A P2P-based Architecture for Semantic Web Service Automatic Composition[*]

Federica Mandreoli, Antonio M. Perdichizzi
DII
Università di Modena e Reggio Emilia
via Vignolese, 905/b, 41100 Modena, Italy
{fmandreoli, aperdichizzi}@unimo.it

Wilma Penzo
DEIS
Università di Bologna
viale Risorgimento, 2, 40136 Bologna, Italy
wpenzo@deis.unibo.it

Abstract

The ultimate vision for eBusiness is an Internet-based global market place, accessible to all enterprises, regardless of size and geographical location, where automatic co-operation and integration among firms are allowed and enhanced. A powerful mean for these purposes is represented by sharing, reusing and composing value-added services made available on the Web, i.e. Web services. In this scenario, by making the Web content machine accessible and understandable, semantic Web services aim to provide efficient and effective Web service automatic discovery, selection, and composition. As part of the NeP4B project, in this paper we propose an architecture to address these issues in a flexible and scalable P2P network.

1. Introduction

Currently the Internet is mainly a collection of information but does not yet support processing this information. Recent research efforts on semantic Web services (SWSs) aim to make this information machine accessible and understandable, trying to lift the Internet to a new level of service. The ultimate vision is to enable full interoperability between users through a semantic Web of services whose properties, capabilities, interfaces and effects are encoded in an unambiguous form [10]. The Internet could then become a global common platform where to carry out various commercial activities. It would be able to lessen market failures lowering the barriers to providing new offers and entering new markets and to allow automatic cooperation and supply chain integrated management to create and sustain competitive advantage. Semantic Web services head in this direction, aiming to realize automatic discovery, selection and composition of existing services. In a business environment, that results in the possibility for enterprises to release from the burden of complex, slow and expensive software integration and to focus instead on the value of their offerings and mission critical tasks.

As part of the NeP4B project, our goal is to design a scalable and flexible framework to provide advanced enterprise interoperation in a common business environment. This is based on a peer-to-peer (P2P) data-driven SWS network for B2B applications. Firms are free to join and leave the network at any time, to act both as a providers of their own services and consumers, and to classify their own profiles, offers, services and other features to gain public visibility to potential customers and partners. To support this objective, in this paper we present the FLO^{2}WER framework. This provides an architectural solution to overcome the heterogeneity of dictionaries and the lack of shared service knowledge due to the autonomy and heterogeneity of peers in such a dynamic context, creating the proper environment for service automatic composition to be performed.

Starting from the context definition (Section 2), we will discuss the main architectural issues (Section 3) and propose a possible architectural solution to face the identified challenges in the NeP4B project (Section 4). We will then go through a sample execution flow (Section 5) and provide conclusions and future work (Section 6).

2 Semantic Web and P2P for Co-opetition

Information and communication technologies (ICTs) over the Web have become a strategic asset in the global economic context. The Semantic Web fosters the vision of an Internet-based global market place, accessible to all small and medium enterprises (SMEs), regardless of size and geographical location, to allow and enhance automatic cooperation and competition (co-opetition) to face globalization main challenges. In this setting, peer-to-peer architectures well support firms engaged in business network relations to interact with each other both as providers and consumers of services. To allow them to communicate, there

[*]This work has been partially supported by the Italian Council co-founded FIRB Project: “NeP4B: Networked Peers for Business”.

is the need to cope with the inherent heterogeneity of this open environment. Therefore peer-to-peer systems empowered by Semantic Web technologies represent a valid mean to provide solutions for both the interoperability and communicability needs.

To develop this vision, NeP4B bases on the development of a suite of intelligent, trusted and distributed data-driven services, with high added value. It relies on the concept of semantic peers constituting a virtual network, where each peer represents a SME exposing Web services on the basis of its own internal and external business needs. Peers are fully autonomous of participating to the network and of joining and leaving it at any time.

Each SWS is marked up by adopting an ontology language such as DAML/OWL-S [5] and relies on its peer's underlying data, described by a schema referring to a local dictionary. Data may be a collection of structure, semistructured, unstructured and multimedia data, which can be shared among peers in order to allow services, particularly information providing services, to collect information by spanning over the network. To this end, schema mappings are provided locally between pairs of peers.

In this distributed and heterogeneous context, services have to be searchable to meet users' requests and invocable both for a stand alone execution or an automatic composition process [4, 7]. The latter is crucial to allow scalability, flexibility and effectiveness of the service network, because it envisions the challenging task of enabling the automatic re-use of existing services to generate new services.

3 Web Service Architectures: State of Art

Much of the work done on Web service architectures proposes solutions based on centralized registries, such as UDDI [1], where every Web service coming on line advertises its capabilities and functionalities with the registry. Centralized control of published services allows to ease discovery and composition of services but it suffers from the traditional weaknesses of centralized systems, namely single point of failure, and performance bottlenecks. Furthermore, these solutions do not ensure the required scalability to support a flexible and dynamic environment such as the NeP4B context.

An alternative to this approach is provided by P2P computing, where Web services interact with each other dynamically, without any centralized control. In such a context, there have been several proposals for Semantic Web service discovery (e.g. [13, 16]). However such a decentralized scenario does not well support automatic composition of services because it does not provide a known and definite service space, as needed for this process. There have been several proposals to overcome these problems by either trolling both the construction of the overlay network and the location of data within the system, i.e. structured systems as Chord (e.g. [6]), or only defining its topology ex-ante (e.g. [15]).

Finally, it should be noted that both approaches rely on a common service dictionary for composition to take place.

Considering all of the above, we propose a hybrid approach, the *FLexible GOal-Oriented WEb SeRvice (FLO^{2}WER)* framework, which exploits the advantages of centralized registries for service discovery and composition, as well as the dynamism of non-structured P2P networks that ensure the peers' full autonomy.

4 The FLO^{2}WER Framework

The main idea of the FLO^{2}WER framework is that, while not centralizing the knowledge of what specific services are available in the system, we keep centralized knowledge of what objectives may be satisfied within the network, namely *Goals*. A Goal is the conceptualization of a domain of services whose ultimate aims are identical or similar. For example, all services that, in the same geographical area provide driving directions from a departure point to a destination. Some could allow the requester to choose among different route options, for example the shortest or the most economical one, while others would only provide the fastest. Even if they may have different reference dictionaries, specific requirements or functionalities, they do answer the same requestor's need: To provide driving directions within a certain geographical area. Therefore they would be well represented by the same Goal. Each Goal specifies therefore a subnetwork of specific services, and it is stored in an appropriate repository, called *Goal Repository*.

Using a repository of Goals has several advantages that do not come with loss of flexibility or scalability. Firstly, Goals constitute the domain for the composition purposes. Now its dimension is greatly reduced w.r.t. the underlying service level as Goals only represent the objectives the network is able to satisfy rather than how they are satisfied. In this way, we move the issues of discovery and defining a composition pattern from service level to Goal level, namely *Goal discovery* and *Goal composition* respectively. Once Goals have been identified, it is possible to limit the search of the most suitable candidates within their own service networks, e.g. for the composition synthesis process [2, 11].

Secondly, the Goal layer acts as a "semantic service integrator" reconciling peers' service models and dictionaries. Goals are indeed described in a common language and consistency of concepts within the Goal space is ensured by referring to a domain ontology. In this way the inherent heterogeneity of an open P2P system is reconciled within a homogenous space allowing the communication of different

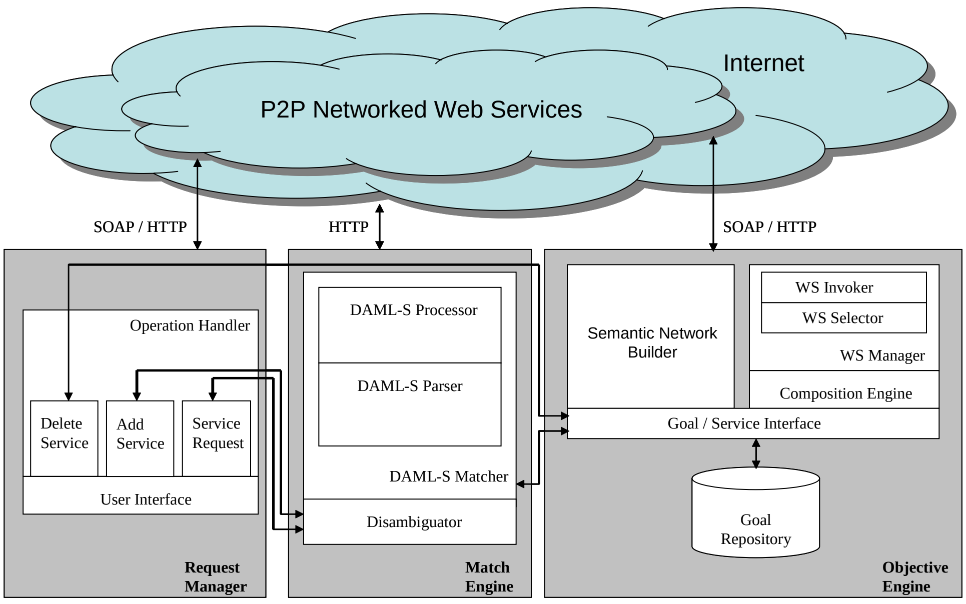

Figure 1. The FLO^{2}WER Architecture

services through the Goal layer. In the literature, there are several works dealing with the issue of building semantic service integration systems for composition purposes (e.g. [2, 17, 14]). However, they mainly focus on the translation of service descriptions into an internal form to apply specific techniques for automatic composition. In this paper, rather than focusing on this aspect, we create a proper interoperable environment for such kind of proposals be applicable in our context, regardless of their specificity.

A schema of the FLO^{2}WER architecture we propose is shown in Figure 1. It is a high level description which is meant to show the basic processes and data flows needed to support automatic semantic Web service handling. It is composed of three main modules, the *Request Manager*, the *Match Engine* and the *Objective Engine*, and it is founded on the community ontology [3] described in the Goal Repository.

The user interacts with the system through the Request Manager. Activating the other modules, which are described in the following subsections, it allows to perform three basic operations: Look for, add, or delete a Web service.

4.1 The Goal Repository

The Goal Repository comprises several aspects: a DAML-S description of Goals and a DAML domain ontology which represents the semantics of the information and data of the application domain. It may also include a service ontology specifying the meaning of the offered operations. To this extent, we adopt two semantic integration approaches [3]: A service-oriented approach for Goal descriptions, as they are built taking into account the service available in the underlying P2P network; a client-tailored approach for the domain ontology which is independent from the services available and that, if needed, can be downloaded off the Web.

Here, DAML-S is used to describe capabilities of a Goal as the Service Profile of a Web service. It describes a Goal as an atomic process thus specifying what the Goal is for, the inputs it requires, the outputs it produces and the pre-conditions that must hold for the Goal to take effect. DAML-S Service Profile also describes the post-conditions, i.e. the service execution effects on the real world. However, for Goal discovery and composition, we are concerned with the knowledge effects (outputs) rather than the physical effects (post-conditions) of executing Web services, which might in turn be relevant for invocation. Inputs, outputs, and pre-conditions refer to the domain ontology and are mapped towards the underlying Goal's subnetwork of services. Thus, we do not maintain mappings from the domain ontology to the peers' schemas, but they can be derived from the mappings between Goals and services.

4.2 The Match Engine

It is invoked by the Request Manager when the user adds or looks for a service and it is composed by the *Disambiguator* and the *DAML-S Matcher*. The Disambiguator takes non-DAML-S user's request and outputs a corresponding DAML-S description disambiguating the information managed against the domain ontology. A disambiguation process is described in [9]: Future research on this approach aims to also disambiguate natural language requests within an accuracy range. The DAML-S Matcher takes a DAML-S user's request to match it with the Goal's descriptions. [13] proposes a possible implementation of the matcher in [12] in an unstructured P2P context. It is composed by the *DAML-S Parser* and the *DAML-S Processor*. The former translates DAML ontologies and DAML-S specifications in a set of predicates, whereas the latter implements a DAML inference engine.

4.3 The Objective Engine

The central component is the *Goal/Service Interface*. Besides mediating between the other components, it allows the automatic creation, activation, deactivation and deletion of a Goal. A Goal is deactivated when its subnetwork is empty, and it is activated again if it matches a new service to be added to its network. There may be several possible policies to adopt for deletion, e.g. timeouts: A deactivated Goal could be deleted after a certain time interval during which it did not match any new added service.

The *Semantic Network Builder* is activated to accomplish a delete or a add service request and drives the integration process necessary to add or delete a service. It maps each Goal to only one service, called the *entry point* of the Goal subnetwork[1]. While a structured network is supposed not to be appropriate in the NeP4B context, we can still control how each subnetwork should evolve, optimizing a trade-off between the cost of establishing mappings and the cost of navigating the network in the search of proper services to be chosen. There are several topologies to compare, such as ring, bus, or Cayley graphs networks like hypercubes or star graph [15]. When a new service joins the network, it is matched against existing Goals. If a matching Goal is found, the Semantic Network Builder maps the new service semantically to a service of the subnetwork, namely the *service integrator*. This is in charge of providing the new service with the semantic mappings needed depending on the policies adopted for the network construction. When a service leaves the network, the same process takes place to create the new mappings as required to maintain the network topology. Such new semantic mappings are derived by the Goal matching information, along the path of mappings from the entry point to the service integrator, assuming the mapping function to be transitive. At last, when a matching Goal is not found for a new service, or when the last service of a Goal subnetwork leaves the system, the Semantic Network Builder passes this information to the Service/Goal Interface, that provides to create or deactivate the Goal, respectively.

[1]There may be also more entry points to one subnetwork to avoid single point of failure. What matters is that the Goal does not have to know the whole set of services which constitute its subnetwork.

The Composition Engine implements the Goal discovery and composition, identifying Goals suitable to answer a user's request. Once one Goal or a pattern of Goals have been found, the Composition Engine invokes the WS Manager which handles the automatic service composition process. It is constituted by the *Web Service Selector* and the *Web Service Invoker*. The Web Service Selector enters, through the entry point, each of the Goal subnetwork identified, and selects the most suitable service based on different ranking criteria such as reliability, cost, quality of service, trust and reputation and, if available, on its process description[2]. Once Web services have been selected, the Web Service Invoker manages their execution and communication. Successful Goal composition patterns are then stored in the Goal Repository for future use.

5 A Sample Execution Flow

There are three main execution flows, associated with the request, the insertion, and the deletion of a Web service, respectively. For space reasons, we show only the case of a Web service request, going through an example. Let us consider a specific application scenario, such as the multimedia industry. In such a context, consider a traditional information provider enterprise which would like to exploit the potentials of providing information through the new mobile media platform (SMS, MMS, WAP). It has the content, but does not have the proper competencies to manipulate it, refining it and making it suitable for the new platform, nor to distribute it via the new infrastructure. Therefore, to reach its objective, it needs to cooperate with other enterprises whose expertise is in these fields of business. The FLO^{2}WER Architecture could ease its discovery task and also provide a composed service able to entirely satisfy its conversion and distribution needs. The information provider starts therefore expressing its needs by using the Service Request module. Suppose it states it is looking for *a service to distribute pictures and text to mobile phones*. The request is passed to the Disambiguator in the Match Engine, which produces a DAML-S description to pass on to the DAML-S Matcher, which, through the Goal/Service Interface, looks for the Goal which best satisfies the request, among those in the Goal Repository. Suppose that a matching Goal is found, whose description is *multimedia distribution through mobile media platform*. The Composition Engine is thus invoked and finds, for example through an algorithm based on outputs-inputs matching, as the one in [8], a possible composable Goal, whose description is *making multimedia suitable for mobile media distribution*. This composition is therefore proposed to the service provider,

[2]DAML-S Web service description includes a Process Model, where it is defined what is needed for a proper interaction of services

which may then accept it, if its format is not already suitable. Once the user has accepted, the Web Service Invoker invokes the best service of each Goal (either automatically chosen under some user's defined criteria, such as cost, reliability, or service execution time, or selected by the user itself) and maps its inputs and outputs to the Goal ones to allow the two different services to communicate and exchange data consistently through the Goal layer. This is performed exploiting the semantic mappings established in the network of services underlying the selected Goals. When the last service is executed, results are returned to the information provider, which has therefore been able to automatically discover and cooperate with other enterprises to satisfy its needs.

6 Conclusions

In this work, we addressed the architectural issues involved in the specification of a flexible framework to support large scale interoperation of semantic Web services in a dynamic and heterogeneous P2P context. At this purpose, we have adopted a hybrid approach which exploits the advantages of centralized registries for service discovery and composition, as well as the dynamism and the scalability of non-structured P2P networks.

The FLO^{2}WER architecture we propose aims to overcome the heterogeneity of dictionaries and the lack of shared service knowledge due to the autonomy of peers in the NeP4B scenario, and allows to create the proper conditions for the application of specific techniques for automatic composition of semantic Web services. This is performed by introducing Goals as centralized knowledge of what objectives may be satisfied within the network, which allows to greatly reduce the domain for the composition purposes, abstracting from service specificities, and focusing on the capabilities of domains of services.

We believe that this work might constitute a solid starting point for the NeP4B project as it defines the basic building blocks and execution flows to enable automatic service discovery and composition. In our future works we will focus on the development of the FLO^{2}WER components.

References

[1] UDDI: *The UDDI Technical White Paper*. http://www.uddi.org/, 2000.

[2] D. Berardi, D. Calvanese, G. De Giacomo, R. Hull, and M. Mecella. Automatic Composition of Transition-based Semantic Web Services with Messaging. In *Proc. of VLDB*, 2005.

[3] D. Berardi, D. Calvanese, G. De Giacomo, and M. Mecella. Automatic Web Service Composition: Service-tailored vs. Client-tailored Approaches. In *Proc. of AISC Workshop (in conj. with ECAI)*, 2006.

[4] D. Berardi, D. Calvanese, D. G. Giuseppe, M. Lenzerini, and M. Mecella. Automatic Composition of E-Services that Export their Behaviour. In *Proc. of ICSOC*, 2003.

[5] M. H. Burstein, J. R. Hobbs, O. Lassila, D. L. Martin, D. V. McDermott, S. A. McIlraith, S. Narayanan, M. Paolucci, T. R. Payne, and K. P. Sycara. DAML-S: Web Service Description for the Semantic Web. In *Proc. of ISWC*, 2002.

[6] F. Emekçi, O. D. Sahin, D. Agrawal, and Amr El Abbadi. A Peer-to-Peer Framework for Web Service Discovery with Ranking. In *Proc. of ICWS*, 2004.

[7] R. Hamadi and B. Benatallah. A Petri Net-based Model for Web Service Composition. In *Proc. of ADC*, 2003.

[8] J. Liu, N. Gu, Y. Zong, Z. Ding, S. Zhang, and Q. Zhang. Web Services Automatic Composition Based on QoS. In *Proc. of ICEBE*, 2005.

[9] F. Mandreoli, R. Martoglia, and E. Ronchetti. Versatile Structural Disambiguation for Semantic-aware Applications. In *Proc. of CIKM*, 2005.

[10] S. A. McIlraith, T. Cao Son, and H. Zeng. Semantic Web Services. *IEEE Intelligent Systems*, 16(2):46–53, 2001.

[11] S. Narayanan and S. McIlraith. Simulation, Verification and Automated Composition of Web Services. In *Proc. of WWW*, 2002.

[12] M. Paolucci, T. Kawamura, T. R. Payne, and K. P. Sycara. Semantic Matching of Web Services Capabilities. In *Proc. of ISWC*, 2002.

[13] M. Paolucci, K.P. Sycara, T. Nishimura, and N. Srinivasan. Using DAML-S for P2P Discovery. In *Proc. of ICWS*, 2003.

[14] M. Pistore, P. Traverso, P. Bertoli, and A. Marconi. Automated Synthesis of Composite BPEL4WS Web Services. In *Proc. of ICWS*, 2005.

[15] M. Schlosser, M. Sintek, S. Decker, and W. Nejdl. A Scalable and Ontology-Based P2P Infrastructure for Semantic Web Services. In *Proc. of P2P*, 2002.

[16] C. Schmidt and M. Parashar. A Peer-to-Peer Approach to Web Service Discovery. *WWW*, 7(2), 2004.

[17] D. Wu, B. Parsia, E. Sirin, J. A. Hendler, and D. S. Nau. Automating DAML-S Web Services Composition Using SHOP2. In *Proc. of ISWC*, 2003.