Making the Difference in Semantic Web Service Composition

Freddy Lécué[1,2] and **Alexandre Delteil**[1]

[1]France Telecom R&D, France
4 rue du clos courtel, F-35512 Cesson Sévigné
{(firstname.lastname)@orange-ftgroup.com}

[2] École Nationale Supérieure
des Mines de St-Etienne, France
158, cours Fauriel, F-42023 Saint-Étienne

Abstract

Automation of Web service composition is one of the most interesting challenges facing the Semantic Web today. In this paper we propose a mean of performing automated Web service composition by exploiting semantic matchmaking between Web service parameters (i.e., outputs and inputs) to enable their connection and interaction. The key idea is that the matchmaking enables, at run time, finding semantic compatibilities among independently defined Web service descriptions. To this end, our approach extends existing methods in order to explain misconnections between Web services. From this we generate Web service compositions that realize the goal, satisfying and optimizing the semantic connections between Web services. Moreover a process of relaxing the hard constraints is introduced in case the composition process failed. Our system is implemented and interacting with Web services dedicated on a Telecom scenario. The preliminary evaluation results showed high efficiency and effectiveness of the proposed approach.

Introduction

The semantic web is considered by many to be the future of the current web (Berners-Lee, Hendler, & Lassila 2001). Resources in the semantic web are enhanced using rich description languages such as the Web Ontology Language (OWL) . Intelligent software agents can, in turn, use these descriptions to reason about web resources and automate their use to accomplish goals including intelligent composition and discovery.

Semantic web services are web services that have been enhanced with formal semantic descriptions. Semantic descriptions are expressed with ontologies and description logics (Baader & Nutt 2003) which enable automated discovery, selection, composition and invocation of services. An ontology is a formal conceptualization of a particular domain, useful for describing the semantics of any web service. From this, a description is created using concepts from that domain, properties of those concepts, and relationships between concepts.

Web service composition enhanced by semantic technologies is currently one of the most hyped and addressed issue in the Service Oriented Computing. Starting from an initial set of Web services, Web service composition aims at selecting and inter-connecting Web services provided by different partners according to a goal to achieve. Previous work in this regard has presented various approaches to automate composition, from use of HTN (Sirin *et al.* 2004), Linear logic (Rao, Kungas, & Matskin 2003), classic AI planning (Pistore *et al.* 2004) to Situation calculus (McIlraith & Son 2002). Most of the work has focused on two main levels of composition: functional and process levels. At functionnal level semantic connections between Web services is considered as the main component to provide suitable composition of Web services semantically well-chained. However previous work mainly focused on the *standard* semantic matching functions introduced by (Paolucci *et al.* 2002) to value the semantic connections between Web services. In this way Web service composition is reduced as a partial order of basic semantic connections where the connections are still not robust enough to support the composition. In this paper we suggest to point out the limits of the *standard* semantic matching functions to solve Web service composition. Moreover we overcome these limits by describing and studying their enhancements.

The rest of the paper is organized as follows. Section 2 presents the scenario where we integrate the suggested approach. In section 3 we introduce the notion of *causal links* and compare service composition as a composition of *causal links*. Section 4 presents the limits of the *standard* semantic matching functions and describe their extensions for service composition. Section 5 explains how to relax a composition problem in case no composition is retrieved. Section 6 briefly comments on related work. Finally section 7 draws some conclusions and talk about possible future directions.

A Motivating Use Case

Our approach is being exploited in the VoIE Project, conceived by a Telecom operator. The motivation is to give to the end user the possibility to create their own commercial offers according to their real needs, without any kind of assistance. The end user is only in charge of selecting offer(s) she wants to subscribe. For instance the offers may be as follows, ADSL eligibility, LiveBox, Voice over IP, Visiophone, IPTV (TV over IP), Voice Messaging, HDTV. Since each offer is interfaced by a semantic Web service (e.g., IPTVService interfaces the IPTV offer), the goal is to provide a correct composition of the latter services in regard to the semantic connection between those services. Figure 1 shows a part of the $\mathcal{ALE}$ Ontology (105 concepts and 37 properties) used to describe the domain i.e., concepts of the ontology refer to all input and output parameters of services. Such a conceptualization enables to model semantic connection between Web service parameters. Here we focus on this last aspect to provide application examples of the approach. $\mathcal{ALE}$ is an interesting tradeoff between expressivity and complexity.

In the example, we consider four Web services i.e., a subset of the 29 Web services included in the real scenario:

- `AdslEligibility`, that, starting from a `PhoneNum`, a `ZipCode` and an `Email` address, returns the `NetworkConnection` of the desired zone;
- `VoiceOverIP`, that, starting from a `PhoneNum` and a `SlowNetworkConnection`, returns the `VoIPId` of the ADSL line a Telecom operator needs to install the line;
- `TvOverIP`, that, starting from a `PhoneNum` and a `FastNetworkConnection`, returns a serial number of a `VideoDecoder` required to access video over IP;
- a `LiveBox` service returns the `Invoice` of the commercial offer the user requested, depending on a `PhoneNum`, `IPAddress` and serial number of a `Decoder`.

In such a scenario automation of Web service composition is a real issue for Telecom operator since the number of offers i.e., services the user can choose is more and more increasing. Indeed the more offers the harder the composition will be. That is why we suggest to compose automatically Web services depending on the user requirements (i.e., the commercial offer she subscribed) and the Web services compatibilities (i.e., through their semantic connections).

$$
\begin{aligned}
&Offer \equiv \forall priceOffer.Price \sqcap \forall interfacedBy.Service\\
&Commercial\_offer \equiv \forall comOffer.Offer\\
&NetworkConnection \equiv \forall netSpeed.Speed\\
&SlowNetworkConnection \equiv NetworkConnection \sqcap\\
&\qquad\qquad \forall netSpeed.Adsl1M\\
&FastNetworkConnection \equiv NetworkConnection \sqcap\\
&\qquad\qquad \forall netSpeed.AdslMax\\
&Adsl1M \equiv Speed \sqcap \forall mBytes.1M\\
&AdslMax \equiv Speed \sqcap \forall mBytes.Max\\
&Max \sqsubseteq 1M\\
&ZipCode \sqsubset \top, EMail \sqsubset \top, Address \sqsubset \top, Invoice \sqsubset \top\\
&IPAddress \equiv Address \sqcap \forall protocol.IP\\
&VoIPId \equiv Address \sqcap \forall network.FTLocal\\
&VideoDecoder \equiv Decoder \sqcap \forall decrypt.Video
\end{aligned}
$$

Figure 1: Part of the $\mathcal{ALE}$ domain ontology $\mathcal{T}$.

Causal Links Composition

At functional level semantic connections between services is considered as the main issue to form new value-added Web services. These connections are necessary to semantically link output to input parameters of Web services. In this way a composition is defined as a plan of services wherein all Web services are semantically well ordered and well linked.

Semantic Web Context

Parameters (i.e., input and output) of Web services in semantic Web referred to concepts in an ontology $\mathcal{T}$ (e.g., Figure 1) through the OWL-S profile (Ankolenkar *et al.* 2004), WSMO capability (Fensel *et al.* 2005), or SA-WSDL (Sivashanmugam *et al.* 2003)). To retrieve a semantic similarity between an output parameter $Out\_s_y \in \mathcal{T}$ of a service s_y and an input parameter $In\_s_x \in \mathcal{T}$ of a service s_x is similar to find a mapping (Küsters 2001) between two knowledge representations encoded using $\mathcal{T}$. A causal link[1] (Russell & Norvig 1995) describes this semantic matchmaking between these two parameters. Thereby s_x and s_y are partially linked according to a matchmaking function $Sim_{\mathcal{T}}(Out\_s_y, In\_s_x)$ where $Sim_{\mathcal{T}}$ describes the match type of its two parameters. $Sim_{\mathcal{T}}$ is used to value and order services connections since different output parameters of service can be matched with a same input parameter of another service. $Sim_{\mathcal{T}}(Out\_s_y, In\_s_x)$ is clearly analogous to $degreOfMatch(Out\_s_y, In\_s_x)$ of (Paolucci *et al.* 2002).

According to (Lécué & Léger 2006) a causal link is refined as a triple $\langle s_y, Sim_{\mathcal{T}}(Out\_s_y, In\_s_x), s_x\rangle$ such that s_x and s_y are the two services involved in the connection. A causal link $\langle s_y, Sim_{\mathcal{T}}(Out\_s_y, In\_s_x), s_x\rangle$ implies that a) s_y precedes s_x since an output of s_y is consumed by an input of s_x, b) no service is interleaved between s_x and s_y.

Since a composition of Web services consists of a partial order of Web services wherein these services are semantically chained by causal links, we suggest to study the Web service composition as a causal links composition.

Limits and Extension of Semantic Matching

The function of matchmaking $Sim_{\mathcal{T}}$ enables to sketch causal links and express which matching type is employed to chain Web services. Even so, in many Web service composition models (Zhang, Arpinar, & Aleman-Meza 2003; Sirin *et al.* 2004; Cardoso & Sheth 2003; Rao, Kungas, & Matskin 2003) this function is often reduced to the four well known matchmaking functions introduced by (Paolucci *et al.* 2002) with the extra match level Intersection of (Li & Horrocks 2003):

- **Exact** If the output parameter Out_s_y of s_y and the input parameter In_s_x of s_x are equivalent concepts; formally, $\mathcal{T} \models Out\_s_y \equiv In\_s_x$.
- **PlugIn** If Out_s_y is sub-concept of In_s_x; formally, $\mathcal{T} \models Out\_s_y \sqsubseteq In\_s_x$.
- **Subsume** If Out_s_y is super-concept of In_s_x; formally, $\mathcal{T} \models In\_s_x \sqsubseteq Out\_s_y$.
- **Intersection** If the intersection of Out_s_y and In_s_x is satisfiable; formally, $\mathcal{T} \not\models Out\_s_y \sqcap In\_s_x \sqsubseteq \perp$.
- **Disjoint** Otherwise Out_s_y and In_s_x are incompatible i.e., $\mathcal{T} \models Out\_s_y \sqcap In\_s_x \sqsubseteq \perp$.

[1] In AI planning area, some authors call causal link **protection intervals** (McAllester & Rosenblitt 1991).

Definition 1. ***(Valid Causal link)***
A causal link $\langle s_y, Sim_{\mathcal{T}}(Out\_s_y, In\_s_x), s_x\rangle$ is valid iff $Sim_{\mathcal{T}}(Out\_s_y, In\_s_x) \neq Disjoint$ i.e., $\mathcal{T} \not\models Out\_s_y \sqcap In\_s_x \sqsubseteq \bot$.

A valid causal link between two Web services describes a potential match (Colucci *et al.* 2005) between two parameters, hence a potential and link between these two services. A valid composition is consisted of only valid causal links.

Example 1. ***(Valid Causal link)***
Let VideoDecoder be the output parameter of the service TvOverIP. This parameter is subsumed by the input parameter Decoder of the service LiveBox according to the ontology (Figure 1), hence a valid causal link $\langle TvOverIP,\ Sim_{\mathcal{T}}(VideoDecoder,\ Decoder),\ LiveBox\rangle$ since $\mathcal{T} \models VideoDecoder \sqsubseteq Decoder$.

Limits of Semantic Matching

These five match levels are far from enough to bring Web service composition as a causal links composition to its full potential. The Exact match is clearly appropriate to chain two Web service parameters since they refer to equivalent concepts. The PlugIn match is also a possible match to plug an output parameter in an input parameter of another Web service since the output is more specific than the input parameter. The Disjoint match informs about the incompatibility of two Web service parameters. Even if the matchmaking Exact, PlugIn, and Disjoint can be used without change to value causal links in a Web service composition, the match levels Intersection and Subsume need some refinements to be fully efficient for causal links composition. Suppose the `AdslEligibility` and `VoiceOverIP` services with their causal link $\langle AdslEligibility,\ Sim_{\mathcal{T}}(NetworkConnection,\ SlowNetworkConnection), VoiceOverIP\rangle$ valued by the Subsume match. It is obvious that such a causal link should not be directly applied in a Web service composition since the `NetworkConnection` is not specific enough to be consumed by the input parameter `SlowNetworkConnection`. Thus the output parameter `NetworkConnection` needs *Extra Description* to enable a composition of these two services. A causal link valued by the Intersection match needs a comparable refinement.

Definition 2. ***(Robust Causal link)***
A causal link $\langle s_y, Sim_{\mathcal{T}}(Out\_s_y, In\_s_x), s_x\rangle$ is robust iff $Sim_{\mathcal{T}}(Out\_s_y, In\_s_x)$ is either Exact or PlugIn.

Property 1. ***(Robust Web Service Composition)***
A composition is robust iff all its causal links are robust.

Extension of Semantic Matching Functions

Another possible way to state the problem we obtain for causal link $\langle s_y, Sim_{\mathcal{T}}(Out\_s_y, In\_s_x), s_x\rangle$ valued by the Intersection and Subsume matchmaking is to find the information contained in the input parameter In_s_x and not in the output parameter Out_s_y. To do this, we use a non standard operation in description logics, the difference or subtraction operation introduced by (Brandt, Kusters, & Turhan 2002) for comparing DL descriptions and we adapt it to the problem of semantic matching between Web service parameters.

The difference operator, first introduced by (Teege 1994), enables to remove from a given description all the information contained in another description. The difference between two concept descriptions C and D with $C \sqsubseteq D$ is

$$C - D := \max_{\sqsupseteq}\{B | B \sqcap D \equiv C\} \tag{1}$$

This definition requires that the second argument subsumes the first one. However the difference $C - D$ between two incomparable descriptions C and D (i.e., C is not subsumed by D) can be given by constructing the least common subsumer of C and D, that is, $C - D := C - lcs(C, D)$. (Brandt, Kusters, & Turhan 2002) proposed a refinement of this definition (1) by taking the syntactic minimum (w.r.t a subdescription ordering $\preceq_d$ (Küsters 2001)) instead of a semantic maximum. Thus they defined the difference between two (in)comparable concept descriptions C and D as (2)

$$C \backslash D := \min_{\preceq_d}\{B | B \sqcap D \equiv C \sqcap D\} \tag{2}$$

Even if (1) captures the real semantic difference between two concept descriptions, (2) has two main advantages. Firstly it does not contain redundancies in its result and secondly it is more readable by a human user. In the following we suggest to use the second definition since it is also defined for $\mathcal{ALE}$ descriptions logics.

The idea behind our approach is the following. In case a causal link $\langle s_y, Sim_{\mathcal{T}}(Out\_s_y, In\_s_x), s_x\rangle$ is valid (not valued by a Disjoint matchmaking) but not robust (i.e., valued by neither an Exact nor a PlugIn matchmaking), we compare Out_s_y and In_s_x to obtain two kinds of information, a) the *Extra Description* $In\_s_x \backslash Out\_s_y$ that refers to the information required but not provided by Out_s_y in order to semantically link it with the input In_s_x of s_x, b) the *Common Description* $Out\_s_y \sqcap In\_s_x$ that refers to the information required by In_s_x and effectively provided by Out_s_y.

The considered causal links are links valued by either a Subsume or an Intersection matching. In the Subsume case we computes the *Extra Description* B contained in In_s_x such that the matching between $B \sqcap Out\_s_y$ and In_s_x is Exact. By (2) this *Extra Description* $In\_s_x \backslash Out\_s_y$ is defined by $\min_{\preceq_d}\{B | B \sqcap Out\_s_y \equiv In\_s_x\}$ since $Out\_s_y \sqsupseteq In\_s_x$. In case the causal link is valued by an Intersection match (i.e., $\neg(Out\_s_y \sqcap In\_s_x \sqsubseteq \bot)$) we computes the *Extra Description* B that is not specified in Out_s_y to reach a PlugIn match between $B \sqcap Out\_s_y$ and In_s_x.

Example 2. **(Extra Description*)***
Let $\langle VoiceOverIP, Sim_{\mathcal{T}}(VoIPId, IPAddress), LiveBox\rangle$ be a causal link valued by an Intersection match since $\mathcal{T} \not\models VoIPId \sqcap IPAddress \sqsubseteq \bot$. This link is not robust enough (definition 2) to be applied in a composition. The description missing in VoIPId to be plugged in the input parameter IPAddress is referred by the Extra Description *i.e., $IPAddress \backslash VoIPId$ i.e., $\forall protocol.IP$ according to (2).*

We illustrated the rationale of our approach by computing what is needed in order to replace a non robust causal link by its robust form. In particular, we change an Intersection by a PlugIn match, and a Subsume by an Exact match in order to obtain robust causal links. We could also consider other

Match Type	Exact	PlugIn	Subsume	Intersection
Exact	-	✗$_{ii)}$	✗$_{ii)}$	✗$_{i),ii)}$
PlugIn	✗$_{i)}$	-	✗$_{i),ii)}$	✗$_{i)}$
Subsume	✓	✗$_{ii)}$	-	✗$_{ii)}$
Intersection	✗$_{i)}$	✓	✗$_{i)}$	-

Table 1: Substitution of Match Levels for Composition.

substitutions of matchmaking functions e.g., find a way to change a Subsume by a PlugIn match, or an Intersection by an Exact match and so on. However these substitutions are out of interest in Web service composition since

i) some substitutions required the computation of $Out\_s_y \backslash In\_s_x$, e.g., from PlugIn to Exact;

ii) some others are not relevant because they implied a loss of matchmaking quality, e.g., from Subsume to PlugIn.

Table 1 summarizes these different levels of substitution. Suppose the substitution of a PlugIn by an Exact match in order to improve a causal link valued by a PlugIn match level. The case under consideration is i) since we have to compute B' such that $Out\_s_y \equiv B' \sqcap In\_s_x$. B' is defined by $Out\_s_y \backslash In\_s_x$ to model the exact match. Unfortunately the description B' can not be added to In_s_x since input parameters of services are supposed static without possible alteration. In the opposite output parameters of services may be enhanced by some *Extra Descriptions* ($Out\_s_y \sqcap B$) in order to be chained with input parameters of other services.

Suppose the case ii) wherein the Subsume match is replaced by a PlugIn match. Consequently, the *Extra Description* B' is computed as $B' \sqsubset B$ such that B is defined by $In\_s_x \backslash Out\_s_y$. By the way $B' \sqcap Out\_s_y \sqsubseteq In\_s_x$ whereas $B \sqcap Out\_s_y \equiv In\_s_x$. It is obvious that B is more appropriate than B'. The former enables an Exact match whereas the latter changes the Subsume by the PlugIn match.

The ultimate goal of Web service composition is to provide a composition wherein causal links are robust. In case some causal links $\langle s_y, Sim_{\mathcal{T}}(Out\_s_y, In\_s_x), s_x \rangle$ are not robust enough but valid, we are able to compute an *Extra Description* from In_s_x in order to substitute the previous link by its robust form. In other words causal links valued by a Subsume or an Intersection match level move to robust causal links in case this *Extra Description* is provided. The latter description is essential to solve a service composition in case some causal links are not robust. The *Extra Description* returned by difference (2) is not only necessary to explain where a causal link composition may fail but also why a causal link failed and how to improve it. A composition failure is due to non robust causal links since the matchmaking between Web services parameters is not robust enough.

Relaxing Constraints on Service Composition

The automation of service composition is one of most challenging and complex task facing the Semantic Web. However the complete automation is still not a reality, especially when a composition comprises non robust causal links.

An intuitive method to immediately retrieve the *Extra Description* consists in discovering which services are able to return the *Extra Description* as a conjunction of some of their output parameters. Let $s_{i,1 \leq i \leq n}$ be services with some of their respective output parameters $Out\_s_{i,1 \leq i \leq n}$ and $\langle s_y, Sim_{\mathcal{T}}(Out\_s_y, In\_s_x), s_x \rangle$ a non robust causal link. In case the conjunction of Out_s_y and $Out\_s_{i,1 \leq i \leq n}$ satisfies the *Extra Description*, this conjunction can be semantically linked to In_s_x to form a robust causal link. Each input parameter of these discovered services has to be either known at run time or linked to an output parameter of another service through a robust causal link. A constraint of this method is related to the complexity of composition. The more non robust links in composition the more important the cardinality of services will be implied in the composition.

Another conceivable method is to replace non robust causal links with their robust forms as presented in the previous section. The composition process is still automated in case the *Extra Description* of a non robust causal links is automatically retrieved. This is still possible if there exists a service that returns this *Extra Description*. Otherwise there is no way to automatically retrieve the *Extra Description*.

Consequently, the latter description has to be retrieved by relaxing some constraints during the composition process since all the available description does not guarantee to find a solution. Relaxing constraints and obtaining a composition of robust causal links is an interesting trade-off to reach composition. Such constraints still guarantee the original feasible solutions and yield additional feasible solutions. Constraints $B_{i,1 \leq i \leq n}$ refer to the *Extra Descriptions*, which able to change non robust causal links in their robust forms. More formally the set of relaxing constraints $\mathcal{B}$ is expressed by definition 3 where $\langle s_y, Sim_{\mathcal{T}}(Out\_s_y, In\_s_x), s_x \rangle$ refers to causal links in the composition model.

Definition 3. ***(Set of Relaxing Constraints)***
The set of relaxing constraints $\mathcal{B}$ is defined by

$$\inf_{\sqsubseteq}\{In\_s_x \backslash Out\_s_y | \langle s_y, Sim_{\mathcal{T}}(Out\_s_y, In\_s_x), s_x \rangle \\ \qquad\qquad is\ a\ valid\ causal\ link\} \backslash_{set} \{\top\}$$

Roughly spoken, the set of relaxing constraints $\mathcal{B}$ of a Web service composition is defined as being the set of descriptions able to change non robust causal links into their robust forms. $\mathcal{B}$ gathers the most specific descriptions of the set $\{In\_s_x \backslash Out\_s_y\}$ where $\langle s_y, Sim_{\mathcal{T}}(Out\_s_y, In\_s_x), s_x \rangle$ is a valid causal link. The latter consideration implies that a same description (i.e., the most specific) can be used by a finite set of non robust causal links to change them in their robust forms. The descriptions used to perform these changes will be the most specific descriptions of *Extra Description* from non robust links. $\mathcal{B}$ does not only explain why the composition process failed but also gives a solution of the robustness problem of causal links hence a way to reach semantic service composition.

Property 2. ***(Constraints for Robust Causal Links)***
The set of relaxing constraints $\mathcal{B}$ of a Web service composition with only robust causal links is the empty set.

Proof. Let a service composition constituted of only robust causal links $cl_{i,1 \leq i \leq n}$, $\langle s_y, Sim_{\mathcal{T}}(Out\_s_y, In\_s_x), s_x \rangle_i$. By definition 2, the match level between Out_s_y and In_s_x is either Exact or PlugIn, hence $Out\_s_y \equiv In\_s_x$ or $Out\_s_y \sqsubseteq In\_s_x$. By difference (2) we obtain in the two cases that $In\_s_x \backslash Out\_s_y \equiv \top$ i.e., $\mathcal{B}$ is defined by the empty set. □

Once the set of *Extra Descriptions* is retrieved through $\mathcal{B}$, this set is suggested to the end user in order to be relaxed. Such a method has the advantage of relaxing constraints on the end user. This user is then responsible of providing the *Extra Description* that the system needed to elaborate the composition. In the motivating example, services and user's requirements are both not specified enough to highlight the *Extra Description* and constraints relaxing advantages. In real scenarios Web service composition often requires some relaxing constraints to turn in automated composition.

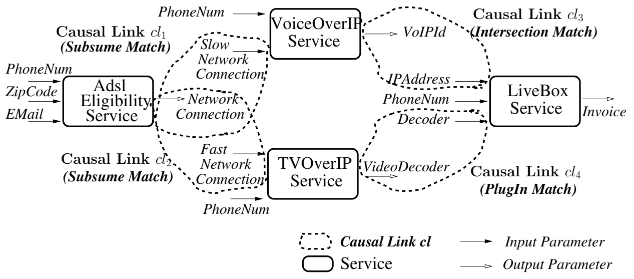

Figure 2: A Web Service Composition and its Causal Links.

Example 3. *(***Relaxed Causal links***)*
The motivating example exposes a Web service composition through 4 causal links $cl_{i,1 \leq i \leq 4}$ (Figure 2). Three of the four valid causal links are not robust i.e., $cl_{i,1 \leq i \leq 3}$. No Web service may provide the Extra Description necessary to form robust causal links. A Relaxing constraints has to be applied to obtain a composition of robust causal links. The discovery of the Extra Descriptions through $\mathcal{B}$ gives directions to obtain robust causal links. According to the definition 3 $\mathcal{B}$ is constituted of an union of three differences in DL i.e., the difference between the concepts 1) SlowNetworkConnection and NetworkConnection to change cl_1 by a causal link valued by an Exact match, 2) FastNetworkConnection and NetworkConnection to change cl_2 in the same way as cl_1, 3) IPAddress and VoIPId to replace cl_3 by a causal link valued by a PlugIn match. Since cl_4 is a robust causal link and $\forall netSpeed.AdslMax \sqsubset \forall netSpeed.Adsl1M$, $\mathcal{B}$ is defined by $\{\forall netSpeed.AdslMax, \forall protocol.IP\}$.

The Web service composition is automatically retrieved in case the *Extra Description* is provided by the end user, other services or any third party, depending on the application we want to automate. For instance $\forall netSpeed.AdslMax$, $\forall protocol.IP$ have to be provided in case we want to automate the composition process of the motivating example.

A Semantic Web Service Composition Approach

The composition approach used and extended in this paper is based on the model introduced by (Lécué & Léger 2006), but can be applied with many other approaches (Related work). Their AI planning-oriented composition Ra_4C performs a regression-based approach and returns a set of correct, complete and consistent plans wherein services are actions semantically linked by causal links. However some of their solutions may refer to non robust compositions since some causal links can be valued by a Subsume match level. Algorithm 1 extends Ra_4C by studying and evaluating *Extra Descriptions* as long as the composition process proceeds.

Algorithm 1: Robust Web Service Composition.

```
Input: A composition process $Ra_4C$.
Result: The best compositions and their Extra Descriptions.
begin
    if $Ra_4C$ returns a robust composition $\pi_i$ then
        return $\{(\pi_i, \emptyset)\}$;
    else
        $\mathcal{B}_{\pi_0} \leftarrow$ compute Extra Description of $\pi_0$;
        $sol \leftarrow \{(\pi_0, \mathcal{B}_{\pi_0})\}$;
        foreach $\pi_{i_{i \neq 0}}$ do
            while $\pi_i$ computation is in progess by $Ra_4C$ do
                $\mathcal{B}_{\pi_i} \leftarrow$ current Extra Description of $\pi_i$;
                if $\mathcal{B}_{\pi_i} \sqsubset sol.\mathcal{B}$ then stop $\pi_i$ computation;
                if $\mathcal{B}_{\pi_i} \sqsupseteq sol.\mathcal{B}$ then continue to build $\pi_i$;
            if $\pi_i$ is valid then $sol \leftarrow sol \sqcup \{(\pi_i, \mathcal{B}_{\pi_i})\}$;
        return $sol$;
end
```

Algorithm 1 differs from Ra_4C, primarily because it does explore non robust causal links in the process of composition computation and stop its computation in case its *Extra Description* is more specific than one of the pre-computed solutions. Finally we save service compositions with the most general *Extra Descriptions*, i.e., with the least constraints.

The theoretical complexity of this method is the same as approaches without relaxation (e.g., Ra_4C) since the difference between concepts is calculated in a pre-processing phase. However more causal links have to be studied in general cases since we consider also links valued by a Subsume or an Intersection match level. Once the *Extra Description* $\mathcal{B}_\pi$ of the most robust composition $\pi_{i,1 \leq i \leq n}$ is computed, $\mathcal{B}_\pi$ is provided and instantiated e.g., by the end user. Then $\mathcal{B}_\pi$ is used to replace a non robust link by its robust form by directly applying $\mathcal{B}_\pi$ to each non robust link. In the worst case the non robust link will become PlugIn which is robust whereas it will become Exact (also robust) in the other case. Many other subtle ways may be also considered to use $\mathcal{B}_\pi$.

Related Work

Different matchmaking approaches have been studied to check if the request description is equivalent, subsumed, consistent (Paolucci *et al.* 2002), or compatible (Li & Horrocks 2003) with the descriptions of service advertisements.

Most of composition models take advantage of these matchmaking functions by valuating the link between Web service parameters to perform automation semantic Web service composition. However most of them consider non robust causal links hence some inconsistencies in their composition model. (Zhang, Arpinar, & Aleman-Meza 2003) consider a composition of services as a directed graph where nodes are linked by a matching compatibility (Exact, Subsume, PlugIn, Disjoint) between input and output parameters. According to this graph they retrieve the shortest sequence of Web services from the initial requirements to the goal. (Lassila & Dixit 2004) compose workflows using breath-first search, starting with the description of the service they want to replace. Their model is also based on the degree of match introduced by (Paolucci *et al.* 2002). (Cardoso & Sheth 2003) expose a method to compare Web service parameters in case the latter parameters are not parts of the same ontology. They augment standard semantic matching functions with a computation of a syntactic similarity.

To the best of our knowledge, there is no composition model that considers non standard matchmaking functions to value causal links. However there are significant matchmaking functions that may overcome the problem of robustness in Web service composition. For instance (Colucci *et al.* 2005) compare the request description and the advertisement service to infer some matchmakings. The compared descriptions could be incomplete or not fully compatible, so when an element in the request that is not consistent with an element in the offer is found, it is removed (contraction) and each required element that is not present in the offer is added (abduction). It is obvious that contraction does not make sense in Web service composition since the causal links would not be valid. In the opposite abduction is close to our approach since the abduction definition is dual with the difference definition. However some appropriate refinements are necessary to apply it to service composition.

Conclusion and Future Work

In this paper we describe an approach to refine Web service composition as a composition of causal links. Our approach has been directed to meet the main challenge facing service composition in the semantic Web context i.e., how effectively do a composition of services with robust links. This work complements and improves existing techniques to determine the semantic quality and robustness between Web services. In contrast to most works on semantic web service composition, our approach does assume that standard semantic matching functions are under-specified to really provide practical matchmaking between Web service parameters. Instead we have introduced a solution that uses causal links and a robustness criteria to solve the problem composition. Moreover we introduced a method to change a non robust causal link in its robust form by discovering the *Extra Description* that is not specified. This description enables to i)locate non robust causal links, ii)give some explanations about a causal link failure i.e., a composition failure, iii)suggest a solution to perform automated composition. Finally we introduced relaxing constraints in case no composition model is retrieved. The main direction for continuing this work is to consider preconditions and effects of service during the composition process. Another direction is to consider quality of services together with quality of causal links.

References

Ankolenkar, A.; Paolucci, M.; Srinivasan, N.; and Sycara, K. 2004. The owl services coalition, owl-s 1.1 beta release. Technical report.

Baader, F., and Nutt, W. 2003. Basic description logics. In *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press.

Berners-Lee, T.; Hendler, J.; and Lassila, O. 2001. The semantic web. *Scientific American* 284(5):34–43.

Brandt, S.; Kusters, R.; and Turhan, A. 2002. Approximation and difference in description logics. In *KR*, 203–214.

Cardoso, J., and Sheth, A. 2003. Semantic e-workflow composition. *Journal of Intelligent Information Systems* 21:191–225.

Colucci, S.; Noia, T. D.; Sciascio, E. D.; Donini, F.; and Mongiello, M. 2005. Concept abduction and contraction for semantic-based discovery of matches and negotiation spaces in an e-marketplace. In *ECRA*, volume 4, 41–50.

Fensel, D.; Kifer, M.; de Bruijn, J.; and Domingue, J. 2005. Web service modeling ontology (wsmo) submission, w3c member submission.

Küsters, R. 2001. *Non-Standard Inferences in Description Logics*, volume 2100 of *Lecture Notes in Computer Science*. Springer.

Lassila, O., and Dixit, S. 2004. Interleaving discovery and composition for simple workflows. In *Semantic Web Services, AAAI Spring Symposium Series*, 22–26.

Lécué, F., and Léger, A. 2006. A formal model for semantic web service composition. In *ISWC*, 385–398.

Li, L., and Horrocks, I. 2003. A software framework for matchmaking based on semantic web technology. In *World Wide Web*, 331–339. ACM Press.

McAllester, D., and Rosenblitt, D. 1991. Systematic nonlinear planning. 634–639. Menlo Park, CA: AAAI.

McIlraith, S. A., and Son, T. C. 2002. Adapting golog for composition of semantic web services. In *KR*, 482–496.

Paolucci, M.; Kawamura, T.; Payne, T.; and Sycara, K. 2002. Semantic matching of web services capabilities. In *ISWC*, 333–347. Springer-Verlag.

Pistore, M.; Barbon, F.; Bertoli, P.; Shaparau, D.; and Traverso, P. 2004. Planning and monitoring web service composition. In *AIMSA*, 106–115.

Rao, J.; Kungas, P.; and Matskin, M. 2003. Application of linear logic to web service composition. In *ISWC*, 3–10.

Russell, S., and Norvig, P. 1995. *Artificial Intelligence: a modern approach*. Prentice-Hall.

Sirin, E.; Parsia, B.; Wu, D.; Hendler, J.; and Nau, D. 2004. Htn planning for web service composition using shop2. *Web Semantics Journal* 377–396.

Sivashanmugam, K.; Verma, K.; Sheth, A.; and Miller, J. 2003. Adding semantics to web services standards. In *ICWS*, 395–401.

Teege, G. 1994. Making the difference: A subtraction operation for description logics. In *KR*, 540–550.

Zhang, R.; Arpinar, I. B.; and Aleman-Meza, B. 2003. Automatic composition of semantic web services. In *ICWS*, 38–41.