Volume 2, Issue 3, March 2012 ISSN: 2277 128X
International Journal of Advanced Research in Computer Science and Software Engineering
Research Paper
Available online at: www.ijarcsse.com

Basic Approaches to Semantic Web Services : A Comparative Study

Dr. Rachna Soni *
Asst. Prof & HoD (CS&A),
DAVC, Yamuna Nagar(Haryana)

Gurmeet Kaur
M.Sc. (CS),MCA,M.Phil,P.hd(P),
MBU, Shimla
grmtkaur02@gmail.com

***Abstract*— It has been recognized that due to the autonomy and heterogeneity of Web services and the Web itself, new approaches should be developed to describe and advertise Web services. The most notable approaches rely on the description of Web services using semantics. This new breed of Web services, termed semantic Web services (SWSs) aim to improve the possibilities for automated discovery, composition and invocation of Web Services by providing ontology-based service descriptions expressed in a formal language. Several approaches have been driving the development of Semantic Web Service frameworks such as OWL-S (Ontology Web Language for Services), WSMO (Web Service Modeling Ontology) and IRS-III (Internet Reasoning Service). This paper focuses on framework of different approaches of Semantic Web Services and also contrast these approaches to SWS according to the proposed dimension.**

***Keywords*— SWS, IRS-III, OWL-S, WSMF, WSMO**

I. INTRODUCTION

Current industry standards for describing Web Services focus on ensuring interoperability across diverse platforms, but do not provide a good foundation for automating the use of Web Services. A key problem with the use of standards for Web Service description (e.g. WSDL) and publishing (e.g. UDDI) is that the syntactic definitions used in these descriptions do not completely describe the capability of a service and cannot be understood by software programs. Semantic Web Services (SWSs) are the result of the evolution of the syntactic definition of Web Services and the semantic Web. Representational techniques being developed for the Semantic Web can be used to augment these standards. The resulting Web Service specifications enable the development of software programs that can interpret descriptions of unfamiliar Web Services and then employ those services to satisfy user goals. The Semantic Web is a set of technologies for representing, and publishing on the Web, computer-interpretable structured information [1]. Semantic Web Services [2] is a research field that endeavours to apply these technologies to the description and use of Web Services. Whereas interoperability is the primary motivation for Web Services, automation of information use and dynamic interoperability are the primary objectives of Semantic Web Services.

1.1 Semantic Web Services will allow the semi-automatic and automatic annotation, advertisement, discovery, selection, composition, and execution of inter-organization business logic, making the Internet a global common platform by providing ontology-based service descriptions expressed in a formal language. Semantic Web Services aim to reduce the human effort required to build Service Oriented Architectures by enabling machines to understand the function and interfaces of Web services through the addition of semantics, by providing formal descriptions with well defined semantics. The research agenda for SWS identifies a number of key areas of concern, namely:

Description/annotation: To effectively perform operations such as the discovery of services, the semantics of the input/output data has to be taken into account. Hence, if the data involved in Web service operation is annotated using an ontology, then the added data semantics can be used in matching the semantics of the input/output data of the Web service with the semantics of the input/output data of the requirements.

Discovery: finding the Web service which can fulfill a task. Discovery usually involves matching a formal task description against semantic descriptions of Web services.

Selection : A selection of services is required if there is more than one service matching the request. Non-functional attributes such as cost or quality can be used for choosing one service.

Mediation: we can not assume that the software components which we find are compatible. Mediation aims to overcome all incompatibilities involved. Typically this means mismatches at the level of data format, message protocol and underlying business processes.

Composition: Composition or Choreography often no single service will be available to satisfy a request. In this case we need to be able to create a new service by composing existing components. Artificial Intelligence (AI) planning engines are

typically used to compose Web service descriptions from high-level goals.

Execution semantics: of a Web service encompasses the ideas of message sequence (e.g., request-response, request response), conversation pattern of Web service execution (peer-to-peer pattern, global controller pattern), flow of actions (sequence, parallel, and loops), preconditions and effects of Web service invocation, etc.

1.2 SWSs Approaches

The Major initiatives in the area of SWSs are documented by recent W3C member submissions: OWL-S [3], WSMO [4] and IRS-III [8]. The proposals differ in scope, modelling approach and the concrete logical languages used. These approaches are complementary in many ways. Each initiative can be characterized in terms of

(1) a conceptual model describing the underlying principles and assumptions; and

(2) a language or a set of languages that provide the means to realize the model.

2. OWL-S

OWL-S (formerly DAML-S) is emerging as a Web service description language that semantically describes the Web using OWL ontologies. OWL is a W3C standard OWL is a Description Logic-based Language, provides the basic constructs to describe ontologies . It provides a framework for describing both the functions and advertisements for Web Services, including a process specification language, a notation for defining process results, and a vocabulary for connecting these descriptions in OWL with syntactic specifications of service message structure in WSDL (and in other existing notations). OWL-S has been used in very different configurations, from traditional Web Service architectures that adopt the service-oriented architecture (SOA) 'triangle' set of interactions (among a service registry, producer, and consumer). The principal high-level objectives of OWL-S are

(1) to provide a general-purpose representational framework in which to describe Web Services;

(2) to support automation of service management and use by software agents;

(3) to build, in an integral fashion, on existing Web Service standards and existing Semantic Web standards; and

(4) to be comprehensive enough to support the entire lifecycle of service tasks.

The Overall structure of OWL-S ontology includes three primary sub-ontologies: the service profile, process model, and grounding.

Service Profile The service profile tells "what the service does". The profile can be used to advertise a service by describing its capabilities. Web Services are viewed as functions that produce a transformation in their inputs generating outputs. A profile description includes three types of information:

1. A human readable description of the service and its provider

2. A specification of the functionalities that are provided by the service

3. Attributes which provide additional information and requirements

The Service Profile describes what the service does by specifying the input and output types, preconditions and effects (IOPE).

• The inputs the service expects. The inputs are the object descriptions that the service works on;

• The output information returned the outputs are the object descriptions that it produces.

• The preconditions that have to be satisfied in order to use the service.

• An effect is a proposition that will become true when the service completes. The expected effects resulting from running the service.

In general, these propositions are not statements about the values of the inputs and outputs, but about the entities referred to in the inputs and the outputs. For example, a service that requires a user to have an account and a credit card might have inputs User and CreditCard and precondition:

hasAcctID(User, AcctID)

& validity(CreditCard, Valid))

(AcctID and Valid are local variables). The precondition requires the user to actually have an account with the service, with the account identifier AcctID. It also requires the credit card to have a known status Valid. In general, the effects of a service vary depending on conditions. In the case at hand, if the credit card is in fact valid, then let us suppose the service will replenish one's Pass balance to $25.3 If the credit card is not valid, then there is no change in the account balance. These are two separate results. The first might be indicated as:

Valid = OK |–>

output(Outcome<=Success)

& AcctBalance(25)

The second might be written:

Valid = Not-OK |–>

output(Outcome<=Failure)

Here, OK and Not-OK are values defined as elements of an OWL class AccountValidity.

The value of the output Outcome depends on tests that occur during the execution of the process, as does the effect from condition Valid=OK then AcctBalance(25). (Note that this test is performed by the service provider, not the service consumer, although if the consumer knew the credit-card status). A profile also allows the definition of service characteristics such as the category (refers to an ontology of services that may be on offer), the degree of quality, the quality guarantees , the geographic radius (refers to the geographic scope of the service), etc.

Process Model The service model describes "how a service works", to enable invocation, enactment, composition, monitoring and recovery. A process model decomposes into an ordered collection of processes. A process consists of other processes, in which case it is said to be a composite process and it is organized on the basis of some control flow structure. A process model allows various types of control flow structure including split, sequence, and non-deterministic

choice. The control constructs made available are the following :

• Sequence: a list of processes to be carried out in a specific order.

• Split: a bag of process components to be executed concurrently.

• Unordered: a bag of process components that can be executed in any order.

• Split Join: consists of concurrent execution of process components with barrier synchronization.

• Choice: allows a choice between alternative and execute one.

• If-then-else: class has properties if Condition, then, and else, which implement the statement.

• Repeat-until and repeat-while: Iterates execution of a bag of processes until/while a condition holds

If a process is not decomposable any further, it is said to be an atomic process and corresponds to operations that can be performed directly. Processes have inputs used to execute the process correctly. For a process to be executed, its preconditions need to be evaluated to true. The results of a process are described as a set of outputs.

Service Grounding The grounding maps the constructs of the process model onto detailed specifications of message formats and protocols. Grounding of a service specifies the details about transport protocols, message formats, serialization, addressing, and other service-specific details such as port numbers used in contacting the service. It answers to the question "How does a client access a service?"OWLS uses Web Service Description Language (WSDL) as a specification for messages exchanged between services and processes.

3. WSMO

WSMO: This initiative defines a model to describe semantic web services, based on the conceptual design set up in the WS Modeling Framework WSMF [5]. The Web Service Modeling Framework (WSMF) provides a model for describing the diverse aspects related to Web services. WSMF is the product of research on modeling of reusable knowledge components. WSMO is moreover accompanied by a formal language, the WS Modeling Language WSML [7], which allows one to write annotations of WSs according to the conceptual model, and by an execution environment WSMX [6] for the dynamic discovery, selection, mediation, and invocation of services. The latter distinguishes four elements: ontologies, services, goals and mediators. WSMF is based on:

1) a strong decoupling of the various components that realize an e-commerce application;

2) a strong mediation service enabling Web services to communicate in a scalable manner

WSMO identifies four main top-level elements:

Ontologies that provide the terminology used by other elements. The main objectives of the use of ontologies are to enhance interoperation by giving formal semantics to the information exchanged by Web services and facilitating the interoperation between humans and Web services. WSMO ontologies include concepts, relations, instances, and axioms, and information describing non-functional properties. The basic blocks of a WSMO ontology are concepts, relations, functions, instances, and axioms. Concepts are defined using a parent-child hierarchy and their attributes, including range specification. Relations describe interdependencies between a set of parameters. Functions are relations that have a unary range beside a set of parameters. Instances are individuals of concepts or relations, by specifying concrete values for attributes or parameters. Axioms are specified as logical expressions to formalize domain specific knowledge.

Goals that state the intentions that should be solved by Web Services. They are described at a high level and describe functionalities that a Web service should provide from the user perspective. WSMO follows a goal-driven approach. Requests and services are decoupled. A goal includes a requested capability definition (what is required from the service), a requested interface definition (which interface is desired) and some ontology imports for semantic contextualizing of the involved elements. In WSMO, a goal is described by non-functional properties, imported ontologies, used mediators, requested capability, and requested interface.

Web Services descriptions which describe various aspects of a service. A service definition in WSMO includes five description elements, namely:

•Non-functional properties, for data about the service creator, publisher, date, etc.

•Imported ontologies, to specify the semantic meaning of the concepts used in the service description.

•Used mediators, to assign the mediators used for importing ontologies and other tasks.

•Capability, which depict the functionalities provided by the service in terms of preconditions, postconditions, assumptions and effects. These properties determine the behaviour of the service, i.e. what is the state of the world before and after the execution of the service, and they are expressed as axioms in logic based language WSML.

Mediators: to resolve interoperability problems.*Mediators* connect heterogeneous components of a WSMO description which have structural, semantic or conceptual incompatibilities. Mismatches can arise at data level (when several distinct ontologies need to be integrated) or at process level (if a translation between communicating elements must be carried out). A mediator declaration includes non-functional properties, imported ontologies, a source component, a target component and a mediation service. There exist four types of mediators: ontology mediators, goals mediators, web service-to-goal mediators and web service-to-web service mediators.

4. IRS-III

IRS-III(Internet Reasoning System [8]) following the WSMO framework [9], provides at the semantic level a distinction between goals (i.e. abstract definition of tasks to be accomplished) and Web services (i.e. description of services that can achieve a goal) and as a result support capability-driven service matching and invocation . IRS-III automatically transforms programming code into a Web service. In addition, any service published on IRS-III automatically appears as a standard Web service to other Web service infrastructures. A

core design principle for IRS-III is to support capability-based invocation . IRS-III will:
a) discover potentially relevant Web services;
b) select the set of Web services which best fit the incoming request;
c) mediate any mismatches at the conceptual level;
d) invoke the selected Web services whilst adhering to any data, control flow and Web service invocation constraints.
IRS-III Server builds upon an *HTTP Server* written in Lisp which has been extended with a SOAP handler. At the heart of the server is the *SWS Library,* where the semantic descriptions associated with Web services are stored using representation language OCML [10] [Motta, 1998]. The library is structured into domain ontologies and knowledge models for goals, Web services and mediators as described previously . All information relevant to a Web service is stored explicitly within the library. Within IRS-III, a Web service is associated with an orchestration and choreography definitions. Orchestration specifies the control and data flow of a composite Web service, whereas, choreography specifies how to interact with a single Web service. The choreography component communicates with an invocation module able to generate the required messages in a SOAP format. A *Mediation Handler* descriptions including running data mediation rules, invoking mediation services and connecting goals and Web services.

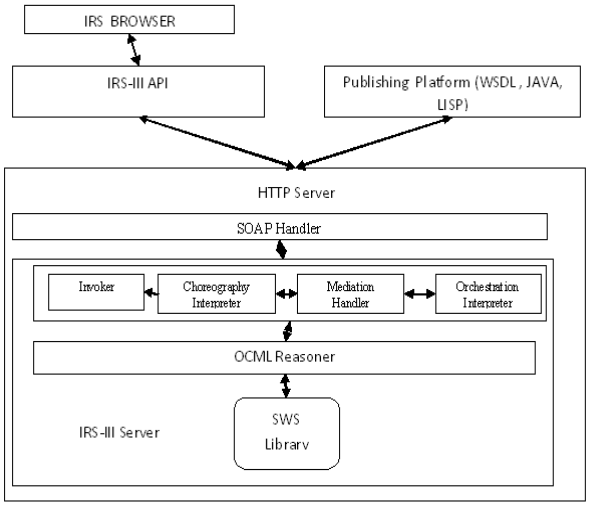

Figure 1 shows framework for IRS-III.

IRS-III Publishing Platforms: Publishing with IRS-III entails associating a deployed Web service with an IRS-III Web service description. When a Web service is published all of the information necessary to call the service - the host, port and path - is stored within the choreography associated with the Web service. Additionally, updates are made to the appropriate *Publishing Platform.* IRS-III contains publishing 15 platforms to support grounding to stand-alone Java and Lisp code and to Web services. Web applications accessible as HTTP GET requests are handled internally by the IRS-III server.

IRS-III Clients & API : IRS-III was designed for ease of use and, *IRS-III Browser* supports this by providing a goal-centric invocation mechanism. An IRS-III user simply asks for a goal to be solved and the IRS-III broker locates appropriate Web service semantic descriptions, using the mediators and then invokes the underlying deployed Web services. Additionally, the IRS-III client and publishing platforms interact with the IRS-III server through the API.
IRS-III, a framework for creating and executing semantic Web services, which takes a semantic broker based approach to mediating between service requesters and service providers. IRS-III approach provide three different applications to domains of Business Process Management, e-Learning and e-Science.

5. SWS approaches comparison

This comparison discusses the delivered results of IRS-III, OWL-S and WSMO as they represent the main approaches driving the implementation of Semantic Web Service components. Some of benefits of using OWL-S are:
1) Benefits to application developers: Ease of use, less code to write, code becomes reusable & less chance of misinterpretation.
2) Benefits to community at large: Everyone can understand each other's data's semantics, since they are in a common language.
But there are some of problems of Owl-S are as:-
OWL-S does not separate what the user wants from what the service provides. OWL-S is restricted to the service profile and cannot expressed the non-functional properties . OWL-S defines only one service model per service, so there is only one way to interact with the service. OWL-S does not explicitly provide linkage facilitate of hetero-geneous resources between one another . OWL-S is more mature in certain aspects (like choreography), but does not provides a more complete conceptual model.
WSMO provides a more complete conceptual model because it addresses aspects like goals and mediators. WSMO description of choreography and combination issues (based on Abstract State Machines) is expected to be more detailed and flexible than OWL-S one. It covers most of the problem of OWL-S approaches as nonfunctional properties can be expressed in any WSMO element, it is expected that there could be more than one way to interact with a particular service, WSMO allows the definition of multiple interfaces for a single service (using choreography and orchestration as choreography describes the external visible behavior of the service and an orchestration describes how other services are composed in order to achieve the required functionality of the service) , to facilitate linkage of heterogeneous resources between one another, various kinds of mediation are required.
IRS-III [8] is a framework and implemented infrastructure for Semantic Web Services that implements the WSMO descriptions & builds upon the previous version (IRS-II) by incorporating and extending the WSMO ontology within the IRS-III server, browser and API. The following table shows the high-level elements of each approach of SWS, including the application tools provided as well.

OWL-S focuses in ease of use, providing primitives not oriented to a particular application domain or problem, OWL-S is based on well-known OWL ontology language, concretely in its DL level, so OWL-S inherits OWL-DL decidability properties. The main contribution of the OWL-S approach is its service ontology, which builds on the Semantic Web stack of standards. OWL-S models capabilities required for Web services to the extent of grounding, which maps to WSDL descriptions. Additionally, the Daml consortium has put a lot of effort in representing the interactions among Web Services through the process model of the OWL-S service ontology. The invocation activity of OWL-S involves a decomposition of the process model. The discovery activity demonstrated in [11] relies on the extension of UDDI registry.

	OWL-S	WSMO	IRS-III
SWS Activities	Composition , Discovery , Invocation	Discovery	Publishing , Selection , Task Achievement
Language	OWL-S	WSML	OCML
Architecture	Daml-s Virtual Machine Matchmaker	Service Registry Profile Crawler	Server Publisher Client
Application Tool	WSDL2DAML-S	Query interface	IRS Browse Editior; Publisher; Java API

Table 1 shows comparison of Owl-S, WSMO, IRS-III

WSMO concentrates in solving integration difficulties with a language near to the user. Thus, WSMO description of choreography and combination issues (based on Abstract State Machines) is expected to be more detailed and flexible than OWL-S one. WSML language of WSMO is based on F-Logic, which covers similar features than description logics. WSMO defines a family of languages for ensuring interoperation with OWL and other types of formalisms, as logic programming and first order logic. The WSMF approach, although delivering a conceptual framework, invested considerable effort in bringing business requirements into account when proposing a conceptual architecture. In particular, a service registry has been proposed for which a high level query language is defined according to the service ontology [12]. WSMO distinguished characteristic is the inclusion of mediators in the ontology specification. IRS-III provides the representational and reasoning mechanisms for implementing the WSMO meta-model in order to describe Web Services. Additionally, IRS-III provides a powerful execution environment which enables these descriptions to be associated to a deployed Web Service and instantiated during selection, composition, mediation and invocation activities [13]. In IRS, service constraints (e.g. pre-conditions and post conditions) must be expressed in OCML but an OWL to OCML parser has recently been completed.

6. Conclusion

It has been concluded that for application communication in business environments through Web Services communication, description, and discovery features are covered, respectively, by SOAP,WSDL, and UDDI standard languages do not allow to perform some complex tasks, as integration or semantic Matching of services. To overcome these difficulties, several proposals are being brought up in research laboratories to combine Semantic Web description capabilities with Web Services current technologies. Some remarkable efforts are OWL-S, WSMO and IRS-III description formalisms, which are described in the paper.

In this paper we have overviewed the current standards, protocols, and supporting technologies for the development of Semantics Web Services approaches. None of the approaches described provide a complete solution according to the dimensions illustrated, but interestingly enough they show complementary strengths. For example, IRS-III has strong user and application integration support while OWL-S provides a rich XML-based service-ontology. WSMF has a comprehensive conceptual architecture, which covers requirements of one of the most demanding web based application area, namely e-commerce. These requirements reflect the way business clients buy and sell services. It is expected that the future standard for description of Semantic Web Services will be a synthesis of these approaches which will combine OWL-S use facilities, WSMO integration features and IRS-III rule-based mechanisms. Hence, submissions and discussions about the several challenges that Semantic Web Services poses are welcome in W3C organization.

References

[1] Berners-Lee, T., Hendler, J., Lassila, O.: *The semantic Web*. Scientific American, May 2001.

[2] McIlraith, S., Son, T.C., Zeng, H. *Semantic web services*. IEEE Intell. Syst. 16(2), 46–53 (2001).

[3] OWL Services Coalition, *OWL-S: Semantic Markup for Web Services*, Dec 2003, http://www.daml.org/services/owls/1.0/owl-s.html.

[4] Dumitru Roman, Holger Lausen, and Uwe Keller. *Web service modeling ontology(WSMO)*. Final Draft D2v1.3, WSMO, 2006. Available from: http://www.wsmo.org/TR/d2/v1.3/.

[5] Fensel, D., Bussler, C. *The Web Service Modeling Framework WSMF*. EletronicCommerce: Research and Applications. Vol. 1. (2002). 113-137 WSMO Working Group. Web Service Modelling Ontology Project. DERI Working Drafts. (2004) http://www.nextwebgeneration.org/projects/wsmo/

[6] *WSMO* working group, www.wsmo.org.

[7] *WSMX* working group, www.wsmx.org.

[8] CABRAL, L. AND DOMINGUE, J. 2005. *Mediation of Semantic Web Services in IRS-III*. In Proceedings of theInternational Conference on Service Oriented Computing (ICSOC 2005), Amsterdam, The Netherlands.

[9] J. Domingue, *"IRS-III: A Platform and Infrastructure for Creating WSMO-based Semantic Web Services"*, Proc. of the Workshop on WSMO Implementations, Germany, 2004.

[10] E. Motta, "*An Overview of the OCML Modelling Language*", the 8th Workshop on Knowledge Engineering Methods and Languages, 1998.

[11] Paolucci, M., Kawamura, T., Payne, T., Sycara, K.: *Semantic Matching of Web Services Capabilities.* In: Horrocks, I. Handler, J. (eds.): The Semantic Web – ISWC 2002 Proceedings. Lecture Notes in Computer Science, Vol. 2342. Springer-Verlag,Heidelberg (2002) 333-347.

[12] *SWWS Consortium. Report on Development of Web Service Discovery Framework.*October 2003. http://swws.semanticweb.org/public_doc/D3.1.pdf

[13] John Domingue, Liliana Cabral, Farshad Hakimpour, "*Demo of IRS-III: A Platform and Infrastructure for Creating WSMO-based Semantic Web Services*", Knowledge Media Institute, The Open University, Milton Keynes, UK 1989.

