Foundations for Service Ontologies: Aligning OWL-S to DOLCE

Peter Mika
Department of Business Informatics
Vrije Universiteit Amsterdam, The Netherlands
pmika@cs.vu.nl

Daniel Oberle
Institute for Applied Informatics and Formal Description Methods (AIFB)
University of Karlsruhe, Germany
oberle@aifb.uni-karlsruhe.de

Aldo Gangemi
Laboratory for Applied Ontology (LOA)
Institute for Cognitive Sciences and Technology
National Research Council, Rome, Italy
gangemi@ip.rm.cnr.it

Marta Sabou
Department of Artificial Intelligence
Vrije Universiteit Amsterdam, The Netherlands
marta@cs.vu.nl

ABSTRACT

Clarity in semantics and a rich formalization of this semantics are important requirements for ontologies designed to be deployed in large-scale, open, distributed systems such as the envisioned Semantic Web. This is especially important for the description of Web Services, which should enable complex tasks involving multiple agents. As one of the first initiatives of the Semantic Web community for describing Web Services, OWL-S attracts a lot of interest even though it is still under development. We identify problematic aspects of OWL-S and suggest enhancements through alignment to a foundational ontology. Another contribution of our work is the Core Ontology of Services that tries to fill the epistemological gap between the foundational ontology and OWL-S. It can be reused to align other Web Service description languages as well. Finally, we demonstrate the applicability of our work by aligning OWL-S' standard example called *CongoBuy*.

Categories and Subject Descriptors

H.4.m [**Information Systems**]: Miscellaneous; H.3.5 [**Information Storage and Retrieval**]: Online Information Services—*Web-based Services*

General Terms

Languages, Design

Keywords

Semantic Web, Web Services, DAML-S, OWL-S, DOLCE, Descriptions and Situations, Core Ontology of Services

1. INTRODUCTION

Ontologies are the basic infrastructure for the Semantic Web whose idea hinges on the possibility to use shared vocabularies for describing resource content and capabilities. Clarity in semantics and a rich formalization of this semantics are important requirements for ontologies designed to be deployed in large-scale, open, distributed systems such as the envisioned Semantic Web. This is due to the fact that ontologies should facilitate mutual understanding, either for enabling effective cooperation between multiple artificial agents, or for establishing consensus in a mixed society where artificial agents cooperate with human beings. Foundational ontologies fulfill those requirements because they serve as a starting point for building new domain and application ontologies, provide a reference point for easy and rigorous comparisons among different ontological approaches and create a framework for analyzing, harmonizing and integrating existing ontologies and metadata.

Clarity in semantics together with a rich formalization are especially important for ontologies describing Web Services because they enable complex tasks involving multiple agents. Web Services are becoming ever more important resources on the Web and standards are being developed for their syntactic description [5]. As one of the first initiatives of the Semantic Web community for semantically describing Web Services, OWL-S [10], formerly known as DAML-S, attracts a lot of interest even though it is still under development. OWL-S is an ontology of general concepts aiming at automatic discovery, composition and invocation of Web Services.

Our contribution to the development of this ontology is to identify some of its problematic aspects and to suggest enhancements through alignment to a foundational ontology. We found that OWL-S suffers conceptual ambiguity, lacks concise axiomatization, is designed too loosely and offers an overly narrow view on Web Services.

Typically, service descriptions cross the boundary between an information system (with objects such as a record about a book) and the external world (with objects such as the physical book). The reason is that the Web Services are only a part of the overall service to which a value is attributed by the requestor. We believe that this phenomenon will characterize most real world services, where users are paying not simply for their information being recorded and manipulated, but for the overall process, which includes actual changes and effects in the real world, such as a book being delivered.

In addition, descriptions are independent views on a world (real or imagined) by the various actors involved and may significantly differ in the notions that are used (e.g. information vs. book) and the granularity of the descriptions (high level tasks vs. detailed processes). Similarities among such views are to be found on the level of *constructs* used to describe these views: both of them discuss

WWW2004, May 17–22, 2004, New York, NY USA.
ACM 1-58113-844-X/04/0005.

roles that can be played by a number of objects and plans or courses of events which can be realized by different sequences of activities.

Through our alignment, we have discovered possible enhancements to these problematic aspects of the ontology. We present these findings for the benefit of the designers and users of OWL-S. We also detail how the OWL-S coalition's standard example, called CongoBuy, is aligned. Furthermore, a Core Ontology of Services is developed as a middle layer which can also be used for aligning other (Web) Service description languages. Lastly, we note that the contribution of our work is not limited to the concrete results reported in this paper, but rather consists of (1) examples of the benefits of alignment to foundational ontologies and (2) a description of the alignment method itself.

The paper is structured as follows. We begin with related work in Section 2. Section 3 identifies and explains several problematic aspects of OWL-S and can also be seen as a motivation for our work. Section 4 presents the main body of work, viz. the alignment of OWL-S to the DOLCE foundational ontology. It also includes a short introduction to the foundational ontology and the alignment of the CongoBuy example. Section 5 details our suggested improvements to the problematic aspects introduced before. We conclude in Section 6.

2. RELATED WORK

Previous efforts responded to some of the problems of OWL-S. We briefly discuss the two initiatives we are aware of by describing their motivation, the parts of OWL-S they focus on, the techniques they use as well as some initial results (when available).

The first initiative [14] is motivated by the need of formal semantics to describe, simulate, automatically compose, test and verify Web Service compositions. It focuses solely on the OWL-S *ServiceModel* which provides all the constructs for specifying composition. The authors establish a situation calculus semantics for the main elements in the OWL-S *ServiceModel* (e.g. atomic and composite processes, conditional effects and outputs), then translate it to the operational semantics provided by Petri Nets. This knowledge representation formalism has a rich theoretical and tool support for the various composition tasks. Indeed, this semantics allowed to re-use an existing simulation and modelling environment. Further, the authors were able to identify more tractable subsets of OWL-S (less expressive but more efficient analysis for verification, composition and model checking).

The second effort [1] also focuses only on the OWL-S *ServiceModel* and proposes a concurrent operational semantics that incorporates subtype polymorphism. The motivation for this work is to provide an initial reference semantics that would discover any possible ambiguity in the developed language. It would also serve for developing techniques for automated verification of OWL-S models. Finally, if other Web standards would provide a similar semantics it would be much easier to compare them and to understand their strengths and weaknesses. The authors of both efforts mutually acknowledge the similarity between the two proposed semantics, except some minor details discussed in [1].

Besides aiming at increased formal axiomatization, we wish to explain the OWL-S concepts in terms of a foundational ontology which reflects several generally accepted theories from linguistics, philosophy, cognitive sciences etc. We show that this "ontological" analysis of OWL-S also brings to surface several irregularities in the model (just like the reference semantics promises to do). Further, one of the long term benefits of alignment is that it allows a comparison between several aligned ontologies (a goal also stated in [1]). As a result we extend our analysis to the entire OWL-S model. From a methodological perspective, the previous approaches provide independent reconstructions of OWL-S, while, through alignment, we embed the OWL-S model in the larger context offered by the foundational ontology. Therefore we can deduce e.g. that OWL-S does not address the difference between a real life object (e.g. book) and its representational counterpart in an information system (e.g. ISBN number), an important ontological distinction. Finally, the semantics established by previous work are not reflected in the current OWL formalization of the model. In our case, the model inherits the axiomatization available for the OWL-DL version of DOLCE.

3. PROBLEMATIC ASPECTS OF OWL-S

This Section identifies and illustrates some of the problematic aspects of understanding OWL-S from a foundational perspective. We revisit most of them when discussing some of our suggestions for improvements in Section 5. We also relate these issues to the question of ontology quality.

Ontology quality is the topic of [4], which provides (among others) three criteria for evaluation: *extensional coverage* (concerning the amount of entities that are supposed to be described by an ontological theory), *intensional coverage* (concerning what kinds of entities are described by an ontological theory), and *precision* (concerning what axioms are required to describe just the models the ontology designer intends to cover). According to these criteria, a good ontology should approximate the domain of discourse that is supposed to be described, it should have a signature that maps all the kinds of entities intended by the designer, and it should axiomatize the predicates in order to: 1) catch all the intended models, and 2) exclude the unintended ones.

Below we introduce four problems encountered in OWL-S. The first one (conceptual ambiguity) features both insufficient intensional coverage and overprecision. The second and the third (poor axiomatization and loose design) are cases of insufficient precision. In the third problem, the weakness is mainly inherited by limitations of OWL expressivity. The fourth (narrow scope) is a case of both extensional and intensional coverage.

3.1 Conceptual Ambiguity

Since there is no clear conceptual framework behind OWL-S, it is often difficult for users to understand the intended meaning of some concepts, the relationship between these concepts as well as how they relate to the modelled services. Many concepts are still being clarified both within the OWL-S coalition and in public mailing lists[1]. In addition, the Web Services Architecture (WSA) Working Group of the W3C introduced an OWL ontology of Web Service concepts that seems to be independent of OWL-S[2]. This probably leads to the necessity of an alignment between the two ontologies, which needs an explication of the respective assumptions.

Conceptual ambiguity affects particularly the upper level of OWL-S shown in Figure 1. The notion of a service is introduced in [10] as follows: "By 'service' we mean Web sites that do not merely provide static information but allow one to effect some action or change in the world, such as the sale of a product or the control of a physical device". Later, we read that "any Web-accessible program/sensor/device that is declared as a service will be regarded as a service".

However, neither of these definitions are operationalized as neither the concept of a "Web site" nor the "Web" appears in the ontology. Instead, the notion of a service is characterized solely by

[1]cf. `http://www.daml.org/services`
[2]cf. `http://www.w3.org/2004/02/wsa/`

its relationship to a number of *ServiceProfiles*, at most one *ServiceModel* and any number of *ServiceGroundings*, which is not sufficient to understand the concept of *Service* considered by OWL-S.

We note that the term Web Service and closely related terms (e-Service, Service, etc.) also suffer from overloading. In our search for possible formalizations, we found a variety of definitions emphasizing different aspects of a service [8]: offering functionality (usefulness for a particular task), interoperability using standards or providing an interface to an existing system. We also refer the reader to the work of Baida et al. [3], which compares and contrasts the definitions used in the business literature, in software engineering and in information sciences.

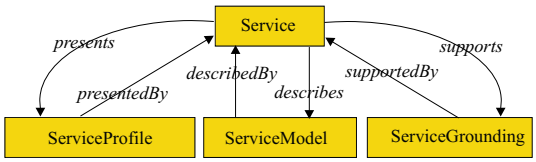

Figure 1: OWL-S Service Ontology

3.2 Poor axiomatization

OWL-S' goal is to be machine processable and it operates in an open environment. Hence, it is important that each concept is characterized by a rich axiomatization in order to support meaningful inferences. In general, we believe that the level of commitment in OWL-S will need to be raised if it shall support the complex reasoning tasks put forward by the coalition.

Unlike the issue mentioned in the previous section, poor axiomatization reflects the lesser problem when the definition of concepts is clear, but axiomatization in the ontology itself needs improvement. In many respects, OWL-S shows the characteristics of a typical application ontology: there is no firm concept or relation hierarchy (most concepts and relations are direct subconcepts of the top level concept or relation) and several relations take *owl:Thing* as their domain or range.

We propose that by adding foundations to OWL-S, the level of axiomatization can be increased. Alignment to a foundational ontology means relating the concepts and relations of an ontology to the basic categories of human cognition investigated by philosophy, linguistics or psychology. This approach has the advantage that restrictions on the level of common sense are inherited by the concepts in the application ontology. This prompts the ontology engineer to sharpen his notions with respect to the distinctions made in the foundational ontology. It also promotes reuse by highlighting commonalities, which especially helps to reduce the proliferation of relations that is typical for application ontologies.

Alignment to a well-modularized foundational ontology also allows to selectively import theories from the ontology such as mereology, time theory etc. We will demonstrate this in Section 5 when aligning the control constructs of OWL-S to the Ontology of Plans which is one of the basic extensions of the DOLCE foundational ontology.

3.3 Loose design

A further problematic aspect of OWL-S from an ontologist's point of view is its entangled design. At the heart of this problem lies the purpose of OWL-S in providing descriptions of various views on Web Services required to support a number of different service related tasks (discovery, composition, invocation). Besides the functional dimension, Web Service descriptions should be contextualized to represent various points of view on a service, possibly with different granularity.[3] Most of these views, however, are overlapping in that they concern some of the same attributes of a service.

A straightforward modularization in such cases results in an entangled ontology, where the placement of certain knowledge becomes arbitrary and intensive mapping is required between modules. This phenomenon is well described in object-oriented design, where the notion of *aspects* [6] was recently proposed to encapsulate concerns that cross-cut the concept hierarchy of a software.

A case in point is the application of attribute binding in OWL-S. The construct of attribute binding is necessary in OWL-S to express, for example, that the output of one process is the input for another process or that the output of a composite process is the same as the output of one of its subprocesses. In programming, such equivalences are expressed by the use of *variables*. Variables are governed by the rules of *scoping*, which define the boundaries of commitment.

Since OWL lacks the notion of variables, argument binding is expressed by explicit value maps. As shown in Figure 2, the value map has the form of a *List*, attached to a *ProcessComponent*. This *List* should contain instances of the *ValueOf* concept as members[4]. Each *ValueOf* concept should point to a single relation of a single concept by using *theParameter* and *atProcess* relations[5]. For example, in case of two processes A and B where process B takes the output of process A as an input, the list would have two *ValueOf* members, one related to concept A and the output relation, while the other would be related to concept B and its input relation.

The reader may also note that the intended meaning of the entire construct, namely that all 'sensible' instantiations of the process should respect the equivalences expressed in the value map, is not encoded in the axiomatization. This is explained by the lack of expressivity of the Description Logic used.

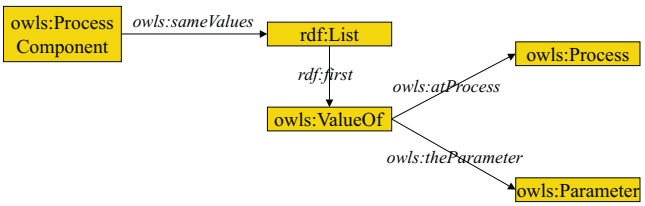

Figure 2: The representation of attribute binding in OWL-S

Besides a tedious representation, an unfortunate consequence of the present solution is that we can only guess about the scope of the commitment represented by the value map. OWL-S seems to suggest attaching the value map to the process whose sub-processes are involved in the value map. As argued above, however, there could be multiple value map restrictions on the inputs/outputs of a process resulting from service composition (expanding/collapsing processes). Taking the current OWL-S proposal, it is unclear how one could approach such a situation.

3.4 Narrow scope

Like mentioned in Section 1, the scope of OWL-S needs to be extended to represent real world services that naturally cross the lines

[3]The OWL-S specification mentions the ability to use the *Profile* for providing such views. However, no actual constructs are provided to map them to possible service executions or to each other.
[4]However, this is not enforced. There's also no explanation given why an ordered collection is used, i.e. what the ordering means.
[5]The cardinality restrictions are missing from the formalization.

between information systems and the physical world. While OWL-S acknowledges this aspect of services, it is unclear how a distinction could be made between the objects and events within an information system (regarding data and the manipulation of data) and the real world objects and events external to such a system. Using a foundational ontology, however, it is possible and even required for the creator of a description to make such distinctions, because they fundamentally affect the ontological nature of the objects and events concerned. We will return to this issue in Section 5.4.

Besides its insufficient intensional coverage, the OWL-S core also shows an overcommitment in precision: the top *Service* concept is related to the *ServiceModel* concept with a cardinality 1:1. This means that for each *Service*, only one *ServiceModel* is expected to hold. This prevents us to consider alternative *ServiceModels*, or to evaluate the relationship between a *ServiceModel* required by a customer's guideline, or by a legal regulation, and the one underlying the provider's system, for instance.

A further contribution of our work is to extend OWL-S with relationships for mapping between descriptions of service and the elements of actual service executions, which are not yet covered by OWL-S. These relationships will be directly inherited from the Descriptions & Situations ontology, another module of DOLCE, which is introduced in Section 4.2.

4. ALIGNMENT

This Section shows how we align OWL-S to a foundational ontology. For the latter, we have chosen DOLCE which is explained in Section 4.1. It is extended by an ontology of Descriptions & Situations further detailed in 4.2. As the epistemological gap between OWL-S and Descriptions & Situations is too large, we constructed a Core Ontology of Services (Section 4.3). 4.4 and 4.5 depict how OWL-S' concepts are to be expressed by using the Core Ontology of Services and how the CongoBuy example is aligned, respectively. We give a short summary of this alignment methodology in 4.6.

4.1 DOLCE

The role of foundational ontologies is to serve as a starting point for building new ontologies, to provide a reference point for easy and rigorous comparisons among different ontological approaches, and to create a foundational framework for analyzing, harmonizing and integrating existing ontologies and metadata standards. They are conceptualizations that contain specifications of domain independent concepts and relations based on formal principles derived from linguistics, philosophy, and mathematics.

DOLCE, a Descriptive Ontology for Linguistic and Cognitive Engineering, belongs to the WonderWeb project Foundational Ontology Library (WFOL) and is designed to be minimal in that it includes only the most reusable and widely applicable upper-level categories, rigorous in terms of axiomatization and extensively researched and documented [15, 11].

The upper part of DOLCE's taxonomy is sketched in Figure 3. DOLCE is based on a fundamental distinction between enduring and perduring entities. The main relation between *Endurants* (i.e. objects or substances) and *Perdurants* (i.e. events or processes) is that of participation: an endurant "lives" in time by participating in a perdurant. For example, a person, which is an endurant, may participate in a discussion, which is a perdurant. A person's life is also a perdurant, in which a person participates throughout its duration. *Qualities* can be seen as the basic entities we can perceive or measure: shapes, colors, sizes, sounds, smells, as well as weights, lengths or electrical charges. Spatial and temporal qualities encode the spatio-temporal attributes of objects or events. Finally, *Abstracts* do not have spatial or temporal qualities, and they are not qualities themselves, e.g. (quality) regions or sets. In particular, regions are used to encode the measurement of qualities as conventionalized in some metric or conceptual space.

DOLCE is axiomatized in a modal logic (S5), but it is maintained also in other languages, used according to the particular trade-off between expressivity and computational complexity that is required by a certain application. For example, the KIF version is suited for most detailed meaning negotiations and for machine-readability of the complete axiomatization. The Loom [9] version has been used until recently to support ontology-driven industrial applications that required both high expressivity and classification services; in this version, some modal and temporal axioms have been removed or transformed, in order to take advantage of the Loom variety of description logic (which is incomplete, but desirable in certain settings). The OWL-DL [12] version is currently maintained for Semantic Web applications. Due to the variety of description logic expressed by OWL-DL, some other constructs have been removed, such as so-called value maps (relation composition). It probably provides the best scaled version due to the completeness of OWL-DL[6]. The strategy applied in porting DOLCE to different languages is quite liberal, and consists in finding the most appropriate naming policy and constructs that sound natural within the best modelling practices for a certain language, provided that the subsumption hierarchy and the axioms have an accurate mapping to the reference S5 version.

Although out of the scope of the paper, we should mention that DOLCE has been chosen as basis for several reasons, some due to its internal structure (rich axiomatization, explicit construction principles, careful reference to interdisciplinary literature, common sense-orientedness, etc.), others due to its modular nature. In fact, being part of the WonderWeb Foundational Ontology Library, DOLCE will be mapped to other foundational ontologies (possibly more suitable for certain applications), and will be extended with many modules covering different domains (e.g. legal and biomedical), problems (e.g. planning, contexts), and lexical resources (e.g. WordNet-like lexica). These features (internal consistency and external openness) make DOLCE specially suited for our needs.

4.2 Descriptions & Situations

While modelling physical objects or events in DOLCE is quite straightforward, intuition comes to odds when we want to model non-physical objects such as social institutions, plans, organizations, regulations, roles or parameters. This difficulty is due to the fact that the intended meaning of non physical objects results from statements, i.e. their meaning emerges only in the combination of other entities. E.g. a norm, a plan, or a social role are usually represented as a set of statements and not as a concept. On the other hand, non physical objects may change and be manipulated similar to physical entities, and are often treated as first-order objects. That means an ontology should account for such objects by modelling the context or frame of reference on which they depend. The representation of context is a common problem in many realistic domains from technology and society (such as law or finance) which are full of non physical objects.

In order to respond to those modelling requirements, we developed a module for DOLCE, called Descriptions & Situations (D & S) [7]. D & S results to be a theory of ontological contexts because it is capable of describing various notions of context or frame of reference (non physical situations, topics, plans, beliefs, etc.) as entities. It features a philosophically concise axiomatization.

[6]The OWL-DL version of DOLCE can be found at `http://www.loa-cnr.it/DOLCE.html`

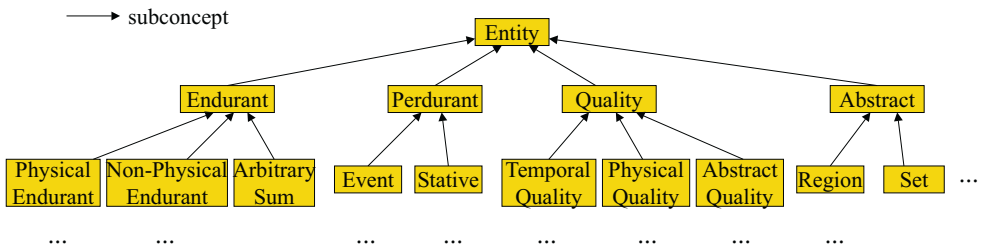

Figure 3: The top-level taxonomy of DOLCE

Like depicted in Figure 4, D & S introduces a new category *Situation* that reifies a state of affairs and is constituted by entities of the ground ontology (in our case DOLCE). A *Situation satisfies* a *Situation Description (S-Description)*, which is aligned as a dolce:non-physical endurant and is composed of descriptive entities (*C-Descriptions*), i.e. *Parameters*, *Functional Roles* and *Courses of Events*. Axioms enforce that each descriptive component links to a certain category of DOLCE: *Parameters* are *valued-by Regions*, *Functional Roles* are *played-by Endurants* and *Courses of Events sequence Perdurants* (cf. Figure 5).

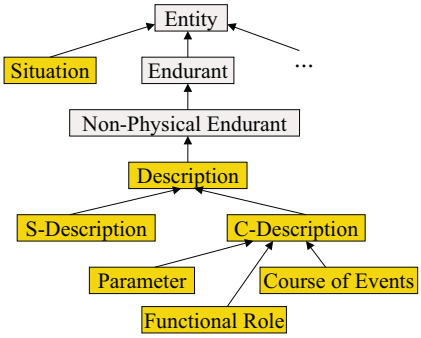

Figure 4: Aligning D & S to DOLCE

S-Descriptions can be used to model contexts, for example, a murder (situation) that has been reported by a witness (functional role), which is played-by a person (endurant), in a testimony (s-description). The same situation may be interpreted according to other, alternative descriptions. This captures that multiple overlapping (or alternative) contexts may match the same world or model, and that such contexts can have systematic relations among their elements.

D & S shows its practical value when applied as an *ontology design pattern* for (re)structuring application ontologies that require contextualization. As we will see in the remainder of this Section, this is the case when describing (Web) Services.

4.3 A Core Ontology of Services

The descriptions of services show a clear contextual nature and are to be modelled as *Situation Descriptions* in the sense of DOLCE and Descriptions & Situations.[7] One may only have to consider the number of different views that may exist on a service: the view of a service provider, that of the service requestor or the legal view of a contract etc. The concepts used to formulate any given view are clearly separate from the actual objects they act upon and often independent from the concepts appearing in other views.

[7]In the following, we will refer to DOLCE with its basic extensions, i.e. D & S, Ontology of Plans, etc., as DOLCE+.

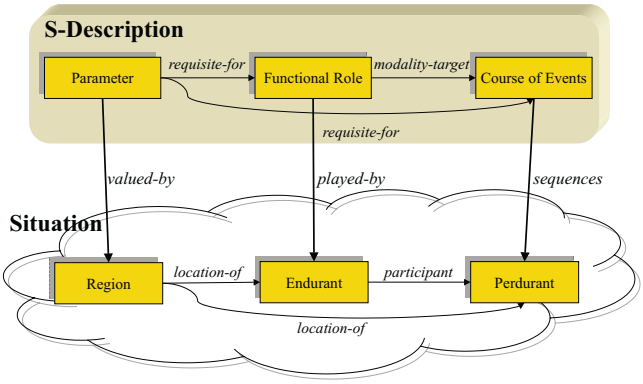

Figure 5: Descriptions and Situations

Different views on the service need not be equally detailed either. For example commercial advertisements typically feature only selected characteristics of a service. The various views also naturally focus on different aspects of a service, which means that the descriptions may only be partially mapped to each other.

Instead of directly aligning OWL-S to Descriptions & Situations, we developed a Core Ontology of Services (COS) and aligned the OWL-S sources to this ontology. This two-stage alignment is a common technique when the conceptual gap between the source ontologies and the foundational ontology is large. The Core Ontology of Services also features a concise axiomatization detailed in [8] and can be reused in other scenarios (e.g. purely commercial services).

Currently, we consider five frequently occurring descriptions of a service, where each represents a separate viewpoint: (Service) Offering, Request, Agreement, Assessment and Norms (more views may be added in the future when needs arise). All service views are specializations of *S-Description* defined in the Descriptions & Situations ontology (cf. Figure 6).

We also introduce specializations of *Course of Event*, viz. *Task*, *Service Task* and *Computational Task*. This allows us to model activities in an information system and in the real world. Axioms ensure that *Service Tasks* only sequence *Service Activities* and that *Computational Tasks* only sequence *Computational Activities*. The activities are new kinds of perdurants especially introduced here (not shown in Figure 6). Further axioms also ensure that only *Information Objects* (a newly introduced non-physical endurant) participate in *Computational Activities*. Our Core Ontology of Services may optionally take advantage of a number of concepts from the Ontology of Plans which is another module for DOLCE+. It allows the division of tasks into elementary and complex and the construction of complex tasks from elementary ones among other features.

The Core Ontology of Services also models frequently occurring *Functional Roles*. The *Requestor* and *Provider* of a service are conceived as *Legally Constructed Person*s, an agentive legal role in DOLCE, while the *Executor* of a service is considered an agentive functional role without a legal nature. Another group of roles is played by the instruments used in services. These include *(Computational) Inputs* and *Outputs*, formalized as instrumentality roles. Our comprehensive axiomatization requires that, e.g., a *Computational Input* is only played by an *Information Object*.

In addition, we introduce the role of Value Objects in the sense of Baida et al. [2] as a subtype of the generic DOLCE+ commerce-role. Such a role distinguishes generic Inputs/Outputs from ones to which a value is attributed. The latter is usually done by the actor whose viewpoint is being modelled.

Note that there are other features in this ontology which are neglected here due to the lack of space. The interested reader is referred to [8].

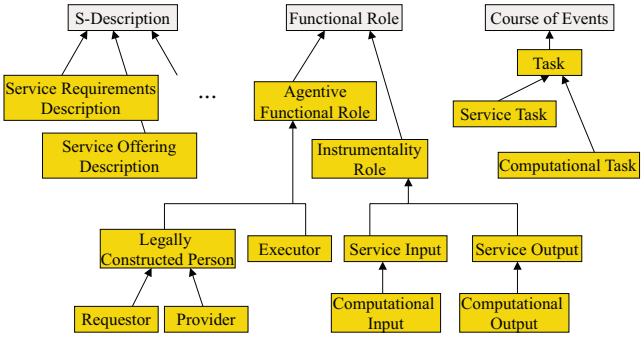

Figure 6: Aligning the Core Ontology of Services to D & S

4.4 Aligning OWL-S to the Core Ontology of Services

In the following we describe the alignment of OWL-S to the Core Ontology of Services and our experiences with the process.[8]

The process of alignment proved its value early on by allowing us to quickly separate concepts of the ontology that had no clear and unique ontological interpretation with respect to the basic categories of DOLCE. For example, the *ValueOf* concept, which seems to be introduced for technical reasons (see Section 3.3). Similar arguments hold for the *ConditionalEffect* class, which models a ternary relationship between a process, a precondition and an effect. Much like *ValueOf*, this class is introduced for representation purposes, its real semantics are not captured by the ontology[9]. Similarly, the distinctions between *ServiceProfile* and *Profile* as well as *ServiceModel* and *ProcessModel* are introduced in OWL-S to provide flexibility in modelling, rather than representing conceptual differences. Although the definition of *Service* is ambiguous even in the natural text description of OWL-S, for the sake of argument we considered *Service* as a Service Offering Description, which has the *ServiceProfile* and *ServiceModel* (also Service Offering Descriptions) as parts. Note that the *ServiceProfile* just expands the *ServiceModel* by process descriptions. In our opinion there is no need to separate both and, e.g., have parameters like *serviceName* only in the *Profile* and not in the *ProcessModel*. However, our intention was to just align OWL-S rather than reorganizing it.

[8]In the following concepts printed in italics are part of the OWL-S namespace, except when indicated otherwise.

[9]This leaves open to interpretation for example the case when multiple conditional effects are given for a process.

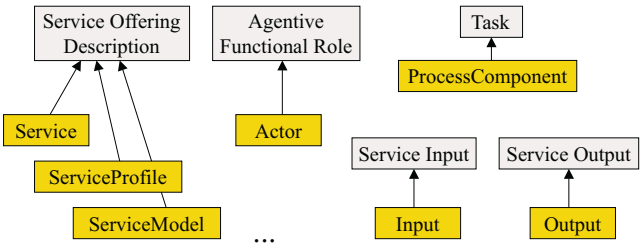

Figure 7: Aligning OWL-S to the Core Ontology of Services

Relations like *serviceName* or *textDescription* regard the profile as a whole and are thus aligned with Service Offering Description as domain and literal as range. The notion of Actor in the *ServiceProfile* is aligned as an Agentive Functional Role like depicted in Figure 7. The *ProcessComponent* concept was aligned to the Task concept of the Core Ontology of Services, while the individual control constructs were mapped to task components included from the Ontology of Plans. For example, the *Repeat-until* control construct was aligned to the cycle-until task concept, a kind of cyclical-task with an exit condition and/or repetition interval. As another example, the *If-then-else* construct maps to the notion of an alternate task, a case-task with exactly two branches (both are not shown in Figure 7). Note that there is a difference, however, between the Control Constructs of OWL-S and the task types of the Core Ontology, because task types, like all other tasks, sequence activities themselves (branching and synchro tasks, in particular, sequence planning activities).

The disambiguation of Inputs, Outputs, Preconditions and Effects (IOPE) was relatively straightforward using the Core Ontology. *Input* and *Output* are aligned to Service Input and Service Output, respectively. On the other hand, the notions of Precondition and Effect are inherited from the Ontology of Plans where they are modelled as Situations and linked to their respective tasks using the task-precondition and task-postcondition relationships.[10] Conditional Outputs and Conditional Effects are modelled using the case-task construct.

We omitted the alignment of the grounding ontology for WSDL [5] because it was not the focus of our work. Nevertheless, the notion of *Software Tool* is present in the Core Ontology of Services as *Information Object* that can be expressed according to any number of description systems.[11] WSDL could be such a description system and modelled to the extent required to express groundings.[12]

4.5 Aligning the Congo Example

The OWL-S coalition uses CongoBuy, a fictitious book-buying service, as standard example for didactic purposes. The provider, Congo Inc., publishes a suite of programs on the Web. These programs (self-described by their names) are LocateBook, PutInCart, SignIn, CreateAcct, CreateProfile, LoadProfile, SpecifyDeliveryDetails, FinalizeBuy.[13]

[10]For preconditions, this means that $task - precondition(p, c) \Leftrightarrow Process(p) \wedge PreCondition(pc) \wedge Condition(c) \wedge hasPrecondition(p, pc) \wedge preCondition(pc, c)$.

[11]A more refined representation we considered was to model *Software* as an S-Description, in the sense of an abstract algorithm.

[12]The Core Ontology of Services, the OWL-S and CongoBuy alignments are available for download at `http://www.cs.vu.nl/~pmika/research/www2004/`

[13]`http://www.daml.org/services/owl-s/1.0/examples.html`

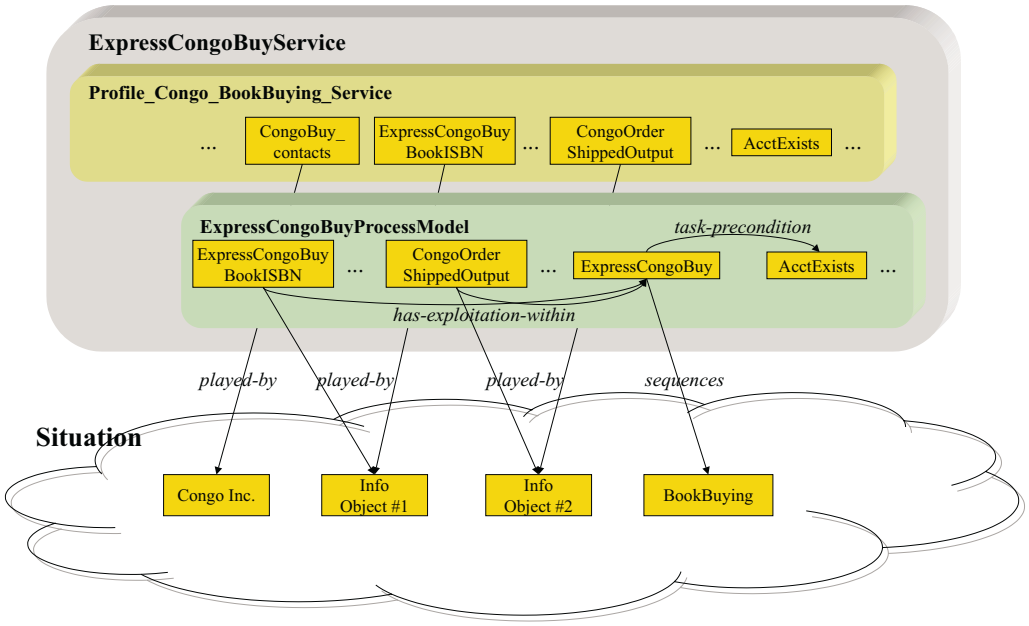

Figure 8: The CongoBuy example.

The example defines two Service instances that mainly differ in their process descriptions. While *ExpressCongoBuyService* only describes an AtomicProcess called *ExpressCongoBuy*, *FullCongoBuyService* describes a more complicated process, *FullCongoBuy*, that uses control structures including alternatives, conditional executions, etc. We limit ourselves to the first Service instance in the following.

Like explained in the previous section, Service is aligned as Service Offering Description (SOD) with both Profile and Process descriptions as parts. Hence, Figure 8 shows *ExpressCongoBuyService* as container for both *Profile_Congo_Book_Buying_Service* and *ExpressCongoBuyProcessModel*.

Profile_Congo_Book_Buying_Service also is a Service Offering Description with *serviceName* and *textDescription* as literal relations (DataTypeProperties in OWL). *CongoBuy_contacts* represents an instance of the Actor role played by a Legally Constructed Person (*Congo Inc.*) in the Situation. Finally, the following IOPEs are related to this description: *ExpressCongoBuyBookISBN* and *CongoBuySignInInfo* which are Inputs played by Information Objects, *CongoOrderShippedOutput* and *CongoOutOfStockOutput* are played by Information Objects, the Preconditions *AcctExists* and *CreditExists* as well as *CongoOrderShippedEffect*. Note that Preconditions and Effects are aligned as additional Situations, independently from those satisfying the Service. In other words, the Service model does not specify how Preconditions or Effects can be the case.

ExpressCongoBuyProcessModel features *ExpressCongoBuy*, an AtomicProcess which is aligned as dolce:elementary-task. It sequences a newly introduced Service Activity called *Book Buying*. The Inputs *ExpressCongoBuyBookISBN* and *CongoBuySignInInfo* as well as the Outputs *CongoOrderShippedOutput* and *CongoOutOfStockOutput* have exploitation within this task. The Preconditions *AcctExists* and *CreditExists* become Situations and are linked to the Task by the task-precondition relation. Similar holds for *CongoOrderShippedEffect* which is related to *ExpressCongoBuy* by task-postcondition. Like mentioned in the previous Section, it becomes obvious now that the Process description is just an extension of the Profile description. All the Inputs and Outputs are identical in both descriptions what creates unnecessary modelling overhead just like the *ExpressCongoBuyService* description solely acting as a container.

Finally, the Congo example introduces new domain concepts that have to be aligned to DOLCE categories. The original Process ontology defined concepts like *CreditCardType*, *PackagingType*, *DeliveryType*, *ValidityType* or *Cart*. *CreditCardType*, for instance, was defined by means of *owl:oneOf* with *MasterCard*, *VISA*, *AmericanExpress* and *DiscoverCard* being instances of *CreditCardType* themselves. After the alignment, *CreditCardType* became a subconcept of dolce+:legal-document which itself is subconcept of dolce+:information-object. Note that we modelled the card as an information artifact rather than a physical object, as it is the information artifact that actually participates in the execution of a Web Service (the physical card could even be missing).

4.6 Summary

The ontology stack in Figure 9 summarizes our alignment effort. We used DOLCE as foundational ontology (4.1), extended it by the Descriptions & Situations module (4.2), defined our Core Ontology of Services (4.3), which was used to align OWL-S (4.4) and a concrete domain ontology (4.5). Note that this methodology of alignment could be used to align and compare other service description efforts as well, e.g. the Web Services Architecture (WSA) or the ontology used within the Application Server for the Semantic Web (both alignments are detailed in [8]). Specialized domain and application ontologies of service descriptions such as [17] are formulated according to one of these generic service ontologies.

Our method was a combination of a bottom-up and a top-down approach. On the one hand, ontologies in the lower layers provided representation requirements for the higher layers, which abstracted their concepts and relationships. On the other hand, the upper layers provided design guidelines to the lower layers. This also meant that although our goal was to preserve the structure of OWL-S as much as possible, our method suggested a rearrangement of the ontology based on the backbone provided by the D & S ontology.

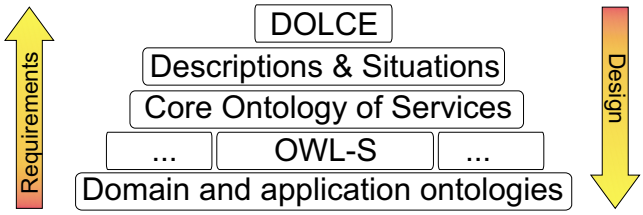

Figure 9: Ontology stack

5. SUGGESTIONS FOR IMPROVEMENT

This Section highlights suggestions for improvement of the problematic aspects of OWL-S, depicted in Section 3. Note that each subsection corresponds to the one introduced in Section 3.

5.1 Conceptual Disambiguations

The alignment to a foundational ontology helped us in understanding and crystalizing several concepts of OWL-S. As an example, ontological analysis explained the difference between an information object, its application domain counterpart and the role it plays in an information system (cf. also Section 5.4). This indicated possible enhanced modelling: since the same information object is modelled both in the *ServiceProfile* and *ServiceModel*, it is more logical to consider a single instance playing multiple roles. This improvement is already implemented by the OWL-S coalition.

In our Core Ontology of Services, we went further to separate the functionality, process and software aspects of a service loaded onto the single concept of *Service* in OWL-S. It replaces *Service* with the concept of different kinds of Service Descriptions, which are S-Descriptions (a context in D & S) that envision a process as well as certain roles related to the individual tasks of the process. Inputs, outputs and abstract tools used to carry out a certain task are examples of roles. In case of information services, inputs and outputs are played by information objects and tools are played by particular software implementations.

While this definition of a Service Description may not be the only one, the fact that it is formulated according to a foundational ontology allows to compare it to alternative definitions and foster discussion on alternative conceptualizations of a Web Service.

5.2 Increased Axiomatization

A key advantage of the alignment to a foundational ontology is that it prompts the engineer to take a stance with respect to the principles established by the foundational ontology. What is typically gained is an increased understanding of one's own ontology and a richer axiomatization through ties to the foundational ontology. DOLCE mitigates the danger of overcommitment in this process (imported theories that are not used or not shared by the engineer) by extensive modularization along world views (3D, 4D, etc.) and domains (law, finance, etc.).

As an example, in our Core Ontology of Services we have made use of an Ontology of Plans which includes subtypes of the generic Task concept for a detailed modelling of plans or process models. These constructs are directly comparable to the control constructs of OWL-S, but provide a higher level of axiomatization. An example of such types is DOLCE's *Synchro-Task* whose OWL definition is depicted in the Appendix. It matches the concept of "join" in the "Split-Join" control construct from OWL-S. A synchronization task is typically used to bind the execution of a "planning" activity rather than of a domain activity, since the referred activity is supposed to re-synchronize a process when it waits for the execution of two or more concurrent (or partly concurrent) activities.

Higher axiomatization is partly possible by the natural linkage to the Ontology of Time, another module for DOLCE, for describing (constraints on) temporal relations between process elements when they are executions of a plan. OWL-S would also need such an Ontology of Time and then it would be natural to adopt or reference an existing ontology instead of creating an ontology from scratch.

The Ontology of Plans also allowed to align relations such as *owl-s:components*, which is used to relate control constructs to their components. In OWL-S this relation is described merely as a subrelation of *owl:Property* with a domain of *ControlConstruct*. In our work, we aligned this relation to the *temporary-component* relation in DOLCE. The latter has a firm foundation as a subrelation of the more basic *component* (*functional proper-part*) mereological relation and *partly-compresent-with* temporally indexing relation, both characterized with formal restrictions on its application to other basic concepts, such as *Object*, *Description*, *Event*, etc.

5.3 Improved design

In our work we propose to complement modularization in OWL-S with contextualization as a design pattern. Contextualization allows us to move from a monolithic process description of a service to the representation of different, possibly conflicting views with various granularity. The Descriptions & Situations ontology provides us the basic primitives of context modelling such as the notion of roles, which allows us to talk of inputs and outputs on the abstract level, i.e. independent of the objects that play such roles.

Using this pattern results in a much more intuitive representation of attribute binding, with clearly defined semantics and scoping provided by Descriptions & Situations. Inputs and outputs can be modelled as Functional Roles (more precisely: Instrumentality Roles), which serve as variables in our ontology. A single endurant — for example, a physical book — can play multiple roles within the same or different descriptions and thus it is natural to express that the given book is output with respect to one process, but input to another. Moreover, it is easier to represent the requirement that the input of a process *has to be* played by the same instance as the output of another process by putting constraints on the *objects* (and not the process or task) which play these roles (however, the expressivity required is the same and therefore goes beyond the power of OWL).

Besides a more intuitive representation, Functional Roles as components have an explicit scope, namely the S-Descriptions they belong to. Although not addressed in the present work, clearly defined limits in scope are necessary to describe semantic relationships among (service) descriptions, e.g. to talk of conflicts between descriptions.

5.4 Wider scope

As we have seen before, Web Services exist on the boundary of the world inside an information system (Infolandia) and the external world. Except for the rare case of a pure information service, Web Services carry out operations to *support* a real world service. Functionality, which is an essential property of a service, then arises from the entire process that comprises computational as well as real wold activities.

Web Service descriptions are thus necessarily descriptions of two parallel worlds. In Infolandia, the world consist of software manipulating (representations of) information objects. Activities are sequenced by computational processes. Meantime in the real world books are being delivered to their destinations.

The connection between these worlds is that some of the information objects in Infolandia represent real world objects. Also,

computational activities comprise part of the service execution in the real world. For example, an order needs to be entered by the Web agent into an information system, so that the warehouse knows which books to deliver to a given address.

The distinction between information objects, events and physical ones is not explicitly made in OWL-S. [14] Nevertheless, we believe that this distinction is important for disambiguating the nature of services in an open environment such as the Semantic Web.[15] In our work this separation naturally follows from the use of the DOLCE+ foundational ontology, where the distinction is an important part of the characterization of concepts. In particular, it makes possible to be more precise about the kinds of relationships that can occur among objects or between objects and events.

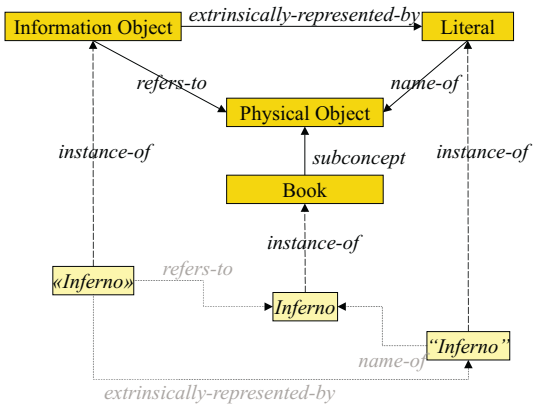

Figure 10: Information Objects in DOLCE

For example, using DOLCE+ we can distinguish between a physical object (such as a book), an information object (such as the name of a book) and a representation of such information using a particular description system (e.g. a string encoding). The relations provided by DOLCE are shown in Figure 10.

The reader may note that by building on the Descriptions & Situations ontology design pattern, our work naturally extends OWL-S with the representation of service situations. Service situations in our work correspond to a possible execution of a service. The description of service executions is already considered by the OWL-S coalition for the purposes of service execution monitoring. We believe that this direction should also be pursued by OWL-S as service requests are often formulated in terms of actual values of input/output parameters (or relatively narrow sets of parameter values). For example, customers of bookshops often have a clear idea of which book they want to buy or at least what kind of book it is. One could imagine an intelligent matching engine that in such case returns only services that offer a particular book or a category of books, instead of returning all known book selling services.

[14]Based on the examples so far, one may conjecture that OWL-S inputs and outputs concern physical objects relating to information objects such as message parts in WSDL through grounding.

[15]In fact, the lack of this distinction stands behind the emergence of the 'Semantic Web identity crisis' that results from the ambiguous use of identifiers in Semantic Web ontology languages such as RDF [16]. In practice a URI can be used to reference a document on the Web, to reference (a fragment of) a document containing some definition of a concept or to represent a concept (without any intended reference to an actual location on the Web). Unfortunately, no standard scheme exists to distinguish between the three kinds of identifiers even though they need to be resolved in different ways.

6. CONCLUSION

The paper identified several problematic aspects of OWL-S and suggested possible improvements by an alignment to a foundational ontology. We used a stack of ontologies for the alignment made up of DOLCE, Descriptions & Situations as well as the Core Ontology of Services. Note, that the alignment is not dependent on DOLCE, because Descriptions & Situations may be aligned to any foundational ontology. Parts of the service description that deal with service quality and assessment are left for future work.

Our exercise of giving an ontological foundation to OWL-S is useful both for better understanding OWL-S and enriching it with additional formal semantics. We see the presented results as an example for the benefits of alignment to foundational ontologies as our methodology is applicable also to other standards. As a matter of fact, our Core Ontology of Services can be applied as a framework for harmonizing the ongoing efforts to characterize Web Services, because it does not commit to a specific software design reference framework, and it is based on a generic, social notion of service. For example, the ontology of the Web Services Architecture (WSA) Working Group of the W3C, as well as other interesting methods for Web Service deployment, such as problem-solving methods [13], can be interpreted (aligned, harmonized, or made interoperable) according to our reusable ontological components.

The alignment of OWL-S to the Core Ontology of Services also means that Web Services described in OWL-S are automatically aligned to DOLCE. Such descriptions can be further enriched by adding DOLCE-based semantics (for example, spatio-temporal relations) to the domain concepts involved. We imagine this would allow a sufficiently sophisticated matching or composition engine to reason with the additional semantics in order to provide more targeted matches as a result. However, building such a tool is beyond the scope of our work.

One of the difficulties we encountered with our method of ontology alignment was that it required us to understand to some extent the principles of the foundational ontology. These principles stem from other sciences (philosophy, psychology, semiotics, communication theory etc.), which means that a (re)engineering of this kind requires a considerable intellectual investment from the knowledge engineer at the moment. We think, however, that this investment, materialized in the Core Ontology of Services, will pay off whenever new (Web) Service ontologies are to be aligned or when a Web Service ontology should communicate with domain ontologies (e.g. in the CongoBuy case) or with workflow ontologies of the service actors, or even in the matching and composition of services that have overlapping domains or tasks. Such a pay off also shows why we have not just taken a reusable ontology, but a foundational one: a reusable ontology could have been used to carry out an analysis of OWL-S, but in order to gain access to conceptual alignment with other service, domain or task ontologies reusability is not enough: we need an appropriate and flexible foundational ontology.

Nevertheless, we will try to make the foundational ontology and the alignment process more accessible in the future. We also believe that next-generation ontology editors will help knowledge engineers to deal with the complexity of rich ontologies by incorporating the idea of ontology design patterns and offering more support for modularization.

Similarly, editors designed specifically for authoring Web Service descriptions will hopefully take away much of the existing burden of creating Web Service descriptions from future Web engineers.

Acknowledgements. This work is financed by WonderWeb, an EU Information Society Technologies (IST) funded project IST-2001-33052 (`http://wonderweb.semanticweb.org`).

7. REFERENCES

[1] A. Ankolekar, F. Huch, and K. Sycara. Concurrent Execution Semantics for DAML-S with Subtypes. In I. Horrocks and J. A. Hendler, editors, *1st Int. Semantic Web Conference (ISWC), Proceedings*, volume 2342 of *LNCS*. Springer, 2002.

[2] Z. Baida, H. Akkermans, and J. Gordijn. Serviguration: Towards Online Configurability of Real-World Services. In *Proceedings of the Fifth International Conference on Electronic Commerce (ICEC03)*, pages 111–118, Pittsburgh, PA, 2003. ACM.

[3] Z. Baida, J. Gordijn, and H. Akkermans. A Shared Terminology for Online Service Provisioning, 2004. Available via `http://www.cs.vu.nl/~ziv`.

[4] S. Borgo, A. Gangemi, N. Guarino, C. Masolo, and A. Oltramari. Ontology RoadMap. WonderWeb Deliverable D15, Dec 2002. `http://wonderweb.semanticweb.org`.

[5] E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana. Web services description language (WSDL). `http://www.w3.org/TR/wsdl`, Mar 2003. W3C Note.

[6] T. Elrad, R. E. Filman, and A. Bader. Aspect-oriented programming: Introduction. *Communications of the ACM*, 44(10):29–32, October 2001.

[7] A. Gangemi and P. Mika. Understanding the semantic web through descriptions and situations. In *DOA/CoopIS/ODBASE 2003 Confederated International Conferences DOA, CoopIS and ODBASE, Proceedings*, LNCS. Springer, 2003.

[8] A. Gangemi, P. Mika, M. Sabou, and D. Oberle. An Ontology of Services and Service Descriptions. Technical report, Laboratory for Applied Ontology, National Research Council, I-00137 Rome, Italy, Nov 2003. `http://www.isib.cnr.it`.

[9] R. M. MacGregor. Using a description classifier to enhance deductive inference. In *Proceedings Seventh IEEE Conference on AI Applications*, pages 141–147, 1991.

[10] D. Martin, M. Burstein, G. Denker, J. Hobbs, L. Kagal, O. Lassila, D. McDermott, S. McIlraith, M. Paolucci, B. Parsia, T. Payne, M. Sabou, E. Sirin, M. Solanki, N. Srinivasan, and K. Sycara. OWL-S 1.0 draft release. `http://www.daml.org/services/owl-s/1.0/`, Dec 2003.

[11] C. Masolo, S. Borgo, A. Gangemi, N. Guarino, and A. Oltramari. Ontology Library (final). WonderWeb Deliverable D18, Dec 2003. `http://wonderweb.semanticweb.org`.

[12] D. L. McGuinness and F. van Harmelen. Web ontology language (OWL) overview. `http://www.w3.org/TR/owl-features/`, Feb 2004. W3C Recommendation.

[13] E. Motta, J. Domingue, L. Cabraland, and M. Gaspari. IRSII: A Framework and Infrastructure for Semantic Web Services. In *The SemanticWeb - ISWC 2003*, volume 2870 of *LNCS*, pages 306 – 318. Springer, 2003.

[14] S. Narayanan and S. McIlraith. Analysis and simulation of Web Services. *Computer Networks*, 42(5):675–693, 2003.

[15] A. Oltramari, A. Gangemi, N. Guarino, and C. Masolo. Sweetening ontologies with DOLCE. In A. Gómez-Pérez and V. R. Benjamins, editors, *Knowledge Engineering and Knowledge Management. Ontologies and the Semantic Web, 13th International Conference, EKAW 2002, Siguenza, Spain, October 1-4, 2002, Proceedings*, volume 2473 of *Lecture Notes in Computer Science*. Springer, 2002.

[16] S. Pepper and S. Schwab. Curing the Web's Identity Crisis. Technical report, Ontopia (`http://www.ontopia.net`), 2003.

[17] D. Richards and M. Sabou. Semantic markup for semantic web tools: A DAML-S description of an RDF-store. In *Proceedings of the 2nd Int. Semantic Web Conference (ISWC2003)*, volume 2870 of *LNCS*, pages 274–289. Springer, Sep 2003.

APPENDIX

A. DOLCE+'S SYNCHRO-TASK IN OWL ABSTRACT SYNTAX

```
Class(<dolce+:synchro-task> partial
  <dolce+:elementary-task>
  restriction(<dolce+:predecessor>
   someValuesFrom (unionOf(
    <dolce+:concurrent-task>
    <dolce+:partly-concurrent-task>)))
  restriction(<dolce+:direct-predecessor>
   minCardinality(2))
  restriction(<dolce+:sequences>
   allValuesFrom
   (dolce+:planning-activity))
  restriction(<dolce+:represented-by>
   allValuesFrom (<dolce+:join-node>))
  annotation(<rdfs:label> "synchro-task")
  annotation(<rdfs:comment>
   "A task that joins a set of
    tasks after a branching"))
```