according to predefined protocols, and combined using an engine that coordinate the interactions among collaborating services. Web services technology is a widespread accepted instantiation of SOC which should facilitate integration of newly built and legacy applications both within and across organizational boundaries avoiding difficulties due to different platform, heterogeneous programming languages, etc.. Exploiting this kind of ubiquitous network fabric would result in an increasing of productivity and in a reduction of costs in *B2B* processes [10]. The business drive to increase the enterprise's agility in responding to customer and market needs has accelerated the integrations of various applications both within and across enterprise boundaries. Information flow and business processes must be streamlined and automated to increase overall business efficiency. Network connectivity with customers, suppliers and partners for quick and automated processing provides a vital and competitive edge for any enterprise. The idea behind this approach is to allow independently developed applications to be exposed as services and interconnected exploiting the already set up Web infrastructure with relative standards (HTTP [6], XML [23], SOAP [3] and WSDL [2]). *Web Services*, which are based on XML-based open standards, promise the interoperability of various applications running on heterogeneous platforms. They enable dynamic connections and automation of business processes within and across enterprises for enterprise application integration and business-to-business integration. Building on the ubiquitous and lightweight standards that are supported by mayor software vendors, Web services enable application integration via the publishing of application's functionality as services, as well as location and invocation of services over the Internet. Although Web services are sufficient for some simple interaction needs, they are not sufficient for integration of business processes that involve multiple enterprises. Business process integration in real business scenarios involve long-running interactions, transactions management, stateful invocations and are often driven by a workflow engine that execute a specified business process model to automate the information flow and the business operations. This raises the needs for *Web services composition* that provides the mechanism to fulfill the complexity of business processes execution. Different organizations are presently working on composition proposal. The most important in the past have been IBM's WSFL [8] and Microsoft's XLANG [7]. These two have then converged in Web Services Business Process Execution Language (BPEL4WS [22]) which is presently a working draft by OASIS. Another recent proposal in phase of standardization by the World Wide Web Consortium (W3C) is WS-CDL [21]. Both allow the definition of workflow-based composition of services with some similarities and some differences. These languages are designed to provide interoperability between various applications. The platform and language independent interfaces of the web services allow the integration of heterogeneous systems but, these web services standards do not

deal with the dynamic composition of existing services. A more challenging problem is to compose services dynamically. In particular, when a functionality that cannot be realized by the existing services is required, the existing services can be combined together to fulfill the request. Currently the Web service technologies fall on the restricted capability to support static service composition. Their limit comes out from the total absence of semantic representation of the services available on the Internet. In response to these limitations, a number of solutions have been proposed by the Semantic Web community (for example DAML-S [17]).

The **Semantic Web** [13] is an extension of the current web in which information is given well-defined meaning, better enabling computers and people to work in cooperation. This is realized by marking up Web content, its properties, and its relations, in a reasonably expressive markup language with a well-defined semantics.

In this paper we present a survey of several Web Service Composition Languages. The objective of this paper is twofold. The first goal is to focus on the ability of each languages to: i) model the business collaboration, ii) model the execution control of processes, iii) represent the roles of participants, iv) manage transactions and compensations over services invocation, v) take into account exceptions handling, vi) have a semantic support, vii) support of Business agreement and viii) have a Software vendor support. The second is to take this analysis as the starting point in order to define the requirements of a new automatic framework for the design, analysis and development of composition in Service Oriented Computing. The structure of the paper is introduced in the next session.

1.1 Outline of the paper

The paper is structured as follows: after the above introduction, section 2 presents on overview of the various models for the Service-Oriented Composition. Section 3 introduce the current service composition languages from the different point of view of Web and Semantic Web services . It lists an overview of BPEL4WS, BPML, WSCI, WS-CDL and DAML-S languages while section 4 is devoted to a comparison on them respect to some requirements. Finally section 5 reports some conclusive considerations.

2 Service-Oriented Composition: Current Models

Composition of Services has received much interest to support business-to-business or enterprise application integration. The business world has developed a number of XML-based standards to formalize the specification of Web Services, their composition and execution. The Semantic Web community focuses instead on reasoning about web resources by explicitly declaring their preconditions and effects by means of terms precisely defined in ontologies.

The objective of this section is to introduce the existing composition models in the two different fields. We start with the definitions of Web Services and Semantic Web Services and then we will conclude with the introduction of various service composition models.

2.1 Web Services and Semantic Web Services

Web Services (WSs) are defined as self-contained, modular units of application logic which provide business functionality to other applications via an Internet connection. Web Services support the interaction of business partners and their processes by providing a stateless model of "atomic" synchronous or asynchronous message exchanges. Moreover WSs are characterized by two specifications: WSDL [2] and SOAP [3]. The former defines the interface that a Web service exhibits in order to be invoked by other services. The Web Services Definition Language (WSDL) is an XML-based language, which specifies a Web service by defining messages that provide an abstract definition of the data being transmitted and operations that a Web service provides to transmit the messages. Four types of communication are defined involving a service's operation (endpoint): the endpoint receives a message (one-way), sends a message (notification), the endpoint receives a message and sends a correlated message (request-response), and it sends a message and receives a correlated message (solicit-response). Operations are grouped into port types, which describe abstract end points of a Web Service such as a logical address under which an operation can be invoked. In Figure 1 we show a sample WSDL fragment of One-Way and Request-Response actions.

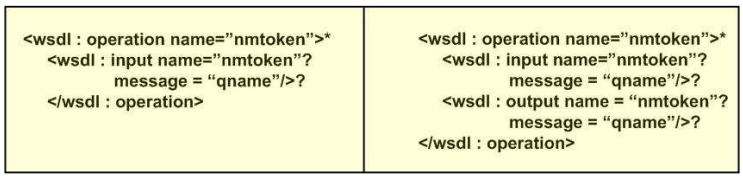

Fig. 1. WSDL Operations: One-Way and Request-Response

WSDL provides a function-centric description of Web services covering inputs, outputs and exception handling.
The Semantic Web [13] provides a process level description of the service which, in addition to functional information, models the preconditions and postconditions of the process so that the evolution of the domain can be logically inferred. It relies on ontologies to formalize domain concepts which are shared among services. The Semantic Web efforts [13,15], especially with respect to the recent trend towards **Semantic Web Services** [12], aim at fully automating all the stages of the Web services lifecycle. The Semantic Web considers the World Wide Web as a globally linked database where web pages are marked with semantic annotations. Semantic annotations are assertions about web resources and their properties expressed in the Resource Description Format (RDF) [18]. Along with RDF, one can use RDF Schema

(RDFS) to express classes, properties, ranges and documentation for resources and DAML-S [17] ontology to represent further relationships and/or properties like equivalences, lists, and data types.
With the Semantic Web infrastructure in place, practical and powerful applications can be written that use annotations and suitable inference engines to automatically discover, execute, compose, and interoperate Web services.

Given the different information that is available to specify a Web service in both the approaches we can subdivided the web services composition models in two classes: *Static* and *Dynamic* Services Composition (see figure 2).

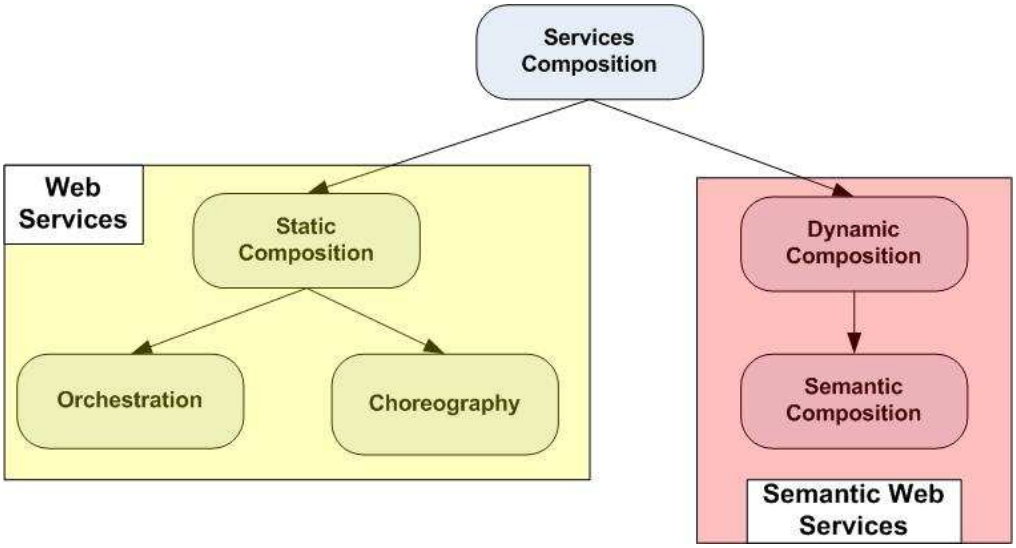

Fig. 2. Services Composition Models

2.2 Static Services Composition

A relevant feature for Web services is the mechanism for their reuse when complex tasks are carried out. It is often the case, to define new processes out of finer-grained subtasks that are likely available as Web services. To this aim, extensions of the Web service technology are considered which support the definition of complex services out of simpler ones. Composition rules deal with how different services are composed into a coherent global service. In particular, they specify the order in which services are invoked, and the conditions under which a certain service may or may not be invoked. Two possible approaches are currently investigated for the static service composition. The first approach, referred to as Web services ***orchestration***, combines available services adding a central coordinator (the orchestrator) which is responsible for invoking and combining the single sub-activities. The second approach, referred to as Web services ***choreography***, does not assume the exploitation of a central coordinator but it defines complex tasks via the definition of the conversation that should be undertaken by each participant. Following this approach, the overall activity is achieved as the composition of peer-to-peer

interactions among the collaborating services. Several proposal already exist for orchestration languages (see e.g. BPML [19], BPEL4WS [22] etc..). On the contrary, choreography languages are still at a preliminary stage of definition. A first proposal, named WS-CDL [21], has been issued from W3C in December 2004.

2.3 Dynamic Services Composition

Web Services are designed to provide interoperability between different applications. The platform and language independent interfaces of the web services allow the easy integration of heterogeneous systems. Web languages such as Universal Description, Discovery, and Integration (UDDI) [5], Web Services Description Language (WSDL) [2] and Simple Object Access Protocol (SOAP) [3] define standards for service discovery, description and messaging protocols. However, these web service standards do not deal with the dynamic composition of existing services. The new industry initiatives to address this issue such as Business Process Execution Language for Web Services (BPEL4WS) [22] focus on representing composition where flow of the informations and the binding between services are known a priori. A more challenging problem it to compose services dynamically. In particular, when a functionality that cannot be realized by the existing services is required, the existing services can combined together to fulfill the request. The dynamic composition of services requires the location of services based on their capabilities and the recognition of those services that can be matched together to create a composition as described in [11]. The full automation of this process is still the object of ongoing research activity, but accomplishing this goal with a human controller as the decision mechanism can be achieved. The main problem for this goal is the gap between the concepts people use and the data computers interpret. This barrier can be overcome using semantic web technologies.

3 Overview of current service composition languages: from Web to Semantic Web

Web services composition that is guided by standards is an important element of the service-oriented computing. It enables broader interoperable business processes and provides reusability benefits. Various standards have been developed by major software vendors like IBM, Microsoft and Sun Microsystems. The process of refinement and convergence of standards are still going on. An overview of major Web and Semantic Web services composition language, namely BPEL4WS, BPML, WSCI, WS-CDL and DAML-S, is presented below.

BPEL4WS [22] is a XML-based specification language for specifying business processes that are exclusively based on Web services. It defines how multiple services can interact, giving the coordination rules of imported or ex-

ported functionality via Web services interface, necessary to achieve the business goals. It supports the definition of both executable business processes and abstract business processes by providing mechanisms to specify common core concepts of both types of processes with essential extensions for each process. An executable business process is defined as the model of the internal, actual behavior of a participant in a business interaction, while abstract business process is defined as mutually visible message behavior of each party involved in business collaboration. BPEL4WS employs the concepts of partner links to directly model peer-to-peer collaborations between business processes and partners services. An instance of a business process can only be created when messages or events are received from partner links. Partner links define the static and abstract relations with other partners based on Web services *portType* used in interactions. BPEL4WS uses the notation of endpoint reference to dynamically select a provider for a particular type of service and to invoke the operation. The notation of Property is used to represent data elements within a message exchanged. A correlation set declares the correlation tokens that are used by BPEL4WS compliant infrastructures to build instance routing. BPEL4WS binds Web services into cohesive units encapsulated in activities.

BPML [19] is another standard proposed by Business Process Management Initiative (BPML.org). BPML was originally developed to enable the standard-based management of e-business processes used with Business Process Management System (BPMS) technology. It can be applied to a variety of scenarios, including enterprise application integration and Web services composition. BPML is a specification language dedicated to executable business processes. It is not surprising to see that BPML includes comprehensive support on control flow, data flow and event flow with structures of sending, receiving and invoking services, and control structures of conditional choice, sequential, iteration and parallel execution with synchronization. The definition of a Process is the template that instantiate process instances. It is the building block for modeling the execution of business processes. The concept of Signal is used to synchronize parallel activities with explicitly raising and waiting for the signal from activities. Exceptions are handled through the exception process and fault handler. Compensation is also supported to revert the effect of activities in processes. Transactions of business operation are supported with three means of declaring the activities as Atomic activity, Persistent process and Open nested transaction. The notation of Schedule is used to generate timing events that can trigger the instantiating of processes. BPML builds on top of WSDL for service description.

WSCI [20] was the first XML-based language that aims to provide a standard for specifying the overall collaboration between Web services providers and services users by describing messages exchanges that happen among the

involving parties. WSCI only specified abstract business process that is observable between modeled Web services. WSCI was built on top of WSDL, which defines the Web services operation involved in WSCI activities. Actions are used to define basic requests and response messages. External services are invoked through Call. WSCI also supports the transactions and exceptions handling. WSCI addresses Web services-based choreography in two primary levels. At the first level, WSCI builds up on the WSDL portType capabilities to describe the flow of messages exchanged by a Web service in the context of a process. This, however, is still a one-sided description of the message exchange, purely from the view point of the Web services own interface (portType). The construct introduced by WSCI permits the description of the externally "observable behavior" of a Web service, facilitating the expression of sequential and logical dependencies of exchanges at different operations in WSDL portType.

WS-CDL [21] is an XML specification targeted for composing interoperable, long running, peer-to-peer collaborations between Web services participants. It describes the global view of the observable behavior of messages exchange of all Web services participants that are involved in the business collaboration. WS-CDL is purely for abstract business processes specification, independent from the platforms and programming languages that are used to implement the Web services participation. WS-CDL models the peer-to-peer collaboration between participants with different roles using Choreography. Choreography makes use of Interaction and Activities notation to define the Relationships, which represents message exchanges between two Web services participants described in WSDL. Choreography supports exceptions and compensations with *Exception Block* and *Finalizer Block*. Messages that are exchanged between participants are modeled with Variables and Tokens, whose type can be specified in XML schema or WSDL. Channel is used to specify how and where messages exchange can take place. Activity performs the actual work of interaction through Interaction, which results in an exchange of messages. Perform invokes another Choreography, and Assign assigns the values to variables. Control structures that model the combined activities are simple, including Sequence, Parallel and Choice. Synchronization among activities can be achieved via WorkUnit, which defines the guard condition that must be fulfilled in order to continue Activity.

The **DAML-S**-services language (DAML-S) [17] is a set of language features arranged in ontologies to establish a framework within which the web services may be described in the semantic web context. DAML-S partitions a semantic description of a web service into three components: the service profile, process model and grounding. The *service profile* describes what the service does by specifying the input and output types, preconditions and effects. The *Process Model* describes how the service works; each service is

either an AtomicProcess that is executed directly or a CompositeProcess that is a combination of other subprocesses. The *Grounding* contains the details of how an agent can access a service by specifying a communications protocol, parameters to be used in the protocol and the serialization techniques to be employed for the communication. The similarities between DAML-S and other technologies may be expressed as follows: The profile description has a similar functionality of the yellow pages in UDDI, the process model is similar to the business process model in BPEL4WS and grounding is just a mapping from DAML-S to WSDL. The main contribution of DAML-S is the ability to express the entities using the concepts defined in Semantic Web ontologies which provide expressive constructs that are suitable for the automatic discovery and composition of services. DAML-S service descriptions are made to link to other ontologies that describe particular service types and their features. The profile description of DAML-S services has a hierarchy as well. A profile hierarchy ontology describes this relationship and this information can be used for filtering the services that can be composed together.

4 A Comparison

We compare BPEL4WS, BPML, WSCI, WS-CDL and DAML-S according to the following requirements:

- *Modeling the collaboration*: The ability to perform long-lived, peer-to-peer collaboration between participating services. Collaboration must be modeled in terms of interactions of messaging exchanges.
- *Modeling the execution control*: The ability to assemble and incorporate individual Web services into the course of business process execution is vital for any Web services composition language.
- *Representation of Role*: Parties involved in business processes play different roles in different process stages. Representation of roles is necessary fo reflection of responsibility and behavior that parties are assumed in various scenarios.
- *Transactions and Compensations*: Business processes usually are long-running processes that may take hours or weeks to complete, and therefore the ability to manage transactions and compensations over services invocation is critical for Web services composition. Compensations are needed to rollback the effects of completed transactions when there is a failure in the enclosed transaction scope.
- *Exceptions handling*: The composition of Web services makes use of external Web services that are purely under the control of the Web services owner. It must take into account exceptions handling in the process of invocation when external Web services do not respond.
- *Semantic support*: Web services composition languages should enable the representation of semantics of composed services to facilitate the automated

composition of Web services. The semantics descriptions that enable dynamic service discovery and invocation are imperative.

- *Business agreement support*: It is important in a business-to-business scenario to define agreement between involved parties. Business agreement defines the contract between two or more parties on Quality of Services (QoS). It is necessary to represent required QoS in composed Web services.
- *Software vendor support*: if the language has a Software support.

The comparison is listed in the table in Figure 3

	BPEL4WS	BPML	WS-CDL	WSCI	DAML-S
Modeling the collaboration	Strong Support	Indirect support	Strong support	Strong support	Strong support
Modeling the execution control	Strong Support	Strong Support	No support	No support	Strong support
Representation of the Role	Weak support	No support	Strong support	Strong Support	No support
Transaction and Compensation	Indirect support	Strong support	Indirect support	Strong support	Indirect support
Exception handling	Strong support	Strong support	Support	Strong support	Strong support
Semantic support	No support	No support	No support	No support	Strong support
Business agreement support	No support	No support	No support	No support	No support
Software vendor support	Many	Few	No	Few	Few

Fig. 3. The comparison of BPEL4WS, BPML, WSCI, WS-CDL and DAML-S

In the comparison table the "Indirect" value means that the requirements are not directly supported by the language. For example in BPEL4WS the transactions are realized through faults handler and compensations. In summary all these languages provide the mechanisms to model the collaborations, only BPML needs of an indirect support. BPML facilitates the modeling of execution of processes, while BPEL4WS and DAML-S attempt to cover both aspects. Finally WS-CDL is more natural to model B2B collaborations. All these languages support the imperative part of service composition, namely exceptions handling and compensations and all possess the capability to compose more complex structures and activities. The support for semantics representation and description is present only in DAML-S. Business collaborations require the established and enforcement of business agreement on QoS, but all these languages do not address this. As for tools support BPEL4WS has gained the widest support from industry. Most major software vendors has pledged BPEL4WS support in their products. Instead there is no too; support to WS-CDL, and it is not clear the industry acceptance of WS-CDL.

5 Conclusions and Future Work

There are other initiatives that work on the comparative study of composition languages [26,27,28]. Analysis of composition languages based on workflow pattern has also been done [29]. These comparison however are conducted almost at the micro level focusing on specific language like structure and control patterns. In this paper, an overview of major Web and Semantic Web services composition languages was presented. Five languages, namely BPEL4WS, BPML, WSCI, WS-CDL and DAML-S, were chosen and compared against eight requirements that a service composition language should support to facilitate the composing of business processes based on Web and Semantic Web services. DAML-S, like other service composition languages, provides a means to create description of Web services that can be interpreted programmatically. The distinguishing characteristic of DAML-S is that, while current Web services specification standards focus on service syntax, its goal is to facilitate the description of the semantics of services, their interfaces, and their behavior. The problem of the composition of web services is addressed by two orthogonal efforts. From the one side, most of the major industry players propose low level process modelling and execution languages, like BPEL4WS [22]. These languages allow programmers to implement complex web services as distributed processes and to compose them in a general way. However, the definition of new processes that interact with existing one must be done manually, and this is a hard, time consuming, and error prone task. From the other side, research within the Semantic Web Community proposes a top-down unambiguous description of services capabilities, e.g. in DAML-S [17], thus enabling the possibility to reason about web services, and to automate web services tasks, like discovery and composition. This paper is an initial results and much more details must be describe. In future work we wish to extend the comparison with more details (e.g., more factual information as the tool support, messaging models supported etc.) in order to understand which language (BPEL4WS, BPML, WSCI, WS-CDL and DAML-S) or model (Static or Dynamic) for the composition is better than an other. An other key point that we would to deepen is the the QoS characteristics that each language is able to describe in order to define a QoS Service Composition. All this with the main objective to understand which are the elements of a framework for the automatic QoS composition of services.

References

[1] M. Aiello, M. P. Papazoglou, J. Yang, M. Carman, M. Pistore, L. Serafini and P. Traverso. *A request language for Web-service based on planning and constraint satisfaction.* In proceedings of the International VLDB Workshop on Technologies for E-Services, pages 76-85, Hong Kong, China, 2002.

[2] E. Christensen, F. Curbera, G. Meredith and S. Weerawarana. *Web Services*

Description Language (WSDL) 1.1, 2001.
http://www.w3.org/TR/2001/NOTE-wsdl-20010315.

[3] W3C. SOAP 1.2 Working draft, 2001.
http://www.w3.org/TR/2001/WD-soap12-part0-20011217.

[4] G. Alonso, F. Casati, H. Kuno and V. Machiraju. *Web Services: Concepts, Architectures and Applications.* Springer-Verlag Berlin Heidelberg 2004.

[5] UDDI Consortium. *UDDI Executive White Paper*, Nov. 2001.
http://uddi.org/pubs/UDDI_Executive_White_Paper.pdf.

[6] W3C. *HTTP: HyperText Transfer Protocol Specification.*
www.w3.org/protocols.

[7] Microsoft Corporation. *Microsoft BizTalk Server 2002 Enterprise Edition*, 2002.
http://www.microsoft.com.

[8] F. Leymann. *Web Services Flow Language (WSFL) Version 1.0.* Technical Report, International Business Machines Corporation (IBM), May 2001.

[9] D. Burdett and N. Kavantzas. *WS choreography model overview.* Technical report, W3C (2004).

[10] N. Kavantzas. *Aggregating web services: Choreography and ws-cdl.*
http://www.lists.w3.org/.

[11] Z. M. Mao, E. A. Brewer and R. H. Katz. *Fault-tolerant, Scalable, Wide-Area Internet Service Composition.* U.C. Berkeley TR UCB//CSD-01-1129, Jan 2001.

[12] S. McIlraith, T. C. Son and H. Zeng. *Semantic Web Services.* IEEE Intelligent Systems, 16(2), Mar. 2002.

[13] T. Berners-Lee, J. Hendler and O. Lassila. *The Semantic Web.* Scientific American, 284(5):34-43, May 2001.

[14] M. Dean, D. Connolly, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider and L. A. Stein. *Web Ontology Language (OWL) Reference Version 1.0.* W3C Working Draft 12 November 2002.
http://www.w3.org/TR/2002/WD-owl-ref-20021112/.

[15] World Wide Web Consortium (W3C). *Semantic Web.*
http://www.w3.org/2001/sw/.

[16] I. Horrocks et al. *DAML+OIL*, 2001.
http://www.daml.org/2001/03/daml+oil-index.html.

[17] DAML Services Coalition. *DAML-S: Web Service Description for the Semantic Web.* In The First International Semantic Web Conference (ISWC), June 2002.

[18] D. Brickley and R. Guha. *Resource Description Framework (RDF) Model and Syntax Specification.* W3C Recommendation submitted 22 February 1999.
http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/.

[19] BPML.org. *Business process Modeling Language (BPML).* http://www. bpmi.org, 2002.

[20] A. Arkin et al. *Web Service Choreography Interface 1.0*, 2002. http://www.w3.org/TR/wsci.

[21] N. Kavantzas. *Web Services Choreography Description Language Version 1.0.* W3C, April 2004.

[22] T. Andrews et al. *Business Process Execution Language for Web Services (BPEL4WS)*, Version 1.1. http://ifr.sap.com/bpel4ws/index.html.

[23] Hewlett-Packard Company. *HP OpenView.* http://www.openview.hp.com.

[24] International Business Machine Corporation (IBM). *MQ Series Workflow for Business Integration*, 1999. http://www.ibm.com.

[25] W. Baush, C. Pautasso and G. Alonso. *Programming for dependability in a service-based grid.* In Proceedings of the International Symposium on Cluster Computing and the Grid (CCGrid), Tokyo, Japan, May 2003.

[26] W. M. P. van der Aalst. *Don't go with the flow: Web Services composition standard exposed.* IEEE Intelligent System. January/February 2003.

[27] C. Peltz. *Web services orchestration, a review of emerging technologies, tools and standards.* January 2003. http://devresource.hp.com/drc/technical_white_papers/WSOrch/WSOrcherstration.pdf.

[28] J. Mednling and M. Muller. *A Comparison of BPML and BPEL4WS*, 2003. http://wi.wu-wien.ac.at/mendling/publication/03-BXML.pdf.

[29] P. Wohed, W. M. P. van der Aalst et al. *Pattern Based Analysis of BPEL4WS.* Technical Report FIT-TR-2002-04,QUT, April 2002.