Approaches to Developing Semantic Web Services

Jorge Cardoso

***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, will enable the automatic annotation, advertisement, discovery, selection, composition, and execution of inter-organization business logic, making the Internet become a common global platform where organizations and individuals communicate with each other to carry out various commercial activities and to provide value-added services. This paper deals with two of the hottest R&D and technology areas currently associated with the Web — Web services and the semantic Web. It describes how semantic Web services extend Web services as the semantic Web improves the current Web, and presents three different conceptual approaches to deploying semantic Web services, namely, WSDL-S, OWL-S, and WSMO.

***Keywords*—**Semantic Web, Web service, Web process, WWW

I. Motivation for the Semantic Web

THE information on the Web can be defined in a way that can be used by computers not only for display purposes, but also for interoperability and integration between systems and applications. One way to enable machine-to-machine exchange and automated processing is to provide the information in such a way that computers can understand it. This is precisely the objective of the semantic Web – to make possible the processing of Web information by computers. "The semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, better enabling computers and people to work in cooperation." [1]. The next generation of the Web will combine existing Web technologies with knowledge representation formalisms [2]. The semantic Web was made through incremental changes, by bringing machine-readable descriptions to the data and documents already on the Web.

Due to the widespread importance of integration and interoperability for intra- and inter-business processes, the research community has tackled this problem and developed semantic standards such as the Resource Description Framework (RDF) [3] and the Web Ontology Language (OWL) [4]. RDF and OWL standards enable the Web to be a global infrastructure for sharing both documents and data, which make searching for and reusing information easier and more reliable as well. RDF is a standard for creating descriptions of information, especially information available on the World Wide Web. What XML is for syntax, RDF is for semantics. The latter provides a clear set of rules for providing simple descriptive information. OWL provides a language for defining structured Web-based ontologies which allows a richer integration and interoperability of data among communities and domains.

Along with the development of the semantic Web, the Internet has created new organizational and working models which are based on electronic transactions. For example, Web services have been heralded as the next wave of Internet-based business applications that will dramatically change the use of the Internet [5]. With the development and maturation of infrastructures and solutions that support Web services, we expect organizations to incorporate e-services into their business processes.

Systems and infrastructures are currently being developed to support Web services. The main idea is to encapsulate an organization's functionality within an appropriate interface and advertise it as Web services. While in some cases Web services may be utilized in an isolated form, it is normal to expect Web services to be integrated as part of Web processes. There is a growing consensus that Web services alone will not be sufficient to develop valuable solutions due to the degree of heterogeneity, autonomy, and distribution of the Web. Several researchers agree that it is essential for Web services to be machine understandable in order to allow the full deployment of efficient solutions supporting all the phases of the lifecycle of Web services.

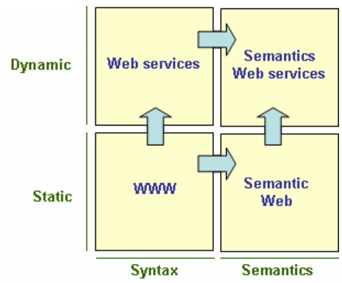

Fig. 1 The nature of semantic Web services

That is why many believe that a new Web will emerge in the next few years, based on the large-scale research and

Manuscript received November 7, 2005.

J. Cardoso is with the Department of Mathematics and Engineering, University of Madeira, 9050-390 Funchal, Portugal (phone: 291-705-156; fax: 291-705-199; e-mail: jcardoso@uma.pt).

development ongoing on the semantic Web and Web services. The intersection of these two, semantic Web services, may prove to be even more significant. Academia has mainly approached this area from the semantic Web side, while industry is beginning to consider its importance from the Web services side [6]. Semantic Web services are the result of the evolution of the syntactic definition of Web services and the semantic Web as shown in Fig. 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 become a global common platform where organizations and individuals communicate between each other to carry out various commercial activities and to provide value-added services.

II. The WWW

The Web was originally a vast set of static Web pages linked together. Many organizations still use static HTML files to deliver their information on the Web. However, in answer to the inherent dynamic nature of businesses, organizations are using dynamic publishing methods which offer great advantages over Web sites constructed from static HTML pages. Instead of a Web site comprising a collection of manually constructed HTML pages, server-side applications and database access techniques are used to dynamically create Web pages directly in response to requests from users' browsers. This technique offers the opportunity to deliver Web content that is highly customized to the needs of individual users. Nevertheless, the technologies available to dynamically create Web pages based on databases' information were insufficient for the requirements of organizations looking for application integration solutions. Businesses required their heterogeneous systems and applications to communicate in a transactional manner. The Extensible Markup Language [7] was one of most successful solutions developed to provide business-to-business integration. XML became a means of transmitting unstructured, semi-structured, and even structured data between systems, enhancing the integration of applications and businesses.

Unfortunately, XML-based solutions for applications and systems integration were not sufficient, since the data exchanged lacked of an explicit description of its meaning. The integration of applications must also include a semantic integration. Semantic integration and interoperability is concerned with the use of explicit semantic descriptions to facilitate integration.

III. Semantics

Semiotics is the general science of signs – such as icons, images, objects, tokens, and symbols – and how their meaning is transmitted and understood. A sign is generally defined as something that stands for something else. The human language is a particular case of semiotics. Compared to the human language, formal languages have precise construction rules for the syntax and semantics of programs. Semiotics is composed of three fundamental components: syntax, semantics, and pragmatics.

- **Syntax** deals with the formal or structural relations between signs (or tokens) and the production of new ones. For example, grammatical syntax is the study in which sequences of symbols are well formed according to the recursive rules of grammar. In computer science, if a program is syntactically correct according to its rules of syntax, then the compiler will validate the syntax and will not generate error messages. This however does not ensure that the program is semantically correct.
- **Semantics** is the study of relations between the system of signs (such as words, phrases, and sentences) and their meanings. As can be seen by this definition, the objective of semantics is totally different from the objective of syntax. The former concerns what something means while the latter pertains to the formal structure/patterns in which something is expressed.
- **Pragmatics** is the study of natural language understanding, and specifically the study of how context influences the interpretation of meaning. Pragmatics is interested predominantly in utterances, made up of sentences, and usually in the context of conversations [8]. The context may include any social, environmental, and psychological factors. While semantics deals with the meaning of signs, pragmatics deals with the origin, uses, and effects of signs within the content, context, or behavior in which they occur.

As we have seen, semantics is the study of the meaning of signs, such as terms or words. Depending on the approaches, models, or methods used to add semantics to terms, different degrees of semantics can be achieved. There are four main representations that can be used to model and organize concepts to semantically describe terms, namely: controlled vocabularies, taxonomies, thesaurus, and ontologies. These four model representations are illustrated in Fig. 2.

A. Controlled Vocabularies

Controlled vocabularies are at the weaker end of the semantic spectrum. A controlled vocabulary is a list of terms (e.g., words, phrases, or notations) that have been enumerated explicitly. All terms in a controlled vocabulary should have an unambiguous, non-redundant definition. Controlled vocabularies limit choices to an agreed upon unambiguous set of terms. The main objective of a controlling vocabulary is to prevent users from defining their own terms which can be ambiguous, meaningless, or misspelled. For example, Amazon.com has a controlled vocabulary which can be selected by the user to search for products. The vocabulary includes terms such as Books, Popular Music, Music Downloads, Classical Music, DVD, VHS, Apparel, Yellow Pages, Restaurants, etc.

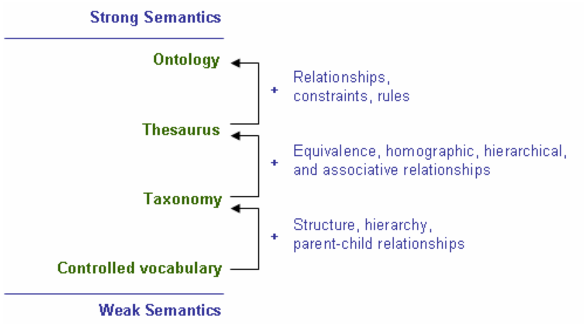

Fig. 2 Levels of semantics

B. Taxonomies

A taxonomy is a subject-based classification that arranges the terms in a controlled vocabulary into a hierarchy without doing anything further. A taxonomy classifies terms in the shape of a hierarchy or tree. It describes a word by making its relationship with other words explicit. The hierarchy of a taxonomy contains parent-child relationships, such as "is subclass of" or "is superclass of". A user or computer can comprehend the semantics of a word by analyzing the existing relationship between the word and the words around it in the hierarchy.

C. Thesaurus

A thesaurus is a networked collection of controlled vocabulary terms with conceptual relationships between terms. A thesaurus is an extension of a taxonomy by allowing terms to be arranged in a hierarchy and also allowing other statements and relationships to be made about the terms. A thesaurus can easily be converted into a taxonomy or controlled vocabulary. Of course, in such conversion, expressiveness and semantics are lost. According to the National Information Standards Organization [9], there are four different types of relationships that are used in a thesaurus: equivalence, homographic, hierarchical, and associative. An equivalence relationship says that a term t_1 has the same or nearly the same meaning as a term t_2. Two terms, t_1 and t_2, are called homographic if term t_1 is spelled the same way as a term t_2, but has a different meaning. This relationship is based on the degrees or levels of "is subclass of" and "is superclass of" relationships. The former represents a class or a whole, and the latter refers to its members or parts. This relationship is used to link terms that are closely related in meaning semantically but not hierarchically. An example of an associative relationship can be as simple as "is related to" as in term t_1 "is related to" in term t_2.

D. Ontologies

Ontologies are similar to taxonomies but use richer semantic relationships among terms and attributes, as well as strict rules about how to specify terms and relationships. In computer science, ontologies have emerged from the area of artificial intelligence. Ontologies have generally been associated with logical inference and recently have begun to be applied to the semantic Web.

An ontology is a shared conceptualization of the world. Ontologies consist of definitional aspects such as high-level schemas and assertional aspects such as entities, attributes, interrelationships between entities, domain vocabulary and factual knowledge – all connected in a semantic manner [10]. Ontologies provide a common understanding of a particular domain. They allow the domain to be communicated between people, organizations, and application systems. Ontologies provide the specific tools to organize and provide a useful description of heterogeneous content.

In addition to the hierarchical relationship structure of typical taxonomies, ontologies enable cross-node horizontal relationships between entities, thus enabling easy modeling of real-world information requirements. Jasper and Uschold [11] identify three major uses of ontologies:

1. to assist in communication between human beings
2. to achieve interoperability among software systems
3. to improve the design and the quality of software systems

An ontology is technically a model which looks very much like an ordinary object model in object-oriented programming. It consists of classes, inheritance, and properties [12]. In many situations, ontologies are thought of as knowledge representations.

IV. Web Services

Web services are modular, self-describing, self-contained applications that are accessible over the Internet [13]. Description of services in a language neutral manner is vital for the widespread use of Web services. Service providers describe their Web services and advertise them in a universal registry [14]. This enables service requestors to search the registry and find services, which match their requirements. XML, the emerging standard for data representation, has been chosen as the language for describing Web services. XML-based specification of a web service should involve both syntactic and semantic information. The syntactic details are about the physical location of the Web service, the operations supported etc. The semantic details give information related to properties, capabilities of the web service and other non-functional attributes. Quality of Service (QoS) [15] attributes give a clearer description of the service quality. Time, cost, reliability are some of the QoS attributes that can describe a service.

WSDL is the major language used to describe Web services. WSDL (Web Service Description Language [16]) is the W3C standard XML language for specifying the interface for a Web service. WSDL defines the syntactical information about a service.

V. Semantic Web Services

Many believe that a new Web will emerge in the next few years, based on the large-scale research and development ongoing on the semantic Web and Web services. The intersection of these two, semantic Web services, may prove to be even more significant. Academia has mainly approached this area from the semantic Web side, while industry is beginning to consider its importance from the Web services side [6]. Three main approaches have been developed to bring semantics to Web services:

- One approach to creating semantic Web services is by mapping concepts in a Web service description (WSDL specification) to ontological concepts. The WSDL elements that can be marked up with metadata are operations, messages, preconditions and effects, since all the elements are explicitly declared in a WSDL description. This approach is termed WSDL-S.
- The second approach uses OWL-S, a description language that semantically describes Web services using OWL ontologies. OWL-S services are mapped to WSDL operations, and inputs and outputs of OWL-S are mapped to WSDL messages.
- The third approach, the Web Services Modeling Ontology (WSMO), provides ontological specifications for the description of semantic Web services. One of the main objectives of WSMO is to give a solution to application integration problems for Web services by providing a conceptual framework and a formal language for semantically describing all relevant aspects of Web services.

These three approaches will be discussed in the following sections.

A. *Semantically Annotated Web services: WSDL-S*

One solution to create semantic Web services is by mapping concepts in a Web service description to ontological concepts. Using this approach, users can explicitly define the semantics of a Web service for a given domain. With the help of ontologies, the semantics or the meaning of service data and functionality can be explained. As a result, integration can be accomplished in an automated way and with a higher degree of success.

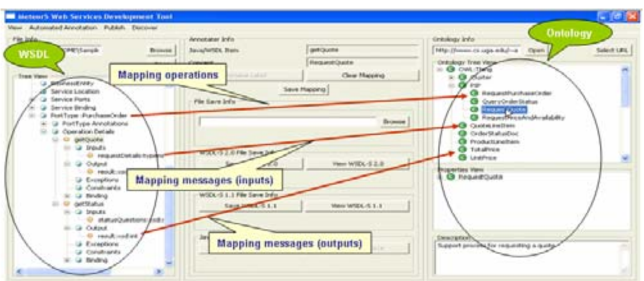

Fig. 3 Annotating Web services with ontological concepts

WSDL-S [19, 20] establishes mapping between WSDL descriptions and ontological concepts. The idea of establishing mappings between service, task, or activity descriptions and ontological concepts was first presented in [21]. Fig. 3 illustrates METEOR-S WSDL-S Annotator tool [19] and the mapping that have been established between WSDL descriptions and ontological concepts.

Based on the analysis of WSDL descriptions, three types of elements can have their semantics increased by annotating them with ontological concepts: operations, messages, preconditions and effects. All the elements are explicitly declared in a WSDL description.

Operations. Each WSDL description may have a number of operations with different functionalities. For example, a WSDL description can have operations for both booking and canceling flight tickets. In order to add semantics, the operations must be mapped to ontological concepts to describe their functionality.

Message. Message parts, which are input and output parameters of operations, are defined in WSDL using the XML Schema. Ontologies – which are more expressive than the XML Schema – can be used to annotate WSDL message parts. Using ontologies not only brings user requirements and service advertisements to a common conceptual space, but also helps to use and apply reasoning mechanisms.

Preconditions and effects. Each WSDL operation may have a number of preconditions and effects. The preconditions are usually logical conditions, which must be evaluated to true in order to execute a specific operation. Effects are changes in the world that occur after the execution of an operation. After annotating services' operations, inputs and outputs, preconditions and effects can also be annotated. The semantic annotation of preconditions and effects is important for Web services, since it is possible for a number of operations to have the same functionality, as well as the same inputs and outputs, but different effects.

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name = "StudentManagement"
  targetNamespace=
  "http:.../StudentManagement.wsdl20"
  xmlns="http://www.w3.org/2004/03/wsdl"
  xmlns:tns="http.../StudentManagement.wsdl20"
  xmlns:sm=“http:.../StudentMng.owl#"
  xmlns:mep=http:.../TR/wsdl20-patterns>

<interface name = "StudentManagementUMA">

  <operation name = "RegisterStudent"
    pattern = "mep:in-out" >

    <action element =
                  "sm:RegisterStudent" />

    <input messageLabel = ”student”
        element = "sm:StudentInfo" />

    <output messageLabel = ”ID”
        element = "sm:StudentID" />
  </operation>

  <operation name = "StudentInformation"
   pattern = "mep:in-out" >
```

```
    <action element =
                  "sm:StudentInformation" />

    <input messageLabel = "ID"
        element = "sm:StudentID" />

    <output messageLabel = "student"
        element = "sm:StudentInfo" />

  </operation>

  <operation name = "checkStatus"
   pattern="mep:in-out" >
...
  </operation>
</interface>
</definitions>
```

The WSDL-S specification indicates that the Web service supplies two operations: ‘RegisterStudent’ and ‘StudentInformation’. The first operation has an input named ‘student’, semantically described by the ontological concept “sm:StudentInfo”, and an output named ‘ID’, semantically described by the concept “sm:StudentID”. The operation ‘RegisterStudent’ is semantically annotated with the ontological concept “sm:RegisterStudent”. The second operation, ‘StudentInformation’, uses similar ontological concepts to annotate the input, output, and action. The ontological concepts are expressed in the ontology http://dme.uma.pt/jcardoso/StudentMng.owl#, which is specified using OWL [4].

To create, represent, and manipulate WSDL-S documents, WSDL4J (http://sourceforge.net/projects/wsdl4j/) can be used. WSDL4J provides JAVA API’s for WSDL parsing and generation. WSDL4J supports extensibility elements providing an easy mechanism to add new extensions. This allows WSDL to represent a specific technology under various elements defined by WSDL.

B. *Pure Semantic Web services: OWL-S*

OWL-S (formerly DAML-S) is emerging as a Web service description language that semantically describes the Web using OWL ontologies. OWL-S consists of three parts expressed with OWL ontologies: the service profile, the service model, and the service grounding. The profile is used to describe “what a service does”, with advertisement and discovery as its objective. The service model describes “how a service works”, to enable invocation, enactment, composition, monitoring and recovery. Finally, the grounding maps the constructs of the process model onto detailed specifications of message formats and protocols. These three parts and their relationships are illustrated in Fig. 4.

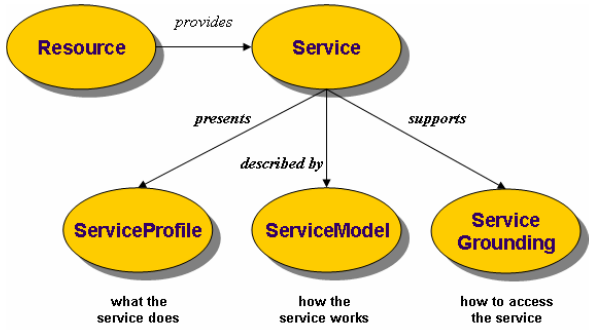

Fig. 4 Organization of OWL-S into modules

1. Service Profile

The service profile tells "what the service does". The profile can be used to advertise a service by describing its capabilities. This structure includes a description of what is accomplished by a service, limitations on service applicability and quality of service. 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 output information returned.
- The preconditions that have to be satisfied in order to use the service.
- The expected effects resulting from running the service.

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 following example, Congo.com provided by www.daml.org, which is available for download at http://daml.semanticweb.org/services/owl-s/1.0/, describes the inputs, outputs, preconditions and effects of the Congo book selling service.

```
...
<profile:hasInput
  rdf:resource=
    "http://www.daml.org/services...
                #ExpressCongoBuySignInInfo" />

<profile:hasOutput
  rdf:resource=
    "http://www.daml.org/services...
               #ExpressCongoOrderShippedOut" />

<profile:hasPrecondition
  rdf:resource=
  "http://www.daml.org/services...
                                #AcctExists" />
...
```

```
<profile:hasEffect
  rdf:resource=
  "http://www.daml.org/services...
              #ExpressCongoOrderShippedEff"/>
...
```

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 (this property provides qualifications about the service), the quality guarantees (guarantees that the service promises to deliver) the geographic radius (refers to the geographic scope of the service), etc.

2. Process Model

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 ifCondition, 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. Additionally, a set of effects that represent changes that result from the execution of the process are also asserted.

```
...
<process:CompositeProcess
                     rdf:ID="FullCongoBuy">
  <process:composedOf>
    <process:Sequence>
      <process:components
               rdf:parseType="Collection">
        <process:AtomicProcess
                rdf:about="#LocateBook"/>
        <process:CompositeProcess
                rdf:about="#CongoBuyBook" />
      </process:components>
    </process:Sequence>
  </process:composedOf>
  <process:hasInput>
    <process:Input
            rdf:ID="FullCongoBuyBookName">
      <process:parameterType
    rdf:resource=".../XMLSchema#string" />
      </process:Input>
  </process:hasInput>
...
```

The previous example describes a process named "FullCongoBuy" composed of a sequence of two sub processes: "locateBook" and "CongoBuyBook". The first sub process is an atomic process, while the second one is a composite process. The description of these processes is not shown in this example. The "FullCongoBuy" has several inputs, but only one is shown, the "FullCongoBuyBookName" which is of type string.

3. Service Grounding

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?" A grounding is the mapping of ServiceProfile and ServiceModel abstract specifications to the ServiceGrounding specification, a concrete level of specification. The main function of an OWL-S grounding is to map inputs and outputs of an atomic process to concrete messages which can be transmitted over various media. OWL-S uses Web Service Description Language (WSDL) as a specification for messages exchanged between services and processes.

Several message specifications could have been used, but since there is extensive work on the WSDL and it has a strong industry backing, WSDL has been selected. OWL-S atomic processes are mapped to WSDDL operations. The inputs and outputs of OWL-S atomic processes are mapped to WSDL messages. The following segment describes the grounding of OWL-S.

The tag **wsdlDocument** identifies the URI of the WSDL document that give the grounding.

```
<grounding:wsdlDocument>
  http://example.com/congo/congobuy.wsdl
</grounding:wsdlDocument>
```

The tag **wsdlOperation** identifies the URI of the WSDL operation corresponding to an atomic process.

```
<grounding:wsdlOperation>
...
  <grounding:portType>
    <xsd:uriReference
      rdf:value="http://example.com/congo/
      congobuy.wsdl#CongoBuyPortType"/>
  </grounding:portType>
...
  <grounding:operation>
    <xsd:uriReference
      rdf:value="http://example.com/congo/
```

```
      congobuy.wsdl#BuyBook"/>
  </grounding:operation>
...
</grounding:wsdlOperation>
```

The tag **wsdlInput** identifies the URI of the WSDL message definition that carries the input of an atomic process, and a list of mapping pairs, for the correspondence between OWL-S input properties and WSDL message parts. The tag **wsdlOutput** is very similar to the tag **wsdlInput** but applies to outputs.

```
...
<grounding:wsdlInputMessage
  rdf:resource="http://example.com/congo/
              congobuy.wsdl#CongoBuyInput"/>

 <grounding:wsdlInputs
              df:parseType="owl:collection">

  <grounding:wsdlInputMessageMap>
    <grounding:owlsParameter
                rdf:resource="#In-BookName">
...
  <grounding:wsdlMessagePart>
    <xsd:uriReference
      rdf:value="http://example.com/congo/
                     congobuy.wsdl#BookName">
  </grounding:wsdlMessagePart>
</grounding:wsdlInputMessageMap>
...
</grounding:wsdlInputs>
```

C. Another Semantic Approach: WSMO

Web Services Modeling Ontology (WSMO) provides ontological specifications for the main elements of semantic Web services. WSMO has been developed by the Digital Enterprise Research Institute (DERI), a European research organization that targets the integration of the semantic Web with Web services. WSMO approach is based on the Web Services Modeling Framework (WSMF) [22], a framework that provides the appropriate conceptual model for developing and describing Web services and their composition based on the maximal de-coupling and scalable mediation service principles. The main objective of WSMO is to solve the application integration problem for Web services, Enterprise Application Integration (EAI), and Service-Oriented Architectures (SOA), by providing a conceptual framework and a formal language for semantically describing all relevant aspects of Web services. These technologies will facilitate the automation of discovering, interoperating, composing, and invoking Web services over the Web. WSMO defines the modeling concepts for describing semantic Web services and includes the following four concepts: Web services, Goals, Ontologies and Mediators. WSMO and OWL-S aim at similar goals, i.e. providing a support for semantic Web services. Despite the similarity of the ultimate goal, the two approaches are very different. The two main components of WSMO initiative are:

Web Service Modeling Language (WSML). It provides a language to formally describe the elements defined in WSMO. WSML is based on different logical formalisms, namely Description Logics, First-Order Logic, and Logic Programming. WSML is divided into five subsets: WSML-Core, WSML-DL, WSML-Light, WSML-Rule and WSML-Full. The various WSML subsets provide different levels of expressiveness (in the same way that OWL provides different levels of expressiveness with OWL-Light, OWL-DL, and OWL-Full).

Web Service Execution Environment (WSMX). It is an execution environment that enables discovery, selection, mediation, invocation and interoperation of semantic Web services and provides a reference implementation of WSMO. WSMX provides an architecture including discovery, mediation, selection, and invocation and has been designed including all required supporting components enabling an exchange of messages between requesters and the providers of services.

1. Ontologies

Ontologies represent a key element in WSMO since they provide the terminology used by WSMO elements and give a shared conceptualization and a formal specification of the domain. 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 following example illustrates how ontologies are defined using WSMO [23]:

```
namespace
{_"http://example.org/tripReservationOntology#
",
  dc _"http://purl.org/dc/elements/1.1#",
  loc _"http://example.org/locationOntology#",
  po _"http://example.org/purchaseOntology#",
  foaf _"http://xmlns.com/foaf/0.1/",
  wsml        _"http://www.wsmo.org/wsml/wsml-
syntax#",
  prs _"http://example.org/owlPersonMediator#"
}
ontology
_"http://example.org/tripReservationOntology"
  nonFunctionalProperties
    dc#title    hasValue    "Trip    Reservation
Ontology"
    dc#creator                          hasValue
_"http://example.org/foaf#deri"
    dc#format hasValue "text/x-wsml"
  endNonFunctionalProperties
  importsOntology{
_"http://example.org/locationOntology",
_"http://example.org/purchaseOntology"}
  usesMediator
_"http://example.org/owlPersonMediator"

concept trip
  origin impliesType loc#location
  destination impliesType loc#location
  departure ofType _date
  arrival ofType _date
```

```
concept tripFromAustria subConceptOf trip
  nonFunctionalProperties
    dc#relation hasValue tripFromAustriaDef
  endNonFunctionalProperties

axiom tripFromAustriaDef
  definedBy
    forall {?x ,?origin}
      (?x memberOf tripFromAustria
        implies
        ?x[origin hasValue ?origin] and
        ?origin[loc#locatedIn          hasValue
loc#austria]).
```

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.

2. Web service

Web service descriptions describe the functional and behavioral aspects of a Web service and expose the interface of businesses on the Web. Web services are characterized by a set of capabilities which specifies the functionalities provided by a Web service. In WSMO, Web services are described from three different perspectives: non-functionality, functionality, and behavior. The following example illustrates the Book Ticket Web Service [23]:

```
webService
_"http://example.org/bookTicketWebService"
  importsOntology
_"http://example.org/tripReservationOntology"
  capability BookTicketCapability
  interface BookTicketInterface
```

A Web service definition in WSMO includes five description elements, namely:

- **Non-functional properties**. Describe a Web service using the Dublin Core Metadata Element Set. Examples of concepts used include dc:contributor, dc:coverage, dc:creator, and dc:date.
- **Imported ontologies**. Importing ontologies allows a modular approach for ontology design and are used to specify the semantic meaning of the concepts used in the service description.
- **Mediators**. Ontology mediators are used when an alignment of the imported ontology is necessary, since when importing ontologies some steps to transform the imported ontologies may be necessary.
- **Capability**. A capability defines the Web service by means of its functionality in terms of preconditions, post-conditions, assumptions, and effects.
- **Interface**. An interface describes how the functionality of Web services is accomplished. Interfaces describe the operational ability of Web services from choreography and orchestration perspectives. Choreography deals with the interaction that may be established with the Web service, while the orchestration deals with the set of functionalities that are required from other Web services.

The capability of a Web service defines its functionality and includes the following components: non-functional properties, imported ontologies, used mediators, shared variables, precondition, postcondition, assumption, and effect. To better understand how these components are used and expressed in WSMO an extract [23] of the capabilities of the Book Ticket service is shown as follows:

```
capability BookTicketCapability
  sharedVariables                 {?creditCard,
?initialBalance,   ?trip,   ?reservationHolder,
?ticket}
  precondition
    definedBy
      ?reservationRequest[
        reservationItem hasValue ?trip,
        reservationHolder              hasValue
?reservationHolder
      ] memberOf tr#reservationRequest and
      ?trip memberOf tr#tripFromAustria and
      ?creditCard[balance              hasValue
?initialBalance
              ] memberOf po#creditCard.
  assumption
    definedBy
      po#validCreditCard(?creditCard)and
        (?creditCard[type              hasValue
"PlasticBuy"] or
        ?creditCard[type               hasValue
"GoldenCard"]).
...
  postcondition
...
  effect
...
```

3. Goals

Goals represent user desires and the objectives that a user may have when he checks a Web service. 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.

The following scenario [23] describes the goal of a user who wants to buy a ticket from Innsbruck to Venice.

```
goal
_"http://example.org/havingAReservationInnsbru
ckVenice"
  importsOntology {

  _"http://example.org/tripReservationOntology
",

  _"http://www.wsmo.org/ontologies/locationOnt
ology"}
  capability
    postcondition
      definedBy
        ?reservation[
          reservationHolder             hasValue
?reservationHolder,
          item    hasValue    ?ticket]    memberOf
tr#reservation and
        ?ticket[trip  hasValue  ?trip]  memberOf
tr#ticket and
        ?trip[origin hasValue loc#innsbruck,
          destination    hasValue    loc#venice]
memberOf tr#trip.
```

4. Mediators

Mediators connect possibly heterogeneous resources of a WSMO description which have structural, semantic, or conceptual incompatibilities. They aim at automatically handling interoperability problems between different WSMO elements. 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 (ooMediators),
- goals mediators (goMediators),
- Web service-to-goal mediators (woMediators) and
- Web service-to-web service mediators (wwMediators).

An example of an ooMediators is given below. The mediator, called owlPersonMediator, imports the target ontology into the source ontology by resolving all the representation mismatches between the source and the target.

```
ooMediator
"http://example.org/owlPersonMediator"
  source
_"http://daml.umbc.edu/ontologies/ittalks/pers
on/"
  target
_"http://example.org/tripReservationOntology"
  usesService _"http://example.org/OWL2WSML"
```

VI. Semantics for Web Services

When bringing semantics to Web services, several types of semantics can be considered (illustrated in Fig. 5): Functional Semantics, Data Semantics, QoS Semantics, Execution Semantics, Domain Semantics, and Cultural Semantics. These different types of semantics can be used to represent the capabilities, requirements, effects and execution of a Web service. In this section we describe the nature of Web services and the need for a different kind of semantics for them.

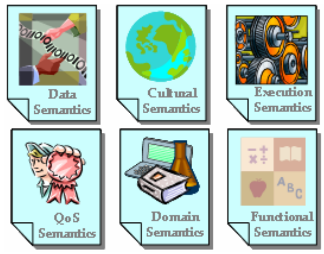

Fig. 5 Semantics for Web services

Functional Semantics. The power of Web services can be achieved only when appropriate services are discovered based on the functional requirements. It has been assumed in several semantic Web service discovery algorithms [24] that the functionality of the services is characterized by their inputs and outputs. Hence these algorithms look for semantic matching between inputs and outputs of the services and the inputs and outputs of requirements. This kind of semantic matching may not always retrieve an appropriate set of services that satisfy functional requirements. Though semantic matching of inputs and outputs are required, they are not sufficient for discovering relevant services. For example, two services can have the same input/output signature even if they perform entirely different functions. A simple mathematical service that performs the addition of two numbers taking the numbers as input and producing the sum as output, will have the same semantic signature as that of another service that performs the subtraction of two numbers that are provided as input and gives out their difference value as output. Hence matching the semantics of the service signature may result in high recall and low precision. As a step towards representing the functionality of the service for better discovery and selection, the Web services can be annotated with functional semantics. This can be done by having an ontology called Functional Ontology in which each concept/class represents a well-defined functionality. The intended functionality of each service can be represented as annotations using this ontology.

Data Semantics. All the Web services begin with a set of inputs and produce a set of outputs. These are represented in the signature of the operations in a specification file. However, the signature of an operation provides only the syntactical and structural details of the input/output data. These details (like data types, schema of a XML complex type) are used for service invocation. To effectively perform 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. The semantic discovery algorithm proposed in [24] uses the semantics of the operational data.

QoS Semantics. New trading models, such as e-commerce, require the specification of QoS metrics such as products or services to be delivered, deadlines, quality of products, and cost of service. In e-commerce and e-business, suppliers and customers define a binding agreement between the two parties, specifying QoS constraints. The management of QoS metrics of semantic Web services directly impacts the success of organizations participating in e-commerce. After discovering Web services whose semantics match the semantics of the requirements, the next step is to select the most suitable service. Each service can have different quality aspects and hence, service selection involves locating the service that provides the best quality criteria match. Service selection is also an important activity in Web service composition [21]. This demands management of QoS metrics for Web services. Web services in different domains can have different quality aspects. For organizations, being able to characterize Web processes based on QoS has several advantages: a) it allows organizations to translate their vision into their business processes more efficiently, since Web processes can be designed according to QoS metrics, b) it allows for the selection and execution of Web processes based on their QoS, to better fulfil customer expectations, c) it makes possible the monitoring of Web processes based on QoS, and d) it allows for the evaluation of alternative strategies when Web process adaptation becomes necessary.

Execution Semantics. The execution semantics of a Web service encompasses the ideas of message sequence, conversation pattern of Web service execution, flow of actions, preconditions and effects of Web service invocation, etc. Some of these details may not be meant for sharing and some may, depending on the organization and the application that is exposed as a Web service. In any case, the execution semantics of these services are not the same for all services and hence before executing or invoking a service, the execution semantics or requirements of the service should be verified.

Some of the issues and solutions with regard to execution semantics are inherited from traditional workflow technologies. However, the globalization of Web services and processes results in additional issues. In e-commerce, using execution semantics can help in dynamically finding partners that will match not only the functional requirements, but also the operational requirements like long running interactions and complex conversations. Also, a proper model for execution semantics will help in coordinating activities in transactions that involve multiple parties.

Domain semantics. With the spread of the Web, on both the public Internet and private Intranets, Web services will be owned and maintained by different organizations, distributed all around the world. As we have seen, an obvious problem is discovering a particular Web service that may be relevant to one's interest. It is hard for users to find the exact service that they are looking for because of the large number of Web service discovery mechanism returns. A search for a Web service in the registry can be a hit or a miss because UDDI does not provide domain specific information to assist the search process.

One approach to enhancing discovery and selection algorithms, as well as the interoperability of Web services, is to use domain-specific semantics. Domain semantics are of crucial importance since organizations have different needs, characteristics, vocabularies, contexts, standards, and protocols. For example, a multinational organization has obviously different needs compared to an organization that only has a regional base, and a financial organization has different requirements from a marketing organization [25]. Experience from business process management systems has shown that processes deployed for specific industries have specific characteristics, vocabularies, and semantics. Numerous processes, deployed for specific domains, have been studied and documented. Examples of specific domains include bio-informatics [26], genomics [27], healthcare [28], telecommunications [29], military [30], and school administration [31]. In the future, we may expect to see industry-specific registries to store Web services with semantic domain information. Each Web service needs to adhere to a semantic domain that establishes the terminology for interacting with other Web services. The development and exploration of semantic domains is a new concept and will require extensive research, development, and integration with actual Web service technologies.

Cultural semantics. E-commerce provides a worldwide set of opportunities and challenges as the Internet economy and globalization trends eliminate geographical boundaries. Web services are more likely to succeed when adapted specifically to the culture in which they are marketed. To make a Web service culture aware, it is necessary to develop international cultural semantics and localized cultural semantics.

With international cultural semantics, a semantic Web service is modified and adapted to use an ontology which is culturally independent. The objective is to design and implement "culturally and technically" neutral Web services. Internationalization helps to decrease localization cost and speed up time-to-market by addressing crucial technical, aesthetic, cultural, and linguistic issues. For example, if a Web service manipulates numbers, date, time and currencies, it is necessary to format them in a locale-independent manner.

Internationalization allows semantic Web services to be subsequently adapted to various languages and regions without engineering changes. The original independent ontology is replaced with culture specific ontology. Localization focuses on adapting details of a Web service to a specific locale. In order to create a product that appeals to the local culture, successful localization requires the deployment of ontologies that must take into account region-specific factors such as units of measurement (for example, length, area, volume, force, pressure), time zones, date formats, currencies, national holidays, icons, geographic examples, personal titles, and gender roles.

VII. Semantic Web Service Lifecycle

The lifecycle of semantic Web services includes the description/annotation, the advertisement, the discovery, the selection of Web services, the composition of Web services that make up Web processes, and the execution of Web services. In this section, we discuss the characteristics of each of these stages.

In order to fully harness the power of Web services, their functionality must be combined to create Web processes. Web processes allow representing complex interactions among organizations, representing the evolution of workflow technology. Semantics can play an important role in all stages of the Web process lifecycle. The main stages of the Web process lifecycle are illustrated in Fig. 6.

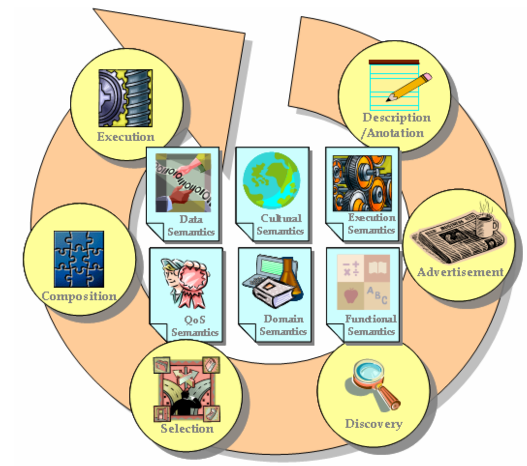

Fig. 6 Web process lifecycle and semantics (revised) [32]

A. Semantic Web Service Annotation

Today, Web service specifications are based on standards that only define syntactic characteristics. Unfortunately, this is insufficient, since the interoperation of Web services/processes cannot be successfully achieved. One of the best recognized solutions for solving interoperability problems is to enable applications to understand methods and data by adding meaning to them.

Many tools are available to create Web services. Primarily programs written in Java or any object oriented language can be converted into Web services. In technical terms any program that can communicate with other remote entities using SOAP [33] can be called a Web service. Since the development of Web services is the first stage in the creation of Web services, it is very important to use semantics at this stage. During Web service development, data as well as functional and QoS semantics of the service need to be specified.

All the Web services (operations in WSDL file [34]) take a set of inputs and produce a set of outputs. These are represented in the signature of the operations in a WSDL file. However, the signature of an operation provides only the syntactical and structural details of the input/output data.

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.

The Meteor-S Web Service Annotation Framework (MWSAF) [19] provides a framework and a tool to achieve automatic and semi-automatic annotation of web services using ontologies. Fig. 7 illustrates one solution to annotate WSDL interfaces with semantic metadata based on relevant ontologies [35].

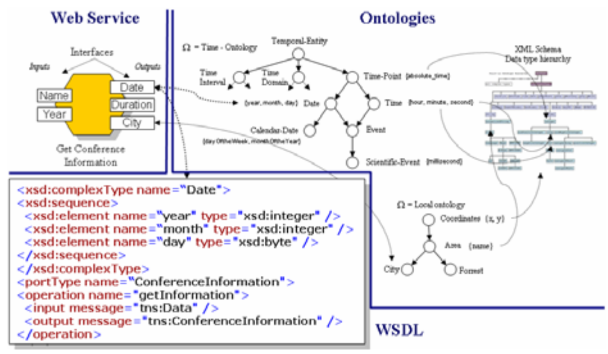

Fig. 7 Semantic annotation of a Web service specified with WSDL

A Web service invocation stipulates an input interface that specifies the number of input parameters that must be supplied for proper Web service execution and an output interface that specifies the number of output parameters to hold and transfer the results of the Web service execution to other services.

B. Semantic Web Service Advertisement

After the service is developed and annotated, it has to be advertised to enable discovery. The UDDI registry is supposed to open doors for the success of service oriented computing, leveraging the power of the Internet. Hence the discovery mechanism supported should be scaled to the magnitude of the Web by efficiently discovering relevant services among tens and thousands (or millions according to industry expectations) of Web services.

The present discovery supported by UDDI is inefficient, as services retrieved may be inadequate due to low precision (many services you do not want) and low recall (missed the services you really need to consider). Locating relevant services effectively and performing the search operation efficiently and in a scalable way, is what is required to accelerate the adoption of Web services. To meet this challenge, Web service search engines and automated discovery algorithms need to be developed. The discovery mechanisms supported need to be based on Web service profiles with machine process-able semantics.

C. Semantic Web Service Discovery

Given the dynamic environment in e-businesses, the power of being able to find Web services on the fly to create business processes, is highly desirable. Discovery is the procedure of finding a set of appropriate Web services, selecting a specific service that meets user requirements, and binding it to a Web process [36]. The Web service search to model Web process applications differs from the task search to model traditional processes, such as workflows. One of the main differences is in terms of the number of Web services available in the composition process. Thousands of Web services are potentially available on the Web. Therefore, one of the problems that needs to be solved is how to discover Web services efficiently [21].

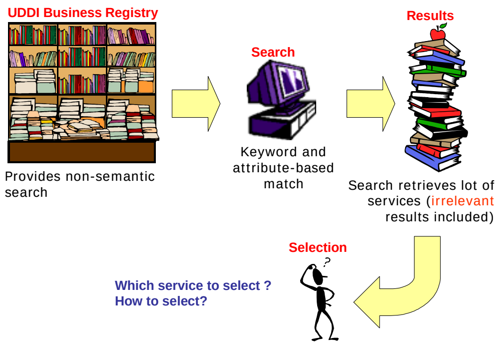

Fig. 8 State of the art in discovery (Cardoso, Bussler et al. 2005)

Currently, the industry standards available to register and discover Web services are based on Universal Description Discovery and Integration specification [14]. Unfortunately, discovering Web services using UDDI is relatively inefficient since the specification does not take into account the semantics of Web services, even though it provides an interface for keyword and taxonomy-based searching as shown in Fig. 8.

The key to the discovery of Web services is having semantics in the description of services itself [37] and then using semantic matching algorithms (e.g. [21]) to find Web services. One approach to semantic Web service discovery is the ability to construct queries using concepts defined in a specific ontological domain. By having both the description and query to declare their semantics explicitly, the results of discovery will be more relevant than keyword or attribute-based matching.

The semantic discovery of Web services has specific requirements and challenges compared to previous work on information retrieval systems and information integration systems. Several issues that need to be considered include:

- Precision of the discovery process. The search has to be based, not only on syntactical information, but also on data, functional, and QoS semantics.
- Enabling the automatic determination of the degree of integration of the discovered Web services and the Web process host.
- The integration and interoperation of Web services differs from previous work on schema integration due to the polarity of the schema that must be integrated [21].

Adding semantic annotations to WSDL specifications and UDDI registries allows improving the discovery of Web services. The general algorithm for semantic Web service discovery requires the users to enter Web service requirements as templates constructed using ontological concepts. Phases of the algorithm can be identified there. In the first phase, the algorithm matches Web services based on the functionality (the functionality is specified using ontological concepts that map to WSDL operations) they provide. In the second phase, the result set from the first phase is ranked on the basis of semantic similarity [21] between the input and output concepts of the selected operations and the input and output concepts of the initial template, respectively. The optional third phase involves ranking services based on the semantic similarity between the precondition and effect concepts of the selected operations and preconditions and effect concepts of the template.

D. Semantic Web Service Selection

Web service selection is a need that is almost as important as service discovery. After discovering Web services whose semantics match the semantics of the requirement, the next step is to select the most suitable service. Each service can have a different quality aspect and hence service selection involves locating the service that provides the best quality criteria match.

Service selection is also an important activity in Web service composition [21]. This demands management of QoS metrics for Web services. Web services in different domains can have different quality aspects. These are called Domain Independent QoS metrics. There can be some QoS criteria that can be applied to services in all domains irrespective of their functionality or specialty. These are called Domain Specific QoS metrics. Both these kinds of QoS metrics need shared semantics for interpreting them as intended by the service provider. This could be achieved by having an ontology (similar to an ontology used for data semantics) that defines the domain specific and domain independent QoS metrics.

E. Semantic Process Composition

The power of Web services can be realized only when they are efficiently composed into the Web process. This requires a high degree of interoperability among Web services. Interoperability is a key issue in e-commerce because more and more companies are creating business-to-customer and business-to-business links to manage their value chain better. In order for these links to be successful, heterogeneous systems from multiple companies need to interoperate seamlessly. Automating inter-organizational processes across supply chains presents significant challenges [38].

Compared to traditional process tasks, Web services are highly autonomous and heterogeneous. Sophisticated methods are indispensable to supporting the composition of Web process. Here again, one possible solution is to explore the use of semantics to enhance interoperability among Web services.

This stage involves creating a representation of Web processes. Many languages like BPEL4WS [39], BPML [40] and WSCI [41] have been suggested for this purpose. The languages provide constructs for representing complex patterns [42] of Web service compositions. While composing a process, four kinds of semantics have to be taken into account. The process designer should consider the functionality of the participating services (functional semantics), data that is passed between these services (data semantics), the quality of these services, the quality of the process as a whole (QoS semantics) the execution pattern of these services, and the pattern of the entire process (Execution semantics). Since Web process composition involves all kind of semantics, it may be understood that semantics plays a critical role in the success of Web services and in process composition.

F. Semantic Processes Execution

Web services and Web processes promise to ease several infrastructure challenges of the present, such as data, application, and process integration. With the emergence of Web services, workflow management systems (WfMSs) become essential to support, manage, enact, and orchestrate Web processes, both between enterprises and within the enterprise. Several researchers have identified workflows as the computing model that enables a standard method of building Web process applications and processes to connect and exchange information over the Web [22].

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.

Traditional formal mathematical models (Process Algebra [43]), concurrency formalisms (Petri Nets [44], state machines [45]) and simulation [46] techniques) can be used to represent execution semantics of Web services. Formal modeling for workflow scheduling and execution are also relevant [47]. With the help of execution semantics the process need not be statically bound to component Web services. Instead, based on the functional and data semantics, the list of Web services can be short-listed. Thereafter, QoS semantics can be used to select the most appropriate service and execution semantics can be used to bind the service to a process and to monitor process execution.

VIII. Conclusion

Since its creation, the World Wide Web has allowed computers only to understand Web page layout for display purposes, without having access to their intended meaning. The semantic Web aims to enrich the existing Web with a layer of machine-understandable metadata to enable the automatic processing of information by computer programs. The semantic Web is not a separate Web but an extension of the current one, in which information is given well-defined meaning, thereby better enabling computers and people to work in cooperation. To make possible the creation of the semantic Web the W3C (World Wide Web Consortium) has been actively working on the definition of open standards, such as the RDF (Resource Description Framework) and OWL (Web Ontology Language), and encourage their use by both the industry and academia. These standards are also important for the integration and interoperability for intra- and inter-business processes that have become widespread due to the development of business-to-business and business-to-customer infrastructures.

Web services are modular, self-describing, self-contained applications that are accessible over the Internet. Semantic Web services are the result of the evolution of the syntactical definition of Web services and the semantic Web. Three approaches have been developed to bring semantics to Web services. One approach to create semantic Web services, WSDL-S, is by mapping concepts in a Web service description (WSDL specification) to ontological concepts. The WSDL elements that can be marked up with metadata are operations, messages, preconditions and effects, since all the elements are explicitly declared in a WSDL description. The second approach uses OWL-S, a description language that semantically describes Web services using OWL ontologies. OWL-S services are then mapped to WSDL operations, and inputs and outputs of OWL-S are mapped to WSDL messages. The third approach, WSMO, provides the appropriate conceptual model for developing and describing Web services and their composition based on the maximal de-coupling and scalable mediation service principles. WSMO defines the modeling concepts for describing semantic Web services and includes Web services, Goals, Ontologies and Mediators concepts. WSMO and OWL-S aim at similar goals, i.e. providing a support for semantic Web services.

In order to fully harness the power of semantic Web services, their functionality must be combined to create semantic Web processes. Semantic Web processes allow complex interactions among organizations to be represented, representing the evolution of workflow technology. Semantics can play an important role in all stages of the Web process lifecycle. The lifecycle of semantic Web processes includes the description/annotation, the advertisement, the discovery, and the selection of Web services, the composition of Web services that make up Web processes, and the execution of Web processes.

References

[1] Berners-Lee, T., J. Hendler, and O. Lassila, The Semantic Web, in Scientific American, 2001.

[2] Grau, B.C. A Possible Simplification of the Semantic Web Architecture. in WWW 2004. 2004. New York, USA.
[3] RDF, Resource Description Framework (RDF). 2002, http://www.w3.org/RDF/.
[4] OWL, OWL Web Ontology Language Reference, W3C Recommendation. 2004, World Wide Web Consortium, http://www.w3.org/TR/owl-ref/.
[5] Fabio Casati, Ming-Chien Shan, and D. Georgakopoulos, E-Services - Guest editorial. The VLDB Journal, 2001. 10(1): p. 1.
[6] Cardoso, J., et al., Academic and Industrial Research: Do their Approaches Differ in Adding Semantics to Web Services, in Semantic Web Process: powering next generation of processes with Semantics and Web services, J. Cardoso and S. A., Editors. 2005, Springer-Verlag: Heidelberg, Germany. p. 14-21.
[7] XML, Extensible Markup Language (XML) 1.0 (Third Edition), W3C Recommendation 04 February 2004. 2005, http://www.w3.org/TR/REC-xml/.
[8] Wikipedia, Wikipedia, the free encyclopedia. 2005, http://en.wikipedia.org/
[9] NISO, Guidelines for the Construction, Format, and Management of Monolingual Thesauri. 2005, National Information Standards Organization: http://www.niso.org/standards/resources/z39-19a.pdf.
[10] Sheth, A., Semantic Meta Data For Enterprise Information Integration, in DM Review Magazine. 2003.
[11] Jasper, R. and M. Uschold. A framework for understanding and classifying ontology applications. in IJCAI99 Workshop on Ontologies and Problem-Solving Methods. 1999.
[12] Fensel, D., Ontologies: Silver Bullet for Knowledge Management and Electronic Commerce. 2001, Berlin: Springer-Verlag, http://www.cs.vu.nl/~dieter/ftp/paper/silverbullet.pdf.
[13] Curbera, F., W. Nagy, and S. Weerawarana. Web Services: Why and How. in Workshop on Object-Oriented Web Services - OOPSLA 2001. 2001. Tampa, Florida, USA.
[14] UDDI, Universal Description, Discovery, and Integration. 2002.
[15] Cardoso, J., et al., Modeling Quality of Service for workflows and web service processes. Web Semantics: Science, Services and Agents on the World Wide Web Journal, 2004. 1(3): p. 281-308.
[16] Chinnici, R., et al., Web Services Description Language (WSDL) Version 1.2, W3C Working Draft 24. 2003, http://www.w3.org/TR/2003/WD-wsdl12-20030124/.
[17] OWL-S, OWL-based Web Service Ontology. 2004.
[18] WSMO, Web Services Modeling Ontology (WSMO). 2004.
[19] Patil, A., et al. MWSAF - METEOR-S Web Service Annotation Framework. in 13th Conference on World Wide Web. 2004. New York City, USA.
[20] Rajasekaran, P., et al., eds. Enhancing Web Services Description and Discovery to Facilitate Composition. International Workshop on Semantic Web Services and Web Process Composition (SWSWPC 2004), ed. A.S. Jorge Cardoso. Vol. LNCS 3387. 2004, Springer-Verlag Heidelberg: California, USA. 147.
[21] Cardoso, J. and A. Sheth, Semantic e-Workflow Composition. Journal of Intelligent Information Systems (JIIS). 2003. 21(3): p. 191-225.
[22] Fensel, D. and C. Bussler, The Web Service Modeling Framework WSMF. Electronic Commerce Research and Applications, 2002. 1(2): p. 113-137.
[23] Feier, C., et al. Towards Intelligent web Services: Web Service Modeling Ontology (WSMO). in International Conference on Intelligent Computing (ICIC 2005). 2005. Hefei, China.
[24] Paolucci, M., et al. Importing the Semantic Web in UDDI. in Proceedings Web Services, E-Business and Semantic Web Workshop, CAiSE 2002. 2002. Toronto, Canada.
[25] Cardoso, J., R.P. Bostrom, and A. Sheth, Workflow Management Systems and ERP Systems: Differences, Commonalities, and Applications. Information Technology and Management Journal. Special issue on Workflow and E-Business (Kluwer Academic Publishers), 2004. 5(3-4): p. 319-338.
[26] Hall, R.D., et al., Using Workflow to Build an Information Management System for a Geographically Distributed Genome Sequence Initiative, in Genomics of Plants and Fungi, R.A. Prade and H.J. Bohnert, Editors. 2003, Marcel Dekker, Inc.: New York, NY. p. 359-371.
[27] Cardoso, J., Quality of Service and Semantic Composition of Workflows, in Department of Computer Science. 2002, University of Georgia: Athens, GA. p. 215.
[28] Anyanwu, K., et al., Healthcare Enterprise Process Development and Integration. Journal of Research and Practice in Information Technology, Special Issue in Health Knowledge Management, 2003. 35(2): p. 83-98.
[29] Luo, Z., Knowledge Sharing, Coordinated Exception Handling, and Intelligent Problem Solving to Support Cross-Organizational Business Processes, in Department of Computer Science. 2000, University of Georgia: Athens, GA. p. 171.
[30] Kang, M.H., et al. A Multilevel Secure Workflow Management System. in Proceedings of the 11th Conference on Advanced Information Systems Engineering. 1999. Heidelberg, Germany: Springer-Verlag.
[31] CAPA, Course Approval Process Automation (CAPA). 1997, LSDIS Lab, Department of Computer Science, University of Georgia: Athens, GA.
[32] Cardoso, J. and A.P. Sheth, Introduction to Semantic Web Services and Web Process Composition, in Semantic Web Process: powering next generation of processes with Semantics and Web services, J. Cardoso and A.P. Sheth, Editors. 2005, Springer-Verlag: Heidelberg, Germany. p. 1-13.
[33] Graham, S., et al., Building Web Services with Java: Making Sense of XML, SOAP, WSDL, and UDDI. 2002: SAMS.
[34] Christensen, E., et al., W3C Web Services Description Language (WSDL), http://www.w3.org/TR/wsdl. 2001.
[35] Cardoso, J., F. Curbera, and A. Sheth. Tutorial: Service Oriented Archiectures and Semantic Web Processes. in The Thirteenth International World Wide Web Conference (WWW2004). 2004. New York, USA.
[36] Verma, K., et al., METEOR-S WSDI: A Scalable Infrastructure of Registries for Semantic Publication and Discovery of Web Services. Journal of Information Technology and Management (in print), 2004.
[37] Sheth, A. and R. Meersman, Amicalola Report: Database and Information Systems Research Challenges and Opportunities in Semantic Web and Enterprises. SIGMOD Record, 2002. 31(4): p. pp. 98-106.
[38] Stohr, E.A. and J.L. Zhao, Workflow Automation: Overview and Research Issues. Information Systems Frontiers, 2001. 3(3): p. 281-196.
[39] BPEL4WS, Web Services. 2002, IBM.
[40] BPML, Business Process Modeling Language. 2004.
[41] WSCI, Web Service Choreography Interface (WSCI) 1.0. 2002, World Wide Web Consortium (W3C).
[42] Aalst, W.M.P.v.d., et al. Advanced Workflow Patterns. in Seventh IFCIS International Conference on Cooperative Information Systems. 2000.
[43] Bergstra, J.A., A. Ponse, and S.A. Smolka, Handbook of Process Algebra. 2001: Elsevier.
[44] Aalst, W.M.P.v.d., The Application of Petri Nets to Workflow Management. The Journal of Circuits, Systems and Computers, 1998. 8(1): p. 21-66.
[45] Hopcroft, J.E., R. Motwani, and J.D. Ullman, Introduction to Automata Theory, Languages, and Computation. 2000, Mass.: Addison-Wesley Publishing Company.
[46] Bosilj, V., M. Stemberger, and J. Jaklic, Simulation Modelling Toward E-Business Models Development. International Journal of Simulation Systems, Science & Technology, Special Issue on: Business Process Modelling, 2001. 2(2): p. 16-29.
[47] Attie, P., et al. Specifying and Enforcing Intertask Dependencies. in Proceedings 19th Intlernational Conference on Very Large Data Bases. 1993. Dublin, Ireland: Morgan Kaufman.

