Semantic Transformation of Web Services

David Bell, Sergio de Cesare, and Mark Lycett

Brunel University,
Uxbridge, Middlesex
UB8 3PH, United Kingdom
{david.bell, sergio.decesare, mark.lycett}@brunel.ac.uk

Abstract. Web services have become the predominant paradigm for the development of distributed software systems. Web services provide the means to modularize software in a way that functionality can be described, discovered and deployed in a platform independent manner over a network (e.g., intranets, extranets and the Internet). The representation of web services by current industrial practice is predominantly syntactic in nature lacking the fundamental semantic underpinnings required to fulfill the goals of the emerging Semantic Web. This paper proposes a framework aimed at (1) modeling the semantics of syntactically defined web services through a process of interpretation, (2) scoping the derived concepts within domain ontologies, and (3) harmonizing the semantic web services with the domain ontologies. The framework was validated through its application to web services developed for a large financial system. The worked example presented in this paper is extracted from the semantic modeling of these financial web services.

1 Introduction

Web services have become the predominant paradigm for the development of distributed software systems. Web services provide the means to modularize software in a way that functionality can be described, discovered and invoked in a platform independent manner over a network (e.g., intranets, extranets and the Internet). Notwithstanding the architectural advantages of such a paradigm, the representation of web services by current industrial practice is predominantly syntactic in nature lacking the fundamental semantic underpinnings required to fulfill the goals of the emerging Semantic Web.

Within a Semantic Web context web services require precise semantic representations, normally achieved through the use of ontologies, in order to provide the necessary relationships with domain models and ultimately mappings to the real world objects that such models refer to. As a consequence, syntactic web services already described in languages like the Web Services Description Language (WSDL) require semantic transformations and subsequent integration with domain ontologies [1].

The de facto standard languages for describing, publishing and invoking web services are currently WSDL, Universal Description, Discovery and Integration (UDDI), and Simple Object Access Protocol (SOAP), respectively. Although such languages

provide the technical means for achieving cross-platform distributed software deployment, they are not sufficient to achieve a level of semantic expression necessary for machines to automatically relate web services to other resources and in doing so discover the services required for composing and choreographing the intended behavior [2]. In relation to the Semantic Web and its goals, syntactically defined web services represent legacy applications which need to be conceptually reengineered in order to extract the semantics (i.e., precise meaning) of the intended behavior and the underlying domain concepts such behavior utilizes. This conceptual reengineering can be referred to as semantic transformation. The ultimate result of semantic transformation is a set of ontological models which would allow an agent (typically a software agent) to navigate through a semantic network which contains references to all types of web resources including available services.

This paper presents a framework aimed at (1) modeling the semantics of syntactically defined web services through a process of interpretation, (2) scoping the derived concepts within domain ontologies, and (3) harmonizing the semantic web services with the domain ontologies. The framework was validated through its application to web services developed for a large financial system. The worked example presented in this paper is extracted from the semantic modeling of these financial web services.

2 Lack of Semantics in Web Services

Web services are a fundamental part of the emerging Semantic Web. Web services are self-contained and self-describing modular Web applications that can be published, located, and invoked across a network (IBM) and capable of supporting interoperable machine-to-machine interaction (World-Wide Web Consortium). A web service has an interface described in a machine-processable format. It is through this interface that a web service communicates with other software applications. Although the tools and methods required to develop web services have matured over recent years, there exists limited support in the area of the semantic representation of web services and their integration with other web resources [3]. Such a need is motivated by the machine-processable nature of all Semantic Web resources. In order for web services to be automatically discovered, selected and composed, it is necessary for a software agent to autonomously navigate the Semantic Web in search of services satisfying specific criteria. Such criteria are generally defined in terms of what a service provides (i.e., output) and what a service requires (i.e., input). For a software agent to recognize such elements, both inputs and outputs should preferably be expressed or typed with reference to ontological models which semantically map to the resources (including domain objects and web services) of the Semantic Web. With such models all web resources would be represented through interrelated web ontologies, thus facilitating the integration of web services with the whole of the Semantic Web.

Currently the scenario just described is not implemented. Web services are primarily adopted in industry as a means to develop architecturally sound information systems. Web services as they are typically developed today do not support the necessary semantic precision and “machine-processability” for software agents to automatically

navigate through the future Semantic Web and pinpoint those services which can suit specific requirements. As this research shows, web services developed in industry today are mostly syntactic in nature. This is simply demonstrated by the elementary typing of the services' input and output parameters. Such parameters are normally typed in relation to traditional programming language types such as strings, integers and floats. Such types do not map directly to web resources such as books, flights, bank accounts and people. As such a software agent searching for a flight booking service would unlikely find a service with an input parameter typed by an ontologically represented 'flight' class, but would most probably find many services with generic string input parameters. Such syntactic representations work in a semantically poor environment based on WSDL, UDDI and SOAP, however they would not be able to scale up to the requirements of the Semantic Web as described above.

More recently serious and important attempts have been undertaken to define languages for semantically representing web services. Initiatives such as OWL-S and WSDL-S are an important step forward. OWL-S, for example, defines a high level ontology for web services. OWL-S is based on OWL (Web Ontology Language) and as such provides the basis for the possible semantic integration between web services and other web resources. In the presence, however, of a vast amount of syntactic web services developed to date, service ontologies like OWL-S are necessary but not sufficient to resolve the problem of how to integrate these technical services within the emerging Semantic Web. In addition, much of the current research assumes the existence of ontology for composition or discovery [4]. A framework for systematically transforming syntactic web services into semantic web services is required to support these assumptions. The remainder of this paper will present such a framework and exemplify it within the context of a financial services example.

3 Framework

3.1 Underlying philosophy and concepts

A framework has been developed for deriving semantic content from syntactic web services and representing such semantics in ontological models. The framework is based on the principles of content sophistication described by Partridge [5] and Daga et al. [6]. Content sophistication represents a process for improving the semantic contents of legacy systems along several dimensions and representing such improvements in technology-agnostic conceptual models. The framework proposed in this paper provides the basis for interpreting the semantics of syntactic web services in a similar fashion. In fact in order to achieve the claimed benefits of the Semantic Web, it is necessary for web services to be semantically well defined and related to other types of web resources [7]. In this sense it is not exaggerated to state that, for the Semantic Web, syntactic descriptions of services developed today represent the 'legacy of the future'.

At the heart of the framework is the adoption of ontology to drive the derivation of semantic content from syntactic web services. From a philosophical perspective ontology can be defined as a set of things whose existence is acknowledged by a particular theory or system [8]. Such 'things' include both types (such as the class of *Bank Accounts*) and individual elements (such as *John Smith's Bank Account*). The adoption of such a definition is important because, when compared with more computationally orientated definitions of ontology (for example, Gruber [9] states that "an ontology is a specification of a conceptualization"), there is an explicit reference to a system's ontic commitment (i.e., things whose existence is acknowledged or recognized). This leads to representations that are more closely mapped to real world objects. Such mapping or reference [10] is essential to ontological modeling. The meaning of a sign, used, for example, to denote a service or a parameter, becomes well understood when it is possible to identify the thing(s) the sign refers to.

The focus of the framework presented in this section is the discovery of the semantics underlying a service description in its fundamental parts (mainly name and parameters). This process of concept discovery, called interpretation, identifies those real world objects that individual service parts ontologically commit to (or refer to). The semantics that are unraveled in this way are then represented in technology-agnostic domain and service ontology models.

The framework addresses the following objectives: (1) Derivation of semantics from previously developed web service syntactic descriptions; (2) Representation of the derived semantics in ontological models; and (3) Integration of models of semantic web services with models of other web resources. These objectives define the scope of the framework. A process was defined in order to achieve the objectives listed above. It is beyond the scope of this paper to describe in detail how the ontological models derived from the framework can be used by a semantic web search facility to discover and compose services.

3.2 Framework Process and Artifacts

The process, which drives the discovery and representation of semantic content from technical web services, is summarized in Table 1. The process is iterative and its outcome (defined in terms of ontological models) outlives one specific reengineering project. The framework's ongoing mission is to develop (within and across domains) interlinked ontological models for the Semantic Web. These models represent simultaneously all types of resources including service offerings. The process consists of three main activities: service interpretation, concept scoping and harmonization. These activities have been adapted from the Content Sophistication process presented by Daga et al. [6]. As a whole the process takes in technical service descriptions and produces ontological representations. The individual process activities also require and produce artifacts which progressively lead to achieving the ontological models.

Table 1. Process for deriving semantic content from web services.

Activities	Description	Input Artifacts	Output Artifacts
Service interpretation	A service description is broken down into its fundamental parts (e.g., name, input and output parameters). Each part is interpreted in order to represent its ontic commitment.	▪ Web service descriptions (e.g., WSDL code)	▪ Individual service ontic commitment models
Concept scoping	The concepts represented in the service ontic commitment models are either mapped to pre-existing ontologies or assigned to newly developed ones.	▪ Service ontic commitment models ▪ Domain ontologies	▪ Objects incorporated or mapped to ontological domain models
Harmonization	Services are represented within ontological models and related to other domain objects.	▪ Service ontic commitment models ▪ Domain ontologies	▪ Extended or specialized domain ontology ▪ Service ontology

3.3 Interpretation

The first activity is Service Interpretation. This activity works on service descriptions with limited or no explicit semantic underpinning. The descriptions are normally represented in the form of a service name with input and output parameters. The parameters themselves are named and typed. For example, in WSDL a typical service description can be found as a combination of service signatures and data type definitions.

Interpretation is aimed at representing the service's ontic commitment. This means unbundling and making as explicit as possible the real world (business) objects that the service descriptions recognize the existence of. In fact interpretation is defined as "the act of clarifying or explaining the meaning" of something (Collins Concise Dictionary 2001, p.761). Analogously identifying the real world objects that a service commits to is an act of clarifying the meaning of service descriptions.

Interpretation produces Service Ontic Commitment (SOC) models adopting the Object paradigm [6]. The Object paradigm, not to be confused with the Object-Oriented paradigm, was specifically designed for business modeling and is quite effective in precisely representing real-world semantics. Precise representation, in this case, refers to being able to clearly identify the mappings between the representation and the represented. It is beyond the scope of this paper to describe the Object paradigm in detail. It is sufficient to note that this paradigm models all "things" (including classes, individuals and relationships) as objects with a four-dimensional extension. The paradigm is attribute-less unlike more traditional paradigms (e.g., entity-relationship or object-oriented).

3.4 Concept Scoping

Concept Scoping is aimed at allocating the "committed" objects of the SOC models to pre-existing ontological models or, in the case of a newly explored domain, to newly developed ontologies. There are various ways in which content scoping can occur. With reference to an ontology language like OWL new objects (such as classes, properties and individuals) can be incorporated into an ontology as exemplified in Table 2.

Table 2. Methods of incorporating identified classes, properties and individuals.

Object Type	Method of Incorporation
Class	Define the class (a) in a newly developed ontology without any relation to pre-existing ontologies, (b) as a subclass of a class defined in a pre-existing ontology, (c) as an instance of a class defined in a pre-existing ontology and (d) as equivalent to a pre-existing class
Property types	Same as for classes
Individuals	Instantiate a class

3.5 Harmonization

Web services are resources which provide agents (human or software) with business offerings whose instantiations produce real world effects. Web services can use other web resources and can produce new resources. In this sense services will become an integral part of the Semantic Web and as such should be modeled similarly and in relation to other types of web resources. Harmonization is aimed at overcoming the traditional divide that is generally adopted between static and dynamic resources. The argument here is that if distinct types of representations are used for web services and other resource types, the necessary integration and semantic binding between them would become more difficult to resolve. Ontological models, which simultaneously represent all types of web resources, provide the benefit of facilitating the semantic discovery and composition of web services by software agents [11]. Agents would be able to traverse semantic graph lattices (or networks) in which services would be associated with the objects they use, transform and produce.

Harmonization uses the SOC models produced by Service Interpretation and the domain ontologies used in Concept Scoping to produce domain ontologies which incorporate service representations. The output artifact is represented in an ontology language such as OWL.

4 Financial Services Case

The research is grounded in a financial services case study which provides: (1) An external validity to the data that is seeding both the framework design process and subsequent scenario based usage analyses; (2) Less bias in that the software services being analyzed are the result of a service-orientation plan that did not encompass semantic web motivations; (3) A likely future industrial application of semantic web

technology as tools and techniques mature and are accepted within such a commercial context.

M-Bank is a leading European bank with both retail and treasury banking operations. The case being investigated resides in the treasury operation. Web services are used to support the reuse of functionality within a core processing system. This functionality comprises the management of trade cash flows and the rate fixing process. Trades may live for up to several decades and involve the transfer of cash between the two contracted parties involved in the trade (monthly, quarterly, etc.). Over time, fixing rates are applied to trades allowing the resulting cash flow to be calculated. It is the fixing rates and the cash flow schedules that are of interest to the trader, as these changes have both a funding and hedging impact. Web services were used to allow the spreadsheet trading console to interact with the operational system that holds the cash flows and fixing rates.

5 Semantic Transformation Applied

The framework presented in this paper was applied to web services described in WSDL. The WSDL code specified about 50 operations with relative parameters. Each operation provides externally accessible offerings and as such can be considered web services in their own right. The worked example presented in this section represents an extract of the semantic transformation carried out. This example refers to two web services. The first getRateSet returns the interest rate that has been fixed for a given trade settlement. The second web service is called getSchedule and returns the schedule of actual and projected settlements at a given point in time. The service receives as input reference to the trade and provides as output a table comprising of start and end dates of settlements, date in which the interest rate will be fixed (decided) for a specific settlement, currency, fixed rate, the notional amount of the settlement and the actual interest. From a semantic perspective such a representation has high levels of implicit meaning which need to be extracted and explicitly modeled. Tables 3 and 4 summarize the services.

Table 3. *getRateSet* web service.

Service name: getRateSet	
Description	This service provides the interest rate that has been fixed for a given settlement.
Input parameters	getRateSetSoapIn: String
Output parameters	getRateSetSoapOut: String

Table 4. *getSchedule* web service.

Service name: getSchedule	
Description	This service provides the schedule of all settlements related to a given trade.
Input parameters	getScheduleSetSoapIn: String
Output parameters	getScheduleSetSoapOut: String

5.1 Interpretation

As the diagrams of Figures 1 show, each part of a service can be unbundled and mapped to real world objects that clearly define the part's semantics. Figure 1 specifically refers to the *getRateSet* web service. Additionally for the interpretation of *getSchedule*, the service name refers to the classes *Trades* and *Schedules*, and the *temporally organize* relationship. Its input parameter, *getScheduleSoapIn*, relates to Trades and its output parameter, *getScheduleSoapOut*, relates to *Start Dates*, *End Dates*, *Rate Fixing Dates*, *Currencies*, *Interest Rates*, *Notional Amounts* and *Interest*.

The object paradigm, as stated previously, helps in this unbundling process given that all objects are explicitly revealed. The Service Ontic Commitment models shown here explicitly highlight those objects (in this case classes) that the individual elements of the services recognize the existence of, hence referring to such objects. Even relationships, such as *applied to* are represented as "committed" objects. This type of representation is similar to OWL in which relationships are explicitly represented as properties.

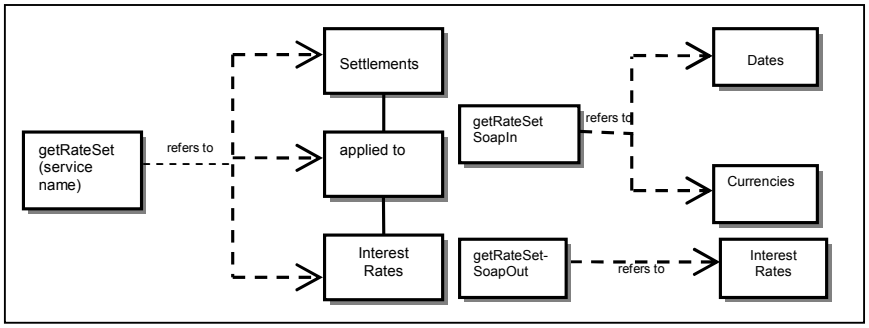

Fig. 1. Interpretation of *getRateSet* service name, input and output parameters.

5.2 Content Scoping

Content scoping allocates the objects identified in the Service Ontic Commitment models to domain ontologies. Within this example it is assumed that a decision was taken to develop a financial ontology and to allocate all objects to such a model. However classes such as D*ates*, S*tart Dates* and *End dates* are typical candidates of classes that are most likely to be scoped within the context of previously existing ontologies. In this case a *Time* ontology would most probably contain the definition of a *Date* class. As such it would be advisable to refer to such a class and subtype it with classes such as *Start Dates* and *End Dates*.

Figure 2 illustrates a first-cut ontological model derived from the previous interpretation phase.

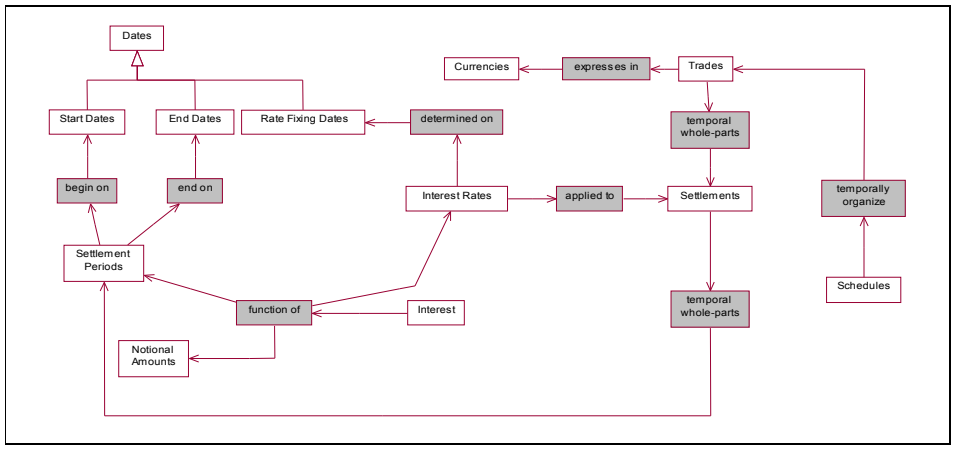

Fig. 2. First-cut financial domain ontology.

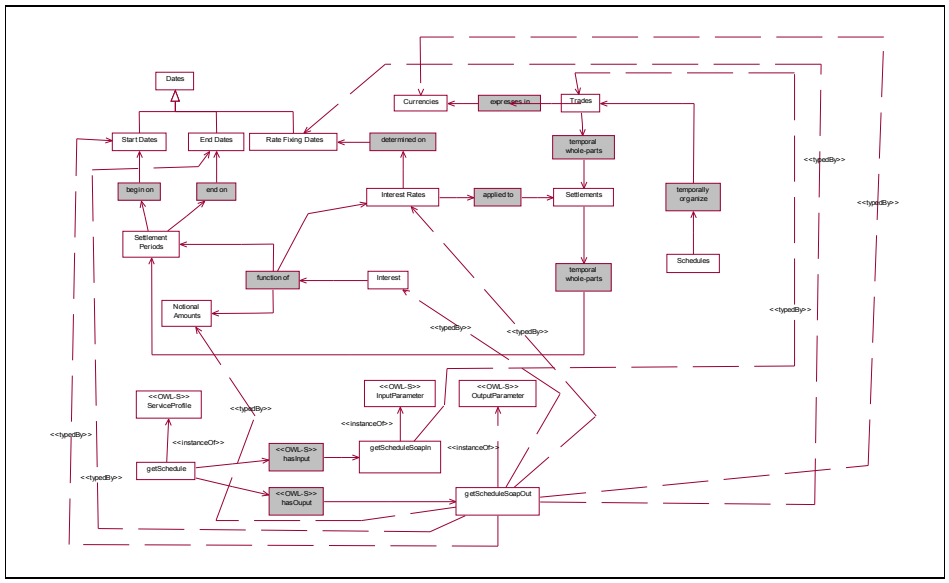

Fig. 3. Harmonized model.

5.3 Harmonization

In harmonization the web services are combined with the domain ontology. Ontologically this enables an explicit mapping between a service (with its parts) and the do-

main it serves. Figure 3 illustrates the harmonization model derived from the previous interpretation and content scoping.

6 Conclusion

This paper presented a framework for enabling the semantic transformation of syntactically defined web services. The framework defines an ontologically-based process in which syntactic service descriptions are interpreted to derive the objects that the services ontologically commit and refer to. The models produced by the interpretation phase are then used to scope the objects identified. These objects are either scoped to pre-existing web domain ontologies or used to develop new ontologies. Finally, the web services themselves are integrated with the domain ontologies. The final integration provides the basis for an effective semantic merging between all types of web resources and, as a consequence, facilitate the task of a software agent to navigate among various and semantically interlinked web services and domain objects (such as books, flights, etc.).

References

1. K. Sycara, M. Paolucci, J. Soudry, and N. Srinivasan, "Dynamic discovery and coordination of agent-based semantic Web services," *Internet Computing, IEEE*, vol. 8, pp. 66-73, 2004.
2. A. Paar, "Semantic software engineering tools " in *Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications* Anaheim, CA, USA ACM Press, 2003 pp. 90-91
3. S. Staab, W. van der Aalst, V. R. Benjamins, A. Sheth, J. A. Miller, C. Bussler, A. Maedche, D. Fensel, and D. Gannon, "Web services: been there, done that?," *Intelligent Systems, IEEE [see also IEEE Intelligent Systems and Their Applications]*, vol. 18, pp. 72-85, 2003.
4. S.A. McIllraith, D.L. Martin , "Bringing Semantics to Web Services" *Intelligent Systems, IEEE*, vol. 18, pp. 90-93, 2003.
5. C. Partridge, *Business Objects: Re-Engineering for Reuse*. Oxford: Butterworth-Heinemann, 1996.
6. A. Daga, S. de Cesare, M. Lycett, and C. Partridge, "An Ontological Approach for Recovering Legacy Business Content," in *Proceedings of the 38th Hawaii International Conference on System Sciences*. Los Alamitos, CA: IEEE Computer Society, 2005.
7. D. Fensel and O. Lassila, "The semantic web and its languages," *Intelligent Systems and Their Applications, IEEE [see also IEEE Intelligent Systems]*, vol. 15, pp. 67-73, 2000.
8. T. Honderich, *Oxford Companion to Philosophy*. Oxford: Oxford University Press, 1995.
9. T. R. Gruber, "A translation approach to portable ontology specifications," *Knowledge Acquisition*, vol. 5, pp. 199-220, 1993.
10. G. Frege, *The foundation of Arithmetic: A logico-mathematical enquiry into the concept of number*, 1884.
11. J. Hendler, "Agents and the Semantic Web," *Intelligent Systems, IEEE [see also IEEE Intelligent Systems and Their Applications]*, vol. 16, pp. 30-37, 2001.