Project Number:	**215219**
Project Acronym:	**SOA4ALL**
Project Title:	**Service Oriented Architectures for All**
Instrument:	**Integrated Project**
Thematic Priority:	**Information and Communication Technologies**

D11.1.1 OASIS Specifications for SEE

Activity N:	4 Exploitation and Impact Activities
Work Package:	11 Standardisation
Due Date:	M6
Submission Date:	29/08 /2008
Start Date of Project:	01/03/2006
Duration of Project:	36 Months
Organisation Responsible of Deliverable:	UIBK
Revision:	1.0
Author(s):	Omair Shafiq (UIBK)

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)		
Dissemination Level		
PU	Public	X
PP	Restricted to other programme participants (including the Commission)	
RE	Restricted to a group specified by the consortium (including the Commission)	
CO	Confidential, only for members of the consortium (including the Commission)	

Version History

Version	Date	Comments, Changes, Status	Authors, contributors, reviewers
0.1	25 June 2008	First version	Omair Shafiq (UIBK)
0.2	15 July 2008	Second version	Omair Shafiq (UIBK)
1.0	30 July 2008	Ready for review	Omair Shafiq (UIBK)
1.1	18 August 2008	Addressing reviewers comments	Omair Shafiq (UIBK)

Table of Contents

List of Figures

Glossary of Acronyms

Acronym	Definition
IRS	Internet Reasoning Service
RIF	Rule Interchange Format
RDF	Resource Description Framework
RM	Reference Model
SWS	Semantic Web Services
SOA	Service Oriented Architecture
SSOA	Semantic Service Oriented Architecture
SEE	Semantic Execution Environments
SOA-RM	Service Oriented Architecture Reference Model
SW	Semantic Web
TC	Technical Committee
WSMO	Web Service Modeling Ontology
WSML	Web Service Modeling Language
WSMX	Web Service Execution Environment

Executive summary

This deliverable provides an insight to the OASIS SEE (Semantic Execution Environment) Technical Committee and the specifications developed by it. The aim of the OASIS SEE TC is to provide guidelines, justifications and implementation directions for an execution environment for Semantic Web services. The resulting infrastructure aims to incorporate the application of semantics to service-oriented systems and provide intelligent mechanisms for consuming Semantic Web services. This document provides an overview of specifications from OASIS Semantic Execution Environment (SEE) Technical Committee, called as Reference Ontology for Semantic Service Oriented Architectures. The Reference Ontology for Semantic Service Oriented Architectures formalises and extends the SOA Reference Model.

1. Introduction to OASIS SEE TC

The aim of the OASIS SEE TC[1] is to provide guidelines, justifications and implementation directions for an execution environment for Semantic Web services. The resulting infrastructure will incorporate the application of semantics to service-oriented systems and will provide intelligent mechanisms for consuming Semantic Web services. The sections below present the purpose, scope, structure, as well as anticipated audience for the document.

1.1 Purpose

The technology of Semantic Web Services (SWSs) envisions easy access to various systems and facilitates the consumption of the functionality exposed by these systems on the Web. Seamless integration, ad-hoc cooperation between various business parties or dynamic collaborations on the Web, can be achieved only if tools for handling semantically enhanced services are provided.

The OASIS SEE TC aims to continue work initiated by the Web Service Execution Environment (WSMX) project and working group visible at http://www.wsmx.org and several other projects in Europe such as DIP[2], ASG[3] and other projects in the area of Semantic Web Services which will start in the coming months. The aim of the SEE TC is to provide guidelines, justifications and implementation directions for an execution environment for Semantic Web services. The resulting architecture will incorporate the application of semantics to service-oriented systems and will provide intelligent mechanisms for consuming Semantic Web services.

Service-oriented architectures anticipate a large number of ambient heterogeneous computational services that may be utilized in various combinations. However, a typical composition of services to meet a business goal is often an attempt to coordinate disparate resources from multiple sources: services that may not know, or fully understand, each other in advance. When planning to invoke multiple services, it is not always readily apparent whether the methods and outputs of one service meet the requirements of another. So some interpretation, mediation or common understanding is essential for any significant deployment. The SEE TC will define methods for using semantic technologies to solve these coordination and automation issues.

The TC will also define the functional components of such an SWS system and the semantics descriptions of these components' interfaces. The TC will also define a formal description of execution semantics of such a system. In addition, the TC will define a generic and open framework, using metadata, to allow for new components to be plugged into the system and made available to the execution engine dynamically. Further, after providing the basic methods described above, or in parallel if appropriate, the SEE TC will seek to develop specifications addressing specific problem sets covering the spectrum from a general purpose environment to a specific business-domain-focused on applications of Semantic Web Services to financial, telecommunication, military and e-Government.

In the course of existing research, it has become clear that Semantic Web services and Grid Computing are closely related research activities with many shared objectives. Both address

[1] http://www.oasis-open.org/committees/semantic-ex/

[2] http://dip.semanticweb.org

[3] http://asg-platform.org

distributed computing systems from different perspectives and we believe that they should be investigated in parallel as complementary technologies enabling the next era of internet applications. In our vision, the usage of arbitrary resources (physical or logical) for building complex business applications should be simplified, so that their discovery, deployment, composition, provisioning and management, can be performed by the means of semantic-enhanced services. To reach this goal, on the one hand, current Grid technologies need to be extended to support semantically enriched resource descriptions and services, e.g. to simplify their discovery and composition. On the other hand, Semantic Web services technology has to be extended to support resource management, including dynamic provisioning of services and resources, execution management, and support of security-related issues concerning virtual organization management. This TC aims to combine Grid Computing with Semantic Web Services technologies and to take advantage of their different perspectives to provide architecture of the infrastructure for machine-to-machine enabled communication and cooperation.

The SEE TC's efforts will foster compatibility across specifications developed for Semantic Web Services, and where possible re-use existing standards and methods that already have been carried in areas of Semantic Web and Web Services. This TC will engage with industry, academic and research communities to facilitate understanding, awareness and possible collaborations regarding emerging semantic technologies and research applicable to semantically-aware Web Services.

1.2 Scope

Initial requirements have indicated that Semantic Web services systems should enable automatic or semi-automatic discovery, negotiation, selection, composition, mediation, invocation and interoperation of multiple services. The SEE TC will assess the subsequent and related works and implementation experience of existing efforts in a variety of sectors (financial, telecommunication, e-health and e-government) to define and implement these functions related with Semantic Web Services. Based on those experiences, the detailed analysis of requirements for Semantic Web services Architecture will be provided.

The SEE TC will provide a test-bed for the Web Services Modeling Ontology (WSMO), which is anticipated as a contribution for use by the TC on a non-exclusive basis, and will seek to demonstrate the viability of using WSMO concepts, relationships and definitions as a means to achieve successful dynamic interoperation of multiple ambient services, whether or not they share a common design or source.

The TC anticipates contribution of the draft WSMX specifications and WSMO ontology on a non-exclusive basis. Other contributions will be accepted for consideration without any prejudice or restrictions, and evaluated on their technical merit, as long as the contributions conform to this charter.

Following a top-down, component based development approach, the TC will provide a whole framework capable of carrying out the dynamic discovery, mediation, selection, invocation and inter-operation of Web Services and any other functionality which will be revealed during the requirements analysis phase. While the focus of this group will remain on a high level semantic description of components interfaces, the TC will seek tight cooperation with any group working on semantics-enabled functional components that fulfill the requirements of such system.

The SEE TC will not implement actual software products or solutions based on the specifications developed along the course of work of this group.

1.3 Structure of the document

It is assumed that readers who are not familiar with SOA concepts and terminologies read first the SOA Reference Model [1] document since this document builds on top of its concepts. Furthermore, readers who are new to the concept of Semantic Technologies are encouraged to read this document in its entirety.

Section 1 introduces the OASIS SEE TC and its objectives. Furthermore, it presents scope and anticipated audience for the specifications.

Section 2 introduces the Semantic SOA Reference Ontology and how it relates to other work (in particular the SOA RM). It defines the audience and also provides a description of the notational conventions used in this document. Both of these elements are important in order for the reader to understand the content of the rest of the document. It further provides an overview of Semantics and how they interrelate with SOAs. It starts by describing the deficiencies of the classical SOA and the problems in building them.

Section 3 describes the SOA Reference Model [1] and builds on top of this by introducing new key concepts required for SSOAs. It first describes what we understand by a service followed by the dynamics of a service – how the service is perceived by the real world. Other related concepts are also described (including, for example, the behavior of the Web service). Section 3 shows the differences between the classical SOA RM and the SSOA RM and provides the necessary building blocks for specifying the Reference Ontology.

Section 4 defines the Reference Ontology for SSOAs. The ontology is first described using Concept Maps and UML Diagrams (notation described in Section 1.4 below). It is then formally described using WSML [7] in Appendix B. Note that any other Ontology language (e.g. OWL) can be used to define such an Ontology. We chose WSML since it provides an easy to use syntax and provides different language variants for different levels of logical expressivity.

The glossary provides definitions of terms that are relied upon within the document. Terms that are defined in the glossary are marked in **bold** at their first occurrence in the document.

Note that while the concepts and relationships described in this document may apply to other “service” environments, the definitions and descriptions contained herein focus on the field of software architectures and make no attempt to completely account for their use outside of the software domain. Examples included in this document, which are taken from a variety of domains, are used strictly for illustrative purposes.

1.4 Anticipated Audience

The anticipated audience for this work includes all OASIS Web Service and ebXML TCs, non-OASIS Web Service standards groups, Semantic Web Services research and interest groups, SOA architects and programmers, vendors and users. The work should be of interest to anyone involved with Semantic Web Services and more generally also in Service Oriented Architectures (SOAs), i.e.

- Architects and developers designing, identifying or developing a system based on the Service Oriented Architectures;
- Standards architects and analysts developing specifications that rely on Service Oriented Architecture concepts;
- Decision makers seeking a "consistent and common" understanding of Service Oriented Architectures;
- Users who need a better understanding of the concepts and benefits of Service

Oriented Architectures;

- Academics and researchers that are researching within the Semantic Web and Semantic Web Service communities;
- I.T. consultants that provide businesses with support on Semantic technologies and SOAs in general.

2. Semantics and SOA

Although Service Oriented Architectures (SOAs) have gathered a lot of attention within business organizations, for a long time there was still no clear understanding of what an SOA in fact is. SOA was consequently defined in the SOA Reference Model [1]. However, with the emerging **Semantic Web** technologies, in particular **Semantic Web Services (SWSs)**, new breeds of SOAs are being developed, namely **Semantic Service Oriented Architectures (SSOAs)**. SSOAs use semantic technologies to further solve problems that SOAs are limited by. They provide a means for further automation for service consumers' tasks, particularly service discovery, selection, composition and execution, as well as easing general interoperability issues between services. In order to use the semantic descriptions present in a SSOA to automated such SOA features, a set of platform services that provide this automation functionality are required within the SSOA. These services are collectively termed a **Semantic Execution Environment (SEE)** for Semantic Web Services, with a SEE being at the core of a SSOA. There are a number of different implementations of SEEs currently under development in the research community, which have some common features. Thus the purpose of this document is to define an extended reference model for SSOAs, as supported by SEEs. This model will be defined formally using an ontology. The aim of this ontology is to provide a point of reference formally specified so that it can support the definition and development of SSOAs.

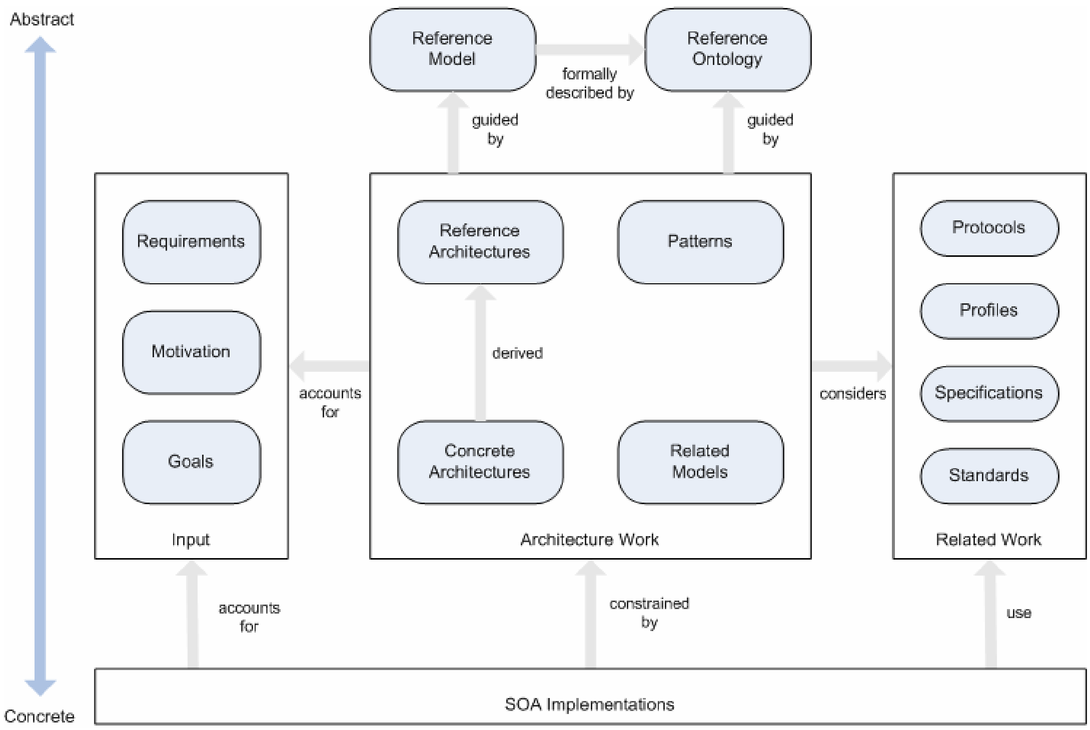

Figure 2-1 – Relationship of the Reference Ontology to Other SOA Specifications and Standards

Figure 2-1 depicts how the Reference Ontology relates to other pieces of work within the SOA community. The figure is derived from Figure 1 in the SOA Reference Model document [1] and introduces the Reference Ontology alongside the Reference Model element. The Reference Ontology presented in this document is a further step towards formalization of the Reference Model but also accommodates the extensions associated with Semantic Web Services resulting in Semantic SOAs. Since the start of this work, the SOA-RM committee have also started work on a Reference Architecture, but we shall take this to mean our own

Semantic SOA Reference Architecture, and Concrete Architectures refer to implementations of semantics-enabled SOAs such as WSMX [2], IRS III [3] and METEOR-S [4]. The Related Models include the Web Service Modeling Ontology (WSMO) 5, Semantic Annotations for WSDL and XML Schema (SAWSDL) [8], the Web Ontology Language for Services (OWL-S)[4] [9] and the Semantic Web Services Ontology (SWSO) [10].

Patterns fulfill the same role in Semantic- as in pre-Semantic- SOA, which is to say that they define more specific categories of service-oriented designs. The Protocols and Profiles (those considered as part of the related work) are the same as for classical SOAs. However, with respect to Specifications and Standards, we further take into account emerging Semantic Web Languages such as the OWL, RDF and RIF standards from W3C, and the WSML and SWSL defacto standards. These “standards” play a very important role since they are the pillars of Semantic Technologies. The Input features (Requirements, Motivation and Goals) are the same as for SOAs, with the addition that we have more emphasis on automation, as stated earlier.

2.1 Introduction

As noted in the Reference Model for Service Oriented Architecture (SOA-RM) committee specification, the notion of Service Oriented Architecture has received a lot of attention in the software design and development community. According to the SOA-RM, a “Service Oriented Architecture (SOA) is a paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains.” Service Oriented Architecture provides an architectural mechanism for building applications from unassociated units of functionality, called services. The perceived value of SOA is that it provides a powerful framework for matching needs and capabilities and for combining capabilities to address those needs, by enhancing the ability of adapting applications more quickly to changes in market conditions and improving the reusability, modularity, composability and interoperability of functionality.

A service, in the context of SOA, refers to a software mechanism that provides access to a capability that may have a real world effect or results in the exchange of information. Such services can be implemented leveraging many different standards and technologies, including Web services using WSDL descriptions and SOAP messaging.

Building Service Oriented Architectures using existing services still involves substantial human effort in the process of finding and using appropriate services. The need for human intervention can be attributed partly to the fact that standards that are typically used for describing services (e.g. WSDL), only focus on the syntactic aspect of the service interface, and provide little support for finding and using services that provide the appropriate desired functionality. In this “classical SOA” scenario, developers building an application using SOA, typically look for services that are available, either within their company’s repository of services or in remote locations. Each time a need to invoke a service is identified, a set of candidate services must be found browsing in repositories (e.g. UDDI or ebXML repositories). While keywords and text search features can be leveraged to identify candidate service, the syntactically focused descriptions typically require evaluation by a human before a service can be used. In many instances further human interaction between the developer on the consumer side and the service provider is required to clarify the functionality and the meaning of the information that is being exchanged. Then tests can be performed on the candidate services. Finally, a service may be selected and added to the application.

[4] It should be noted that no Semantic Execution Environments exist for OWL-S; however a list of all OWL-S tools is available as http://www.daml.org/services/owl-s/tools.html

Not only is this process labour intensive, but the solution is fairly static, limiting the ability to adapt to changes quickly, which is a key promise of the SOA approach. Changes, whether it is new services that provide improved functionality or unavailability of currently used services, typically require human interaction in the classical SOA. The goal of a Semantically-enabled SOA is to add features that can help overcome these limitations and provide mechanisms to automate tasks that currently require human intervention.

2.1.1 Semantics

A key limitation of a “classical SOA” as mentioned above is that the standards used for describing Web services are syntactic in nature and lack the ability to provide further meaning about a service.

Semantics is the study of meaning. A formal semantic description offers the opportunity of providing a mechanism for describing things more clearly and extensively. A formal semantic description is unambiguous within the context of the formalism and opens the opportunity for automated reasoning. Semantics come in many forms. Very basic forms of semantics include annotations or tags that can be associated with an entity in order to give a description of what that thing is. Annotations or tags can be seen in action on sites like flickr.com®, where they are used for denoting what content appears in a particular picture or what a picture is about. This mechanism, of course, is very rudimentary. To bring more meaning to the annotations, taxonomies can be introduced. Such structures give a mechanism for providing a controlled vocabulary of terms (i.e., a controlled set of annotations) and the relationship between them. For example we can state that the term banana is a sub class of the term fruit. This additional semantic information enables us to reason about the semantic descriptions we have and make decisions based on the semantic descriptions, for example the query “show me all photos containing a piece of fruit” is posed, then those pictures that are annotated with the term banana would be found, as banana is a subclass of fruit. To add more semantics we can go even further and allow logical expressions to be added to taxonomies to turn them into ontologies, such that more complicated relationships between entities can be expressed. The addition of axiomatic information in this way also allows for much more sophisticated reasoning to take place and for new information to be inferred from existing information, for example the axiom “all fruit is edible” placed in a reasoner with the previous example would allow the fact “bananas are edible” to be inferred and thus queries like “show me all photos containing things that are edible” would find pictures of bananas.

2.2 Applying Semantics to SOA

As indicated earlier, the syntactically focused descriptions of services in the “classical SOA” scenario limits the ability to automate tasks that are important for a quickly and reliably adapting to changes. The idea here is to apply semantics to SOA and enhance service descriptions with additional semantic information that can be used in conjunction with semantic processing mechanisms (i.e., mediation).

By extending ontologies to describe services in a SOA, a machine can reason about the functionality they provide, the mechanism to invoke them, and the data they expect as input and return as output. In other words, each service that currently has a syntactic description (i.e., a WSDL document) will also have a semantic description in some formalism. Thus services within a Semantic SOA are not a reinvention of services, but an enhancement of them. In order to effectively describe services semantically we need to have an understanding of what elements need to be modeled within our semantic description. Within this document you will find the Reference Ontology for Service Oriented Architectures, which provides such a description of what elements need to be modeled in order to effectively provide semantic description for services and build a SOA that is semantically-enabled, referred to as a Semantic SOA (SSOA).

Once services are described semantically, many of the tasks previously requiring human intervention in building and maintaining and application using SOA can be automated. For example, services can be discovered based upon the functionality they advertise in their semantic description, can be selected based upon the advertised (or observed) quality of the service, heterogeneity issues with respect to the data they exchange or the process to invoke them can be mediated. This allows for a SSOA, to dynamically bind to services at run time, removing the hard-wired behaviours that are typically for classical SOAs. When new services appear on the market that fulfill functionality needed by the application, they can be considered alongside existing services that are being used already by the application and may be selected over these existing services based on the requirements of the application. Also if a given service that is usually used by the application is no longer available, it can be automatically replaced by another service that fulfills the same function.

3. SOA Reference Model

The notion of Service Oriented Architecture has been greatly used in the last couple of years by the software design and development communities. Yet, the various and very often conflicting definitions and terminology for SOA and its elements could hamper the adoption process and threaten the success and the impact of this technology. In order to provide a standard reference point in the design and implementation of SOAs the OASIS SOA-RM Technical Committee[5] proposes an abstract framework for understanding the main entities and the relationships between them within a services oriented environment [1].

The resulting specification is a SOA Reference Model (SOA-RM), which is not directly dependent of any standards, technologies and implementation details. Its goal is to define the essence of Service Oriented Architecture, a normative vocabulary and a common understanding of SOA. The Reference Ontology takes this reference model as a starting point in defining the main aspects of a Semantically-enabled Service Oriented Architecture and it specifies how the normative elements of the SOA-RM can be augmented with semantics. As a consequence, this section gives a brief overview of the SOA-RM, along the several aspects it covers: the notion of *service*, the *dynamics of service* and the service-related concepts such as *service description*, *service execution context* and *service contracts and policies*.

3.1 What is a service?

SOA-RM defines a service as "…*a mechanism to enable access to one or more capabilities, where the access is provided using a prescribed interface and is exercised consistent with constraints and policies as specified by the service description.*" It identifies four main aspects regarding the service that have to be considered in any SOA:

- A service *enables access to one or more capabilities*;
- A service *enables access through a prescribed interface*;
- A service is *opaque to the service consumer* except from the information and behavioural models in the interface and the information requires to assess if a service meets the requesters needs;
- *Consequences of invoking a service* should either be response information to the invocation or a change to the shared state of the defined interface.

It is important to note that SOA-RM makes a clear distinction between the capability of a service (i.e. some functionality created to address a need) and the point of access where the capability can be consumed in the context of SOA.

3.2 Dynamics of Services

SOA-RM also provides guidelines regarding the interactions of the requester with a service. As such, it identifies three fundamental concepts related with dynamics of the service: *Visibility, Interaction* and *Real World Effect* (see Figure 3-1).

[5] For more details, see http://www.oasis-open.org/committees/soa-rm.

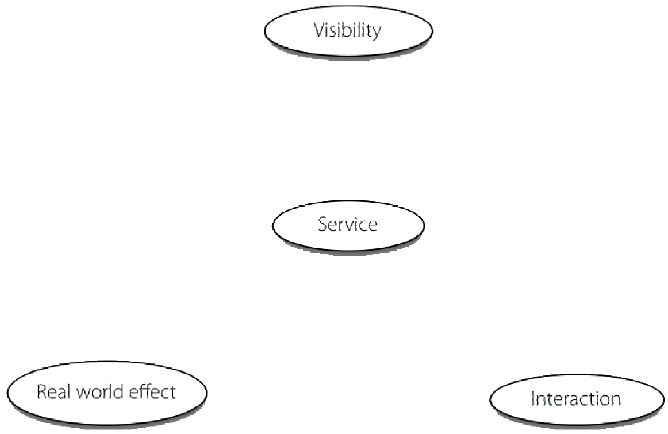

Figure 3-1 - Fundamental Concepts of Service Dynamics (from [1]) [Concept Map]

Visibility in terms of SOA-RM is characterized in terms of *Awareness*, *Willingness* and *Reachability* (see Figure 3-2) where:

- *Awareness* is the state whereby the service requester is aware of the service provider or the other way around. It is normally achieved by having either the requester or the provider discovering the information the other party published in for example a public directory.
- *Willingness* concerns the intent to communicate. Even if the discovery process has been successful, without willingness to communicate from both requester and provider the interaction will fail.
- *Reachability* is the state that characterizes service participants that are able to interact, for example by exchanging information.

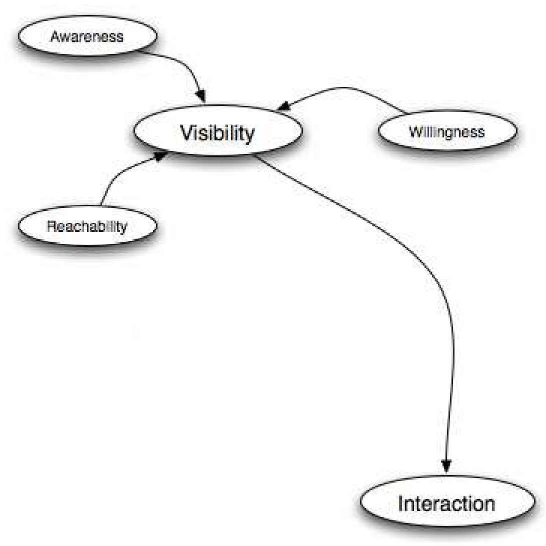

Figure 3-2 - Service Visibility (adapted from [1]1) [Concept Map]

The *interaction* with a service is reflected by the actions performed on the service, for example exchanging messages with the services. According to SOA-RM the key concepts affecting the interaction with a service are the following (see Figure 3-3):

- *Information Model* of a service characterizes the information that may be exchanged with the services and only descriptions of data and information that can be potentially exchanged with the service are included in the information model. The information model can be also portioned in:
  - *Structure (Syntax)* refers to the representation, structure, and a form of information;
  - *Semantics* refers to the actual interpretation and intent of the data. Semantics becomes important especially when interaction occurs across ownership boundaries since the interpretation of data must be consistent between the participants in a service interaction.
- *Behavior Model* deals with *"knowledge of the actions invoked against the service and the process or temporal aspects of interacting with the service".* It consists of two distinct aspects:
  - *The action model* characterizes the actions that can be invoked against the service. Since a great part of the behavior implied by an action is private, the public view of the service includes the implied effects of actions;
  - *The process model* defines temporal relationships of actions and events associated when interacting with a service. SOA-RM does not fully define the process model since it could include aspects that are not strictly part of SOA, e.g. orchestration of services.

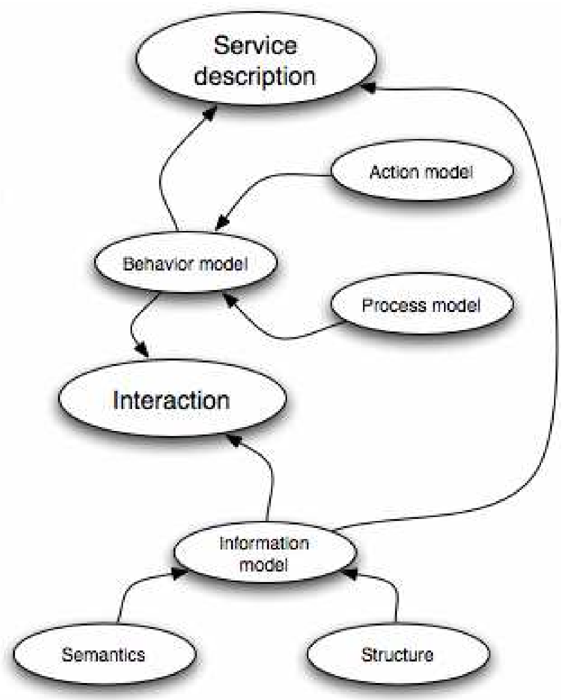

Figure 3-3 - Service Interaction (adapted from [1]1) [Concept Map]

The r*eal world effect* it is the ultimate purpose associated with the interaction with a particular service. It can be the response to a request for information or the change in the state of some shared entities between the participants in the interaction.

3.3 Service Related Concepts

SOA-RM identifies a set of concepts crucial in enabling the interaction between a service consumer and a service. These concepts are the *service description*, the *service policies and contracts* and the *execution context.*

The *service description* encompasses the information needed in order to use the service (see Figure 3-4). The purpose of the service description is to facilitate the interaction of the visibility especially if the participants are part of different ownership domains. By using the service description the service consumer should be able obtain the following items of information:

- Whether the service is reachable or not;
- Whether the service provides the function required by the requester;
- The set of policies the services operates under;
- That the service complies with the service consumer's policies;
- The means to interact with the service, including the format and content of the information to be exchanged, as well as the expected sequence of the information exchange.

As a consequence, there are several important aspects that have to be captured by the service description: the service reachability, the service functionality, the service-related policies, and the service interface.

- *Service reachability* is assured by including in the service description enough information to enable the service providers and services consumers to interact with each other. Such information could include service metadata (e.g. location, supported or required protocols), dynamic information about service (e.g. if the service is currently available), etc.
- *Service functionality* should be unambiguously captured by the service description and it should contain information about the function of a service and the real world effects that result from it being invoked. This piece of information should be expressed in a general-enough way to be understandable by service consumers while at the same time the vocabulary used should be expressive enough to capture the domain-specific details of the service functionality. Such information could include a textual description (for human consumption) or identifiers or keywords referencing machine-processable definitions.
- *Service-related policies* should be reflected by the service description in order to enable the prospective service consumer to determine if the service will act in a manner consistent with consumer's own constraints.
- The *service interface* describes the means to interact with the service. It could include specific protocols, commands and information exchange by which actions are initiated. It prescribes what information needs to be provided to the service in order to access its capabilities and interpret responses. This information is also referred as the information model of the service.

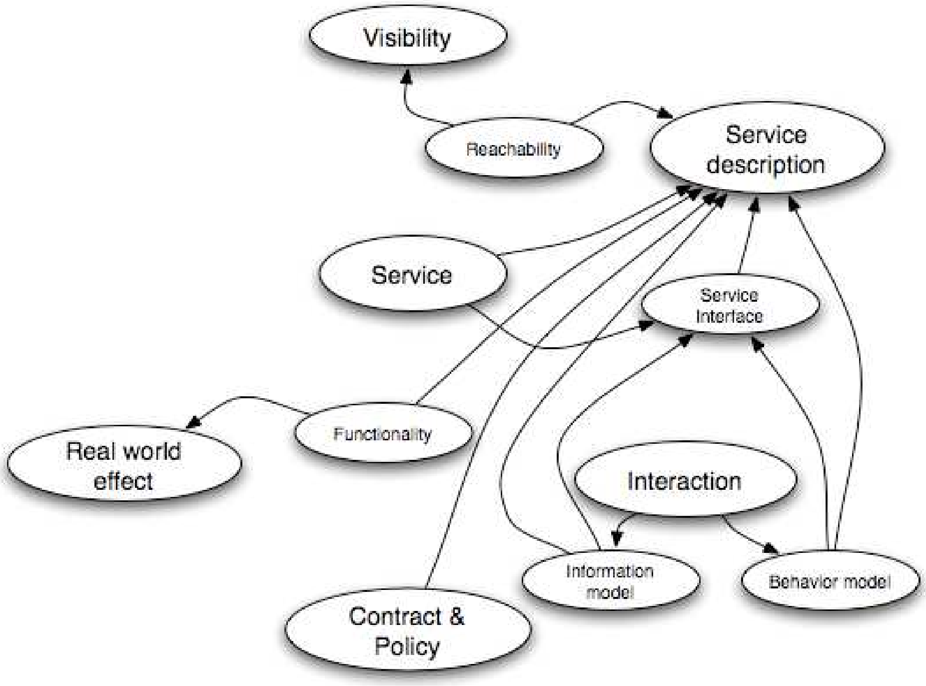

Figure 3-4 - Service Description (from [1]) [Concept Map]

The *service policy* represents the constraints or the conditions on the use, deployment or description of a service while a *contract* is a measurable assertion that governs the requirements and expectations of one or more parties. Policies potentially apply to various aspects of SOA such as security, manageability, privacy, etc. but they could also be applied to business-oriented aspects, e.g. hours of business. In their turn contracts can as well cover a wide range of aspects of services: quality of services agreements, interface and choreography agreements, commercial agreements, etc.

The *execution context* represents the set of infrastructure elements, process entities, policy assertion and agreements associated with a particular service interaction, forming a path between service consumers and service providers. The execution context is not limited to one side of the interaction but rather concerns the overall interaction, which includes the service provider, service consumer and the infrastructure in between.

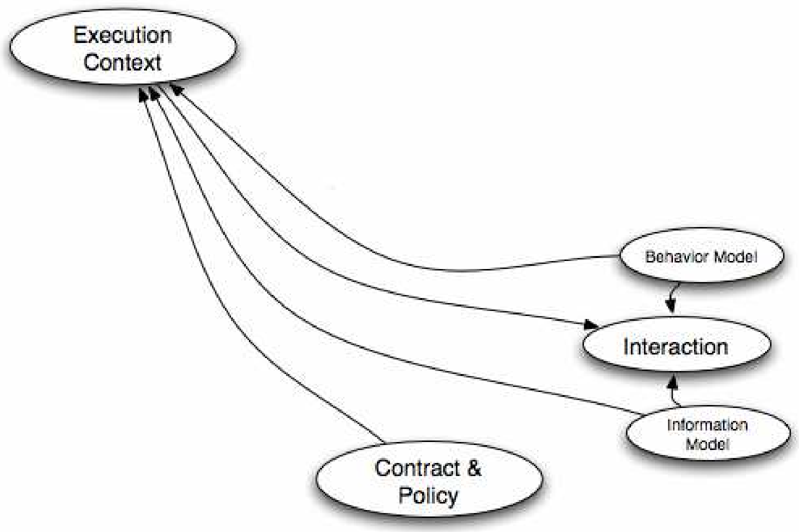

Figure 3-5 - Execution Context (adapted from [1]1) [Concept Map]

4. Reference Ontology for Semantic Service Oriented Architectures

The reference ontology for Semantic SOA formalises and extends those sections of the SOA Reference Model described above, as illustrated in Figure 2-1.

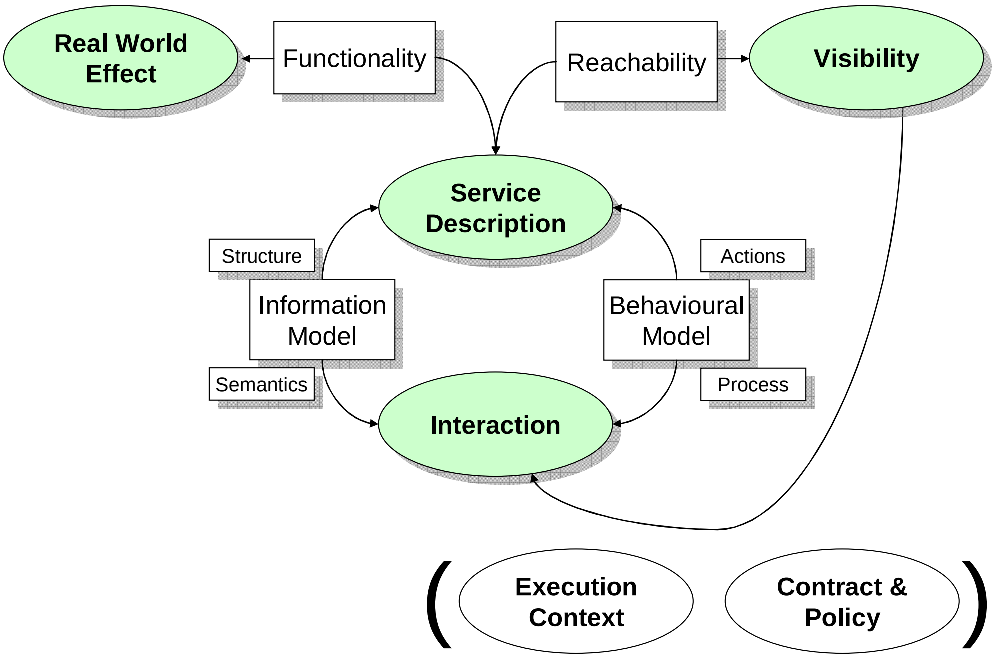

Figure 4-1 – Concepts from SOA-RM as preserved in Reference Ontology [Concept Map]

Oval shapes are used to represent the top-level elements from the SOA Reference Model, rectangles the others, and those which are shaded are the ones on which we concentrate in the Semantic SOA Reference Ontology. Although *Execution Context* and *Contracting & Policy* are all important issues for SOA, they are less mature and less ready for standardisation.

In Figure 4-2 we show how we have extended and arranged the Reference Model to enable a thorough semantic description. The most notable difference is that we replace the *Visibilty* concept with the concept of *Mediator*. *Visibility* is taken as more fundamental to the semantics-driven approach and shown underlying all concepts. Secondly, as well as a *Service Description* we introduce the first class notion of *Goal Description*, which is a top-level element like *Mediator* in our extended model. *Goal Description* is a formal description of the requirements for a service from the point of view of a consumer. In this way we can make a first class representation of the more restricted sense of *Visibility*, from the SOA RM, and *Reachability* via *Mediator*. The more general concept of *Mediation* is a grouping concept, and represented by a shaded area. In a similar way, we group the description of functionality into a concept *Capability*, and the *Behavioural Model* and *Information Model*, describing *Interaction*, into a concept *Interface*.

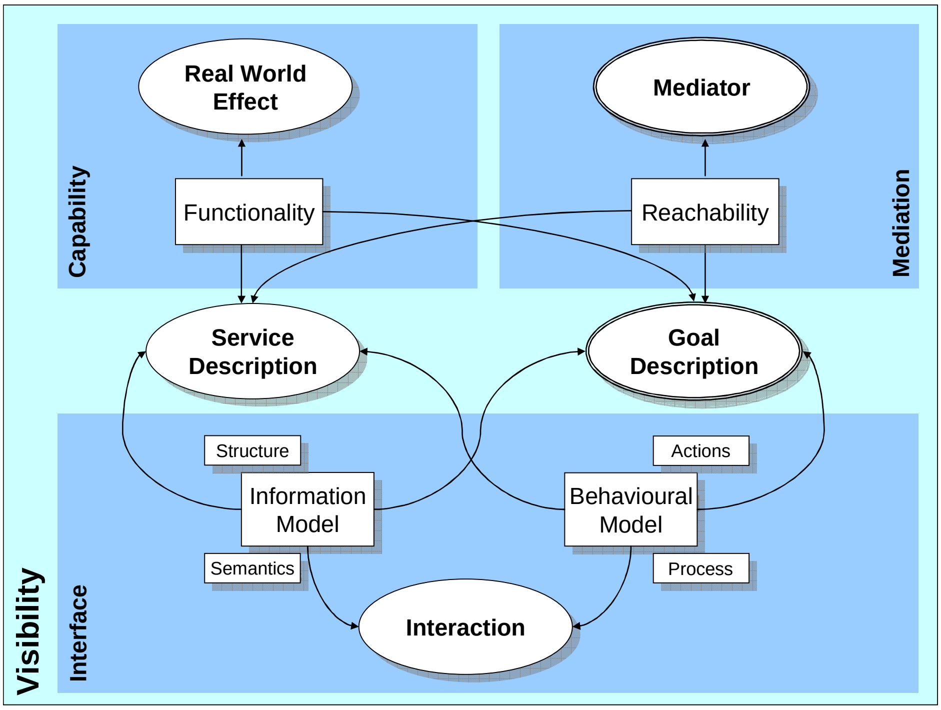

Figure 4-2 - Extension of SOA RM in the Reference Ontology [Concept Map]

The Reference Ontology is introduced in small pieces over the next sections and the complete Reference Ontology can be seen in Figure 4-10.

4.1 Visibility

The two fundamental principles of the semantics-based approach are that: all descriptions of service-oriented concepts should be made in an ontology-based formalism; that all ontology-based descriptions should be capable of being connected via mediation. For this reason we see visibility, which is the ability to access a description and thereby the service it represents, as the underlying concept of the entire approach. In the following, we introduce the concepts and requirements for a formalism to be based on ontologies.

4.1.1 Ontologies

Ontologies provide the basis for all elements in the Reference Ontology and contain Concepts, Relations, Instances, and Axioms. Service Descriptions, Goal Descriptions, and Mediators can import Ontologies in order to utilize the terminology that they provide.

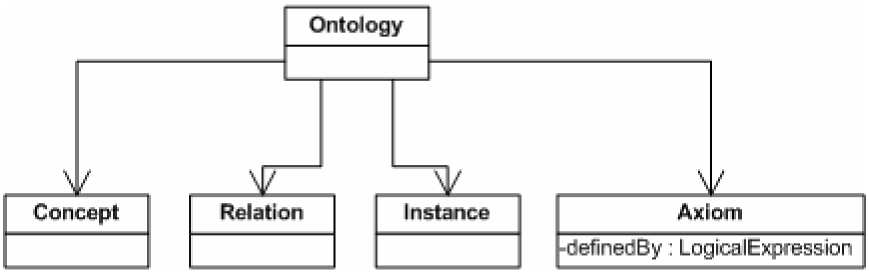

Figure 4-3 – Fundamental Modeling Elements Contained within Ontologies [UML]

4.1.2 Concepts

Concepts provide a means for describing pieces of terminology and can be related to each other via the subclass-superclass relationship. Concepts also have attributes that allow other relationships between classes to be captured.

4.1.3 Instances

Instances are identifiable or anonymous members of concepts and provide values to the attributes of those concepts. Instances may be explicitly declared as members of concepts or they may be implicit via axioms.

4.1.4 Axioms and Logical Expressions

Through the use of logical expressions, axioms define constraints that must hold over all contents of their containing ontology in order for this to be consistent. These can be used to support an explicit style of modeling, where instances and their concept memberships are declared explicitly and axioms merely constrain their allowed membership and attribute values (cf. relational database constraints), or intentionally, where concepts may be implicitly populated via axioms.

4.2 Service Description

SOA RM requires: *"The service description represents the information needed in order to use a service,"* and states that *"The service concept above emphasizes a distinction between a capability that represents some functionality created to address a need and the point of access where that capability is brought to bear in the context of SOA."* In SSOA we regard this as the critical division in the description of a service: the capability and the interface.

In the Semantic SOA Reference Ontology, these core service descriptions represent a core element in defining Semantic Web Services, which we aim to support automated reasoning over by the use of semantic technologies. Therefore, semantic descriptions are associated to all resources, thus services as well. The semantic descriptions are grounded to concrete service realizations, such that once the semantic description is known the implementation of the service can be found as well.

It is important to point out that the Semantic SOA Reference Ontology allows for both functional, including behavioral, and non-functional descriptions of the service. While the functional descriptions are formal definitions expressed in terms of ontologies, the non-functional properties are extension of the Dublin Core, and might contain human-readable descriptions as well.

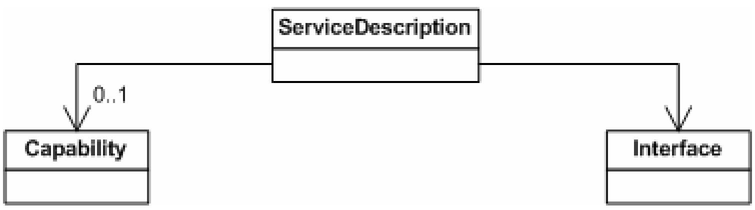

Figure 4-4 - The Top-Level Structure of a Service Description [UML]

4.3 Goal Description

SOA RM defines *awareness* as the state "whereby one party has knowledge of the existence of the other party". Semantic technologies aim to automate as much as possible the process of bringing the service requesters and the services providers in the "awareness state" and to create a dynamic infrastructure able to support all the necessary communication aspects.
Along these lines, the Semantic SOA Reference Ontology has adopted the ontological role separation principle by which the service consumers exist in a specific context, different than the one of the services to be consumed. As a consequence, the requester needs can be

independently formalized as *Goals* in accordance with their internal requirements, isolated from the peculiarities of the provider infrastructure, data or behavior models.
Nevertheless, in order to facilitate the matchmaking process between requester goals and provider services, the Reference Ontology defines a GoalDescription as being formed from the same elements as a ServiceDescription: namely a *Capability* and a set of *Interfaces*. The Capability of a GoalDescription represents the requested capability, i.e. the capability the requester desires to find and consume. The Interface of a GoalDescription describes the interfaces the requester intends to use during the communication with the matching service.

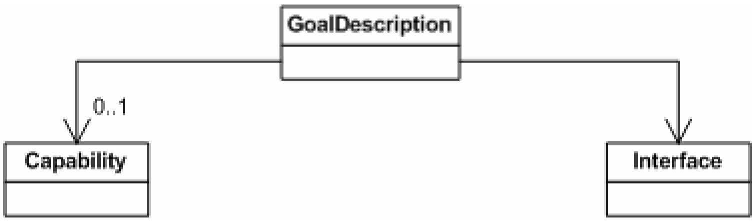

Figure 4-5 - The Top-Level Structure of a Goal Description [UML]

4.4 Capability

SOA-RM requires: *"A service description SHOULD unambiguously express the function(s) of the service and the real world effects that result from it being invoked."*

As we have seen in sections 4.2 and 4.3, a Capability is a description of the functionality provided by a service or the functionality desired by a service requester and as such can be linked to one or more Service or Goal Descriptions. Capabilities are generally used for automating the process of discovering services, by comparing the offered functionality of each provider with the desired functionality of the requester. A Capability is described in terms of conditions on the state of the world that must exist for execution of the service to be possible and conditions on the state of the world that are guaranteed to hold after execution of the service. We make a distinction between the state of the information and the state of the real world, thus these conditions can be broken down into two groups namely those related to the state of the information space (preconditions and postconditions) and those related to the to the state of the real-world (assumptions and effects). By providing these 4 elements, the Reference Ontology allows the state change that occurs in both the information space and in the real world to be effectively described.

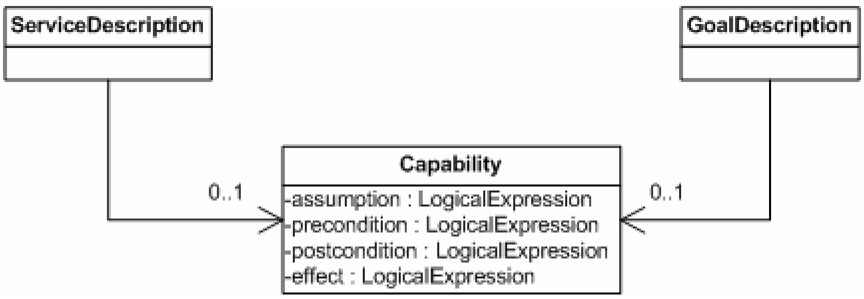

Figure 4-6 – Service and Goal Capabilities [UML]

4.4.1 Functionality

In terms of the SOA-RM the preconditions and postconditions of a service make up the description of its functionality. Preconditions describe the state of the information space prior to execution and postconditions describe the state of the information space after execution. Therefore preconditions can be used to specify what information needs to be available in order for a service to be invoked and postconditions describe what information will be generated by the service into the information space.

4.4.2 Real World Effect

Many services that can be invoked will have as the SOA-RM describes a *Real World Effect*, that is that the process of invoking a service will not only change the state of the data sources related to the service requester and service provider but also an actual change will occur to the state of the world, for example when buying a book from a book selling service the physical book will change location from the warehouse to the home of the purchaser. In the Reference Ontology we consider this real world effect by describing the state of the world prior to execution in terms of Assumptions and the state of the world after execution by Effects.

4.5 Interface

SOA-RM specifies that "*the service interface is the means for interacting with a service*". Furthermore, SOA-RM recommends that the interface consists of two parts, Information Model and Behavioral Model, and their roles will be described in the following subsections.
For the Semantic SOA reference Ontology, the service interface is also an important part of the service description. It specifies in detail how the communication with the service should take place, from two different perspectives:

- Service requester perspective - the information that is needed for service execution by the service requester, specified as *Choreography*;
- Communication with other services – information on how the service can coordinate the cooperation between other services in order to fulfill its functionality, specified as the *Orchestration*.

The Service Interface encapsulates all the information from the Information and Behavioral Model, providing a clear and concise description of the information and communication pattern needed for interacting with the service from the invoker's perspective.

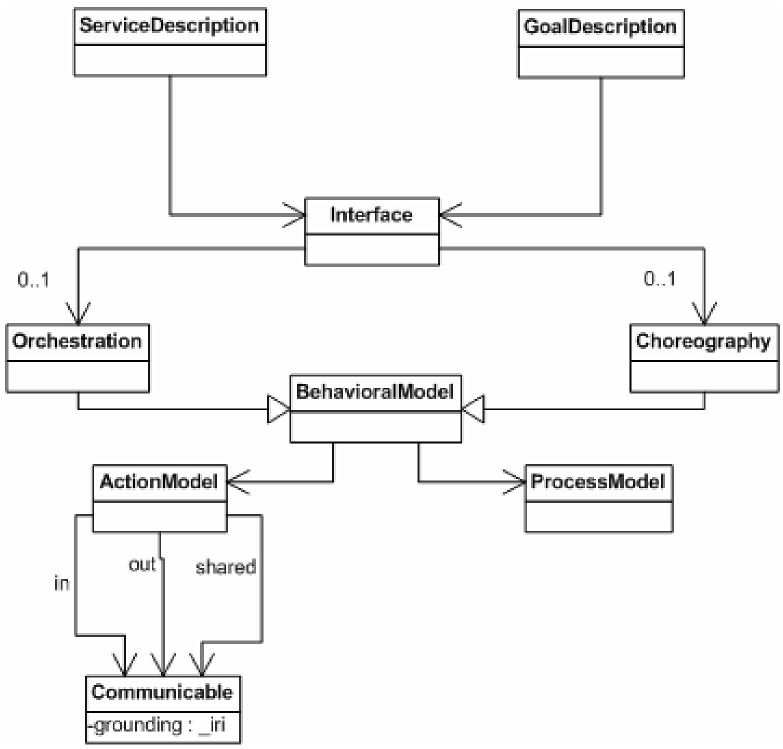

Figure 4-7 - The Structure of an Interface [UML]

4.5.1 Information Model

"*The information model of a service is a characterization of the information that may be exchanged with the service*". As previously described, for Semantic SOA this information is provided by the domain ontology of the service; this ontology specifies all the information needed for the service execution and for its communication with other services or with the requestors.

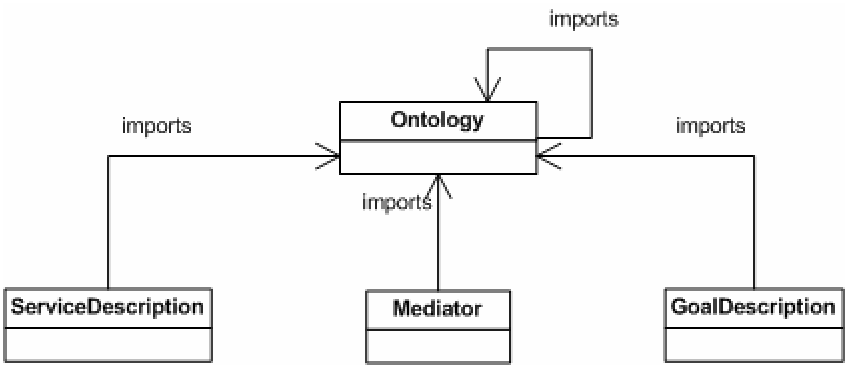

Figure 4-8 Ontologies as Information Model [UML]

4.5.1.1 Structure

The information model of a service has to have a given structure, a standard form of the required information in order to ensure the successful invocation of the service. This structure is given by the domain ontology, which prescribes the format of the information needed or provided by the service.

The information model is described (like any other entity presented in this document) in terms of ontologies.

4.5.1.2 Semantics

The parties involved in a communication need to have a common understanding of the semantic of the exchanged messages. When the parties use ontologies for describing their information model, this common understanding implies either a previous agreement regarding what ontologies are used, or the existence of a mediator for solving any heterogeneity problems. This will ensure a high degree of automation for the communication.

4.5.2 Behavioral Model

The SOA RM defines the Behavioral Model as "*knowledge of the actions invoked against the service and the process or temporal aspects of interacting with the service*". For Semantic SOA this knowledge is encapsulated by the definition of what information needs to be exchanged during the communication, the concepts and relations of an ontology being marked to support a particular role (or mode). Furthermore, the order in which the messages are exchanged needs to be unambiguously specified.

4.5.2.1 Action Model

For specifying what information needs to be exchanged during the communication the concepts and relations of an ontology are marked to support a particular role (or mode). There are five modes defined in the state signature:

- *static* - meaning that the extension of the concept cannot be changed;
- *in* - meaning that the extension of the concept or relation can only be changed by the environment and read by the service;

- *out* - meaning that the extension of the concept or relation can only be changed by the service and read by the environment;
- *shared* - meaning that the extension of the concept or relation can be changed and read by the service and the environment;
- *controlled* - meaning that the extension of the concept is changed and read only by the service.

4.5.2.2 Process Model

For using the modes defined in the state signature a grounding mechanism needs to be provided for allowing the environment (i.e. the communication partner) to read or to write information in the services ontology. For each mode except static and controlled, a different grounding mechanism needs to be provided as follows:

- *in* - a **grounding** mechanism for the in items, that implements *write* access for the environment, must be provided;
- *out* - a **grounding** mechanism for the out items, that implements *read* access for the environment, must be provided;
- *shared* - a **grounding** mechanism for the shared items, that implements *read/write* access for the environment and the service, must be provided.

For the static and controlled items a grounding mechanism is not needed, as these items can either be changed only by the service or remain unchanged for the duration of the communication.

Furthermore, a set of rules are needed for defining the order in which the messages can be exchanged. These rules could be specified using the Abstract State Machine methodology, each rule evaluating some conditions on the current state of the service, and prescribing what activities should be performed if the conditions are fulfilled.

4.6 Mediation

SOA RM defines Visibility as "*the relationship between service consumers and providers that is satisfied when they are able to interact with each other*". Visibility itself subsists in the publication of Service and Goal Descriptions, but a prerequisite of Visibility is represented by Reachability, and when two entities are aware of each other and willing to interact in order to fulfill a need, heterogeneity can be a barrier that prevents this prerequisite to be fulfilled. Given two heterogeneous entities, mediation enables Reachability by resolving mismatches between them.

A mediator is described in terms of the entities it is able to connect and states how it will resolve mismatches. Ontology to Ontology mediators (OO-Mediators) connect ontologies and resolve terminological and representational mismatches, Web service to Web service mediators (WW-Mediators) connect Web service descriptions resolving mismatches between the representation of their functionality and/or in the means by which they are accessed (i.e., between their capabilities and/or interfaces), Goal to Goal mediators (GG-Mediators) connect Goal descriptions resolving mismatches in the requirements of the service requestor, again either in capability or interface terms, and Web service to Goal (WG-Mediators) connect Web service descriptions and Goal descriptions, mediating between the consumer's and provider's viewpoint of the functionality and/or its access. By using a Mediation Service, a Mediator explicitly describes the link to a concrete solution to perform mediation. This mechanism allows Mediators to be used to describe pieces of functionality offered by complex services that are able to perform concrete mediation scenarios. A mediation service can either be a Goal or a Service Description. The former links to a Goal that is to be used in the discovery process to find a Service offering the functionality described by the Mediator, while the latter directly links to a Service that is able to offer the functionality described by the Mediator.

By publishing the description of the Mediator and all its needed Ontologies, Goal and Service Descriptions, the requirements for Visibility are met, thus allowing a Goal to interact with the Service.

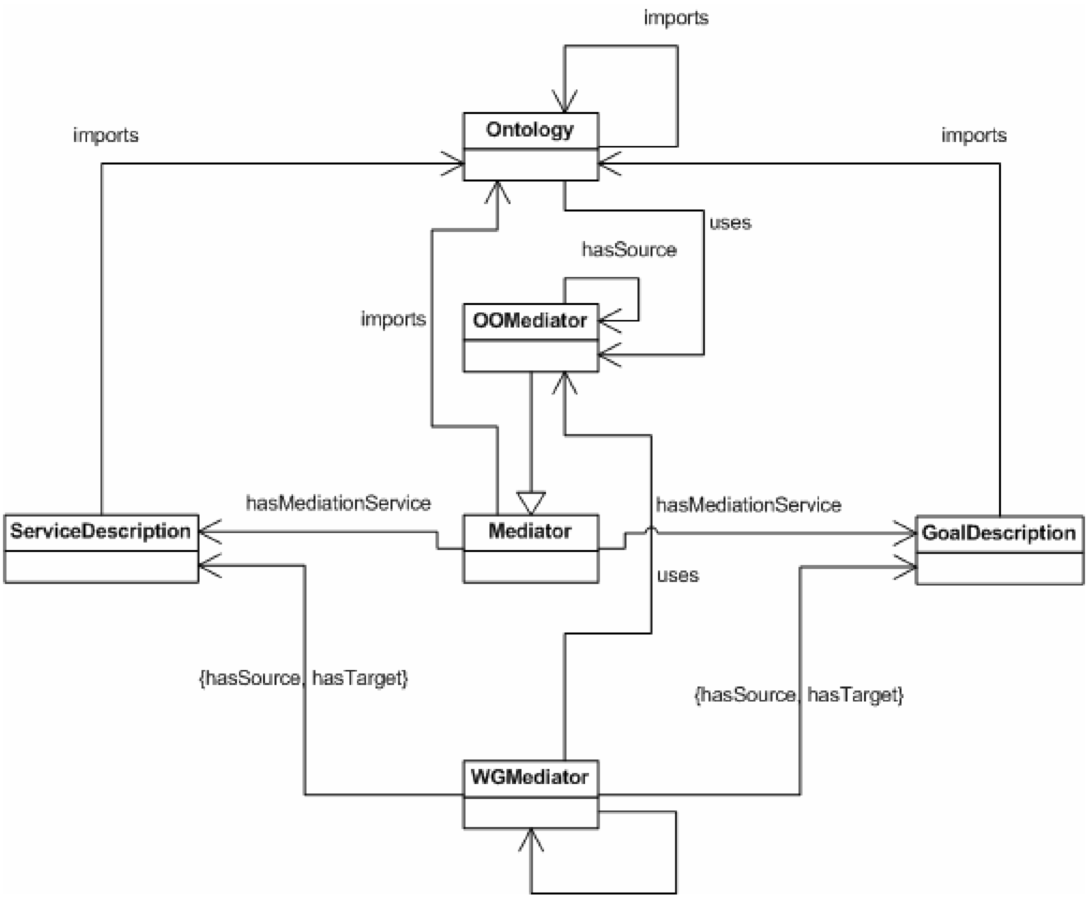

Figure 4-9 – Mediators and their Connection of other RO Concepts [UML]

4.7 Complete Reference Ontology

In Figure 4-10 shows complete UML diagram for the Reference Ontology, which combines all the information from Figure 4-3 to Figure 4-8. The formalisation of this ontology in WSML is presented in Appendix B.

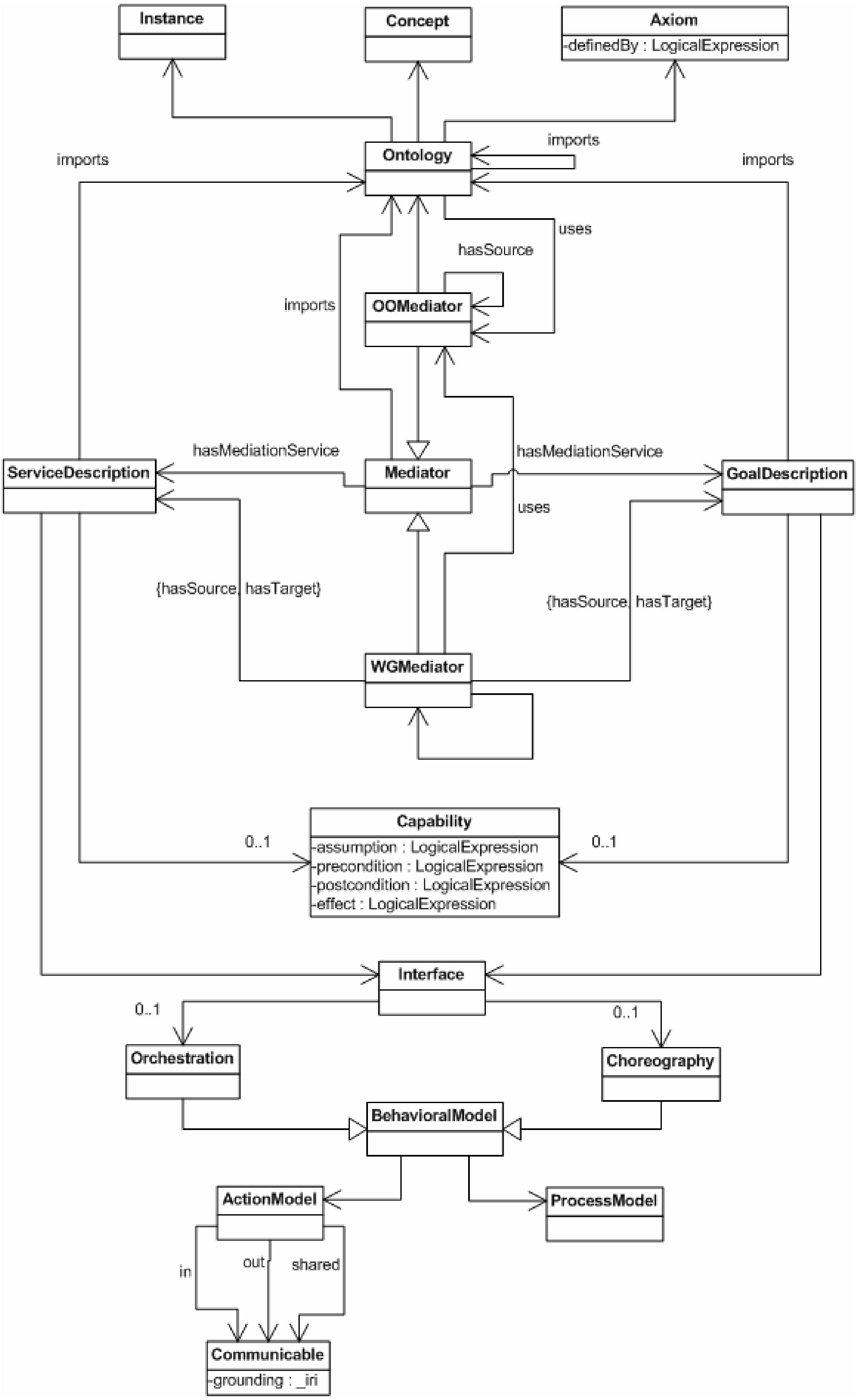

Figure 4-10 - The Complete Reference Ontology [UML]

5. Conclusions

This deliverable presents the charter of OASIS SEE (Semantic Execution Environment) Technical Committee and its current state of specification called as Reference Ontology for Semantic Service Oriented Architectures. The reference ontology for semantic service oriented architectures is an abstract framework for understanding significant entities and relationships between them within a Semantically enabled Service Oriented Environment. This Reference Ontology has been built based on the OASIS Reference Model for Service Oriented Architecture (SOA-RM) and combines it with the key concepts of semantics that are relevant for Semantically-enabling Service Oriented Architectures. The reference model is not directly tied to any standards, technologies or other concrete implementation details. It does seek to provide a common understanding that can be used unambiguously across and between different implementations. Just as the SOA-RM, this reference ontology focuses on the field of software architecture. The concepts and relationships described may apply to other "service" environments; however, this specification makes no attempt to completely account for use outside of the software domain. The OASIS SEE TC can be found at http://www.oasis-open.org/committees/semantic-ex/ and the original specifications are available at [11].

This specification helps in initializing the next deliverable D11.1.2 (Initialise OASIS WG on SEE With SOA4All Extensions) which identifies necessary extensions to the specifications based on SOA4ALL and provides a draft charter for the new OASIS Technical Committee. These extensions include Reference Ontology based on Semantic Annotations to WSDL (SAWSDL), WSMO-Lite as well as User Context.

6. References

1. C. M. MacKenzie, K. Laskey, F. McCabe, P. F. Brown, R. Metz (eds.): Reference Model for Service Oriented Architecture 1.0, OASIS SOA-RM Technical Committee Specification, 2 August, 2006, available at: http://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf
2. A. Haller, E. Cimpian, A. Mocan, E. Oren, C. Bussler: WSMX: A Semantic Service-Oreinted Architecture. In Proceedings of the International Conference on Web Services (ICWS 2005), Orlando, Florida.
3. J. Domingue, L. Cabral, S. Galizia, V. Tanasescu, A. Gugliotta, B. Norton, and C. Pedrinaci. IRS-III: A broker-based approach to semantic Web services. Web Semantics: Science, Services and Agents on the World Wide Web, 6(2):109–132, 2008.
4. World Wide Web, 6(2):109–132, 2008.K. Verma, K. Gomadam, A.P. Sheth, J.A. Miller, Z. Wu: The METEOR-S Approach for Configuring and Executing dynamic Web Processes. LSDIS Technical Report, 24 June, 2005, available at: http://lsdis.cs.uga.edu/projects/meteor-s/techRep6-24-05.pdf
5. J. de Bruijn, C. Bussler, J. Domingue, D. Fensel, M. Hepp, M. Kifer, B. König-Ries, J. Kopecky, R. Lara, E. Oren, A. Polleres, J. Scicluna, M. Stollberg: The Web Service Modeling Ontology WSMO.. Forschungsinstitut at the University of Innsbruck Technical Report, 16 February 2007, available at: http://www.wsmo.org/TR/d2/v1.4/
6. S. Bradner, RFC2119 – Keywords for use in RFCs to indicate Requirement Levels, http://www.rfc.net/rfc2119.html
7. H. Lausen and J. de Bruijn, A. Polleres and D. Fensel, WSML – A Language Framework for Semantic Web Services, Proceedings of the W3C workshop on Rule Languages for Interoperability, April 2005.
8. J. Farrell, H. Lausen: Semantic Annotations for WSDL and XML Schema. W3C Recommendation, 28 August 2007, available at: http://www.w3.org/TR/sawsdl/
9. D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, B. Parsia, T. Payne, E. Sirinin, N. Srinivasan, K. Sycara: OWL-S: Semantic Markup for Web Services. DARPA DAML Program Technical Report, available at: http://www.ai.sri.com/daml/services/owl-s/1.2/overview/
10. S. Battle, A. Bernstein, H. Boley, B. Grosof, G. Kruninger, R. Hull, M. Kifer, D. Martin, S. McIlraith, D. McGuinness, J. Su, S. Tabet: Semantic Web Services Ontology (SWSO). DARPA DAML Program Technical Report, 9 May 2005, available at: http://www.daml.org/services/swsf/1.0/swso/
11. M. Kerrigan, B. Norton, A. Mocan (Editors): “Reference Ontology for Semantic Service Oriented Architectures”, OASIS SEE (Semantic Execution Environment) Technical Committee specifications, Release Candidate, June 2008. Available at http://www.oasis-open.org/apps/org/workgroup/semantic-ex/document.php?document_id=27923

Annex A. Glossary

This section extends the terminology described in Glossary (Appendix A) of the "Reference Model for Service Oriented Architecture, Public Review Draft 1.0" and introduces any new terms needed by the Semantic SOA Reference. The two glossaries are intended to be used together, therefore terms from the other glossary will not be repeated here.

Goal-to-Goal Mediator (GG-Mediator)

Connects Goal descriptions resolving mismatches in the requirements of the service requestor in terms of the requested functionality and/or in the means by which they wish to access the service

Internet Reasoning Service 3 (IRS III)

A framework and infrastructure that supports the creation of Semantic Web Services according to the WSMO ontology.

Managing End-To-End OpeRations for Semantic Web Services and Processes (METEOR-S)

Project that aims to extend Web service –related standards with Semantic Web technologies to achieve greater dynamism and scalability for Service-oriented Architectures.

Object-oriented Design (OOD)

Object-oriented design is part of OO methodology and it forces programmers to think in terms of objects, rather than procedures, when they plan their code.

Ontology-to-Ontology Mediator (OO-Mediator)

Connects ontology and resolves terminology as well as representation or protocol mismatches.

Resource Description Framework (RDF)

Resource Description Framework (RDF) is a family of World Wide Web Consortium (W3C) specifications originally designed as a metadata model but which has come to be used as a general method of modeling information, through a variety of syntax formats.

Rule Interchange Format (RIF)

The Rule Interchange Format (RIF) is a W3C recommendation-track effort to develop a format for interchange of rules in rule-based systems on the semantic web. The goal is to create an interchange format for different rule languages and inference engines.

Semantic Annotations for WSDL (SAWSDL)

The Semantic Annotations for WSDL and XML Schema (SAWSDL) W3C Recommendation defines mechanisms using which semantic annotations can be added to WSDL components.

Semantic Execution Environment (SEE)

Execution environment capable to consume semantic messages, discover semantically described Web services, and invoke and compose them for the end-user benefit.

Semantic Web

The **Semantic Web** is an evolving extension of the World Wide Web in which the semantics of information and services on the web is defined, making it possible for the web to understand and satisfy the requests of people and machines to use the web content. [cite: Wikipedia]

Semantic Service Oriented Architecture (SSOA)

A **Semantic Service Oriented Architecture** (**SSOA**) is a computer architecture that allows for scalable and controlled Enterprise Application Integration solutions. SSOA describes a sophisticated approach to enterprise scale IT infrastructure. It leverages rich, machine-interpretable descriptions of data, services, and processes to enable software agents to autonomously interact to perform critical mission functions. [cite: Wikipedia]

Semantic Web Services (SWS)

Semantic Web Services are self-contained, self-describing, semantically marked-up software resources that can be published, discovered, composed and executed across the Web in a task driven semi-automatic way. Semantic Web Services can be defined as the dynamic part of the semantic web.

Semantic Web Service Ontology (SWSO)

An ontology for Semantic Web Services, which is expressed in two forms: FLOWS, the First-order Logic Ontology for Web services; and ROWS, the Rules Ontology for Web services, produced by a systematic translation of FLOWS axioms into the SWSL-Rules language.

Service-oriented Architecture (SOA)

Service Oriented Architecture (SOA) is a paradigm for organizing and utilizing distributed 128 capabilities that may be under the control of different ownership domains.

Unified Modeling Language (UML)

The Unified Modeling Language (UML) is a standardized visual specification language for object modeling. UML is a general-purpose modeling language that includes a graphical notation used to create an abstract model of a system, referred to as a UML model.

Web Ontology Language for Services (OWL-S)

OWL-S is an ontology built on top of Web Ontology Language (OWL) by the DARPA DAML program. It replaces the former DAML-S ontology.

Web Service Description Language (WSDL)

The Web Services Description Language is an XML-based language that provides a model for describing Web services.

Web service-to-Goal Mediator (WG-Mediator)

Connects Web service descriptions and Goal descriptions, mediating between the consumer's and provider's viewpoint of the functionality and/or its access

Web service-to-Web Service Mediator (WW-Mediator)

Connects Web service descriptions resolving mismatches between the representation of their functionality and/or in the means by which they are accessed.

Web Service Modeling eXecution environment (WSMX)

An execution environment for business application integration where enhanced Web services are integrated for various business applications. It is the reference implementation of WSMO (Web Service Modeling Ontology).

Web Service Modeling Language (WSML)

A language that formalizes the Web Service Modeling Ontology (WSMO).

Web Service Modeling Ontology (WSMO)

WSMO or Web Service Modeling Ontology is an ontology currently developed to support the deployment and interoperability of Semantic Web Services.

Annex B. WSML Formalization of Reference Ontology

```
wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-flight"
namespace { _"http://www.semantic-soa.org/ReferenceOntology#",
            dc _"http://purl.org/dc/elements/1.1/" }

ontology _"http://www.semantic-soa.org/ReferenceOntology#"

concept Ontology
  imports ofType Ontology
  hasConcept ofType Concept
  hasInstance ofType Instance
  hasAxion ofType Axiom
  uses ofType OOMediator

concept Concept
  hasInstance ofType Instance

concept Instance

concept Axiom
  hasLogicalExpression ofType _"http://www.wsmo.org/wsml/wsml-
syntax#logicalExpression"

concept ServiceDescription
  imports ofType Ontology
  offersCapability ofType (0 1) Capability
  hasInterface ofType Interface

concept GoalDescription
  imports ofType Ontology
  requiresCapability ofType (0 1) Capability
  hasInterface ofType Interface

concept Capability
  hasPrecondition ofType _"http://www.wsmo.org/wsml/wsml-
syntax#logicalExpression"
  hasAssumption ofType _"http://www.wsmo.org/wsml/wsml-
syntax#logicalExpression"
  hasPostcondition ofType _"http://www.wsmo.org/wsml/wsml-
syntax#logicalExpression"
  hasEffect ofType _"http://www.wsmo.org/wsml/wsml-
syntax#logicalExpression"

concept Interface
  hasChoreography ofType (0 1) Choreography
  hasOrchestration ofType (0 1) Orchestration

concept Choreography subConceptOf BehaviourModel

concept Orchestration subConceptOf BehaviourModel


concept BehaviourModel
  hasActionModel ofType (1) ActionModel
  hasProcessModel ofType (0 1) ProcessModel

concept ActionModel
```

```
   hasInAction ofType (1) Communicable
   hasOutAction ofType (1) Communicable
   hasSharedAction ofType (1) Communicable

concept Communicable
   grounding ofType (0 1) _iri

concept MediationService
   nfp
     dc#relation hasValue { aServiceIsAPotentialMediationService,
                            aGoalIsAPotentialMediationService}
   endnfp

axiom aServiceIsAPotentialMediationService definedBy
   ?m memberOf ServiceDescription implies
   ?m memberOf MediationService.

axiom aGoalIsAPotentialMediationService definedBy
   ?m memberOf GoalDescription implies
   ?m memberOf MediationService.

concept Mediator
   imports ofType Ontology
   hasMediationService ofType (0 1) MediationService


concept WGMediator subConceptOf Mediator
   hasSource ofType (1) WGMediatorSource
   hasTarget ofType (1) WGMediatorTarget
   RO#usesMediator ofType (1) OOMediator

concept WGMediatorSource
   nfp
     dc#relation hasValue { aServiceIsAPotentialWGMediatorSource,
                            aGoalIsAPotentialWGMediatorSource,
                            aWGMediatorIsAPotentialWGMediatorSource}
   endnfp

axiom aServiceIsAPotentialWGMediatorSource definedBy
   ?x memberOf ServiceDescription
   implies
   ?x memberOf WGMediatorSource.

axiom aGoalIsAPotentialWGMediatorSource definedBy
   ?x memberOf GoalDescription
   implies
   ?x memberOf WGMediatorSource.

axiom aWGMediatorIsAPotentialWGMediatorSource definedBy
   ?x memberOf WGMediator
   implies
   ?x memberOf WGMediatorSource.
concept WGMediatorTarget
   nfp
     dc#relation hasValue { aServiceIsAPotentialWGMediatorTarget,
                            aGoalIsAPotentialWGMediatorTarget,
                            aWGMediatorIsAPotentialWGMediatorTarget}
   endnfp
```

```
axiom aServiceIsAPotentialWGMediatorTarget definedBy
  ?x memberOf ServiceDescription
  implies
  ?x memberOf WGMediatorTarget.

axiom aGoalIsAPotentialWGMediatorTarget definedBy
  ?x memberOf GoalDescription
  implies
  ?x memberOf WGMediatorTarget.

axiom aWGMediatorIsAPotentialWGMediatorTarget definedBy
  ?x memberOf WGMediator
  implies
  ?x memberOf WGMediatorTarget.

concept OOMediator subConceptOf Mediator
  hasSource ofType OOMediatorSource

concept OOMediatorSource
  nfp
    dc#relation hasValue { anOntologyIsAPotentialOOMediatorSource,

anOOMediatorIsAPotentialOOMediatorSource}
  endnfp

axiom anOntologyIsAPotentialOOMediatorSource definedBy
  ?x memberOf Ontology
  implies
  ?x memberOf OOMediatorSource.

axiom anOOMediatorIsAPotentialOOMediatorSource definedBy
  ?x memberOf OOMediator
  implies
  ?x memberOf OOMediatorSource.
```

Listing 1: Semantic SOA Reference Ontology Expressed in WSML

Annex C. Acknowledgements

Acknowledgements to the chairs and all the members of OASIS SEE TC, editors and authors of this specification, as well as all other committee members as listed below:

Chairs:

John Domingue, Open University, <j.b.domingue@open.ac.uk>

Michal Zaremba, STI, <michal.zaremba@sti2.at>

Editors of this specification:

Mick Kerrigan, STI, <mick.kerrigan@sti2.at>

Barry Norton, Open University, <b.j.norton@open.ac.uk>

Adrian Mocan, STI, <adrian.mocan@sti2.at>

Contributing Authors to this specification:

Alessio Carenini, CEFRIEL, <alessio.carenini@cefriel.it>

Emilia Cimpian, STI, <emilia.cimpian@sti2.at>

Marc Haines, Individual <mhaines@uwm.edu>

James Scicluna, STI, <james.scicluna@sti2.at>

Michal Zaremba, STI, <michal.zaremba@sti2.at>

List of all members:

Name	Organization	Role
--------	-----------------	-------
Peter Matthews	CA	Member
Alessio Carenini	CEFRIEL	Voting Member
Carl Mattocks	CheckMi	Member
Marc Haines*	Individual	Voting Member
Dugki Min*	Individual	Member
Robert Vitello	New York State Department of Labor	Member
Eunju Kim	NIA (National Information Society Agency)	Member
Elmar Dorner	SAP AG*	Member
Emilia Cimpian	STI Innsbruck	Voting Member
Mick Kerrigan	STI Innsbruck	Secretary
Srdjan Komazec	STI Innsbruck	Voting Member
Adrian Mocan	STI Innsbruck	Voting Member

Omair Shafiq	STI Innsbruck	Member
Michal Zaremba	STI Innsbruck	Chair
John Domingue	Open University	Chair
Barry Norton	Open University	Voting Member
Carlos Pedrinaci	Open University	Voting Member
Hong-Gee Kim	Seoul National University	Member
Paul Tazbaz	Wells Fargo	Member

Furthermore, acknowledgements to Carlos Pedrinaci from Open University and Jean-Pierre Lorre from EBM-WebSourcing for reviewing this deliverable.