A P2P Discovery mechanism for Web Service Execution Environment

Ioan Toma, Brahmananda Sapkota, James Scicluna, Juan Miguel Gomez, Dumitru Roman, and Dieter Fensel

Digital Enterprise Research Institute (DERI), Galway, Ireland and Innsbruck, Austria.
<firstname>.<lastname>@deri.org

Abstract. A fundamental step towards the realization of Semantic Web Services vision is the automatic location of services given a specific user request. One important aspect that has to be considered is the distributed location of services. In this paper we present a scalable approach for automatic discovery of services over distributed execution environments. Our solution is based on P2P technologies that proved to be scalable, efficient and robust solutions for distributed systems. The Web Service Execution Environment (WSMX) is our test bed for such an architecture.

1 Introduction

With the advent of Web services, the Web is changing from a mere repository of information to a new vehicle for business transactions and information exchange. Large organizations are increasingly relying on Web services technologies for large-scale software development and sharing or exposing of services within an organization. Web services are loosely coupled software components published, located and invoked across the web. The growing number of Web services available on the Web raises a new and challenging problem, *the location and discovery of such services*. The lack of a proper discovery mechanism is hindering the potential of these technologies. Human user intervention is required to locate a suitable service for a given request making these technologies limited with regard to scalability and efficiency. Semantic Web technologies promise to make information understandable by computers through it's key-enabling technology: ontologies. Ontologies are a formal, explicit and shared specification of a conceptualization [8]. The breakthrough of adding semantics to Web services leads to the Semantic Web services paradigm. A successful realization of the Semantic Web services paradigm requires an automatic and scalable discovery mechanism. Furthermore, the distributed nature of Semantic Web services has to be considered when designing such a mechanism.

This paper addresses the discovery problem for a specific Semantic Web service execution environment, namely Web Service Modelling Execution Environment(WSMX) [1]. More precisely, it proposes a P2P Discovery mechanism for

[1] http://www.wsmo.org/wsmx/

Semantic Web services descriptions that are registered with Web Service Modelling Execution Environment. The rest of the paper is structured as follows: Section 2 provides a short overview of the execution environment for Semantic Web services - WSMX, the "test bed" for our P2P discovery mechanism. Section 3 presents some insights of Web service discovery domain. The possible approaches for distributed discovery in WSMX are detailed in Section 4. Our P2P based solution for WSMX distributed discovery is described in Section 5 and the related work is presented in Section 6. Finally, Section 7 concludes the paper and presents our future work.

2 WSMX Overview

The Web Services Execution Environment(WSMX) is the reference implementation for WSMO [16]. WSMX aims to provide a test bed for WSMO and as well to demonstrate the viability of using Semantic Web Services as a means to achieve dynamic interoperation between business partners. WSMX uses Semantic Web technologies to discover, mediate, select and invoke Web services based on their formal descriptions. In short, WSMX functionality could be summarized as performing discovery, mediation, selection and invocation of Web services on receiving a user goal specified in WSML [4], the underlying formal language of WSMO. The user goal is first matched against the formal descriptions of Web services registered with WSMX. In case of success, one or more service descriptions (ranked according to user preference) can be returned. The most appropriate service selected by the user is further invoked and the result is given back to user. Prior the invocation step, WSMX ensures that the data provided for the service invocation is in the format that Web service expects. If necessary a data mediation process is performed to assure the inter-operability between different entities. Presently, the WSMX architecture relies on a set of loosely-coupled main components that provide functionality for each step of Web service usage process: discovery, selection, mediation and invocation.

3 Some insights of Web Service Discovery

The problem of automatic location of services, also known as service discovery, is a popular research topic. A workable solution to this problem must rely on a complete and correct discovery model derived by an in-depth analysis of major issues in service discovery. Completeness implies that all relevant entities are discovered; correctness implies that only the relevant entities are discovered. Such a conceptual model is provided by WSMO Discovery [10], and is based on two fundamental principles: *(1) a strict distinction between the notion of service and Web services* and *(2) a clear separation between different steps of the discovery process.* According to [10], a service can be seen as a concrete instance of a Web service which has all the inputs specified, while a Web service can be seen as an abstract entity, as a class of concrete services. Our proposed

approach for distributed discovery in WSMX is based on this model which we will briefly describe below.

The different steps of discovery process are: *Goal Discovery*, *Goal Refinement*, *Web service Discovery* and *Service Discovery*. The first step, *Goal Discovery* is about discovering abstract, pre-defined, reusable goals given the input provided by the user. This input can be in natural language or some specific formalism. The second step, *Goal Refinement* is about refining the pre-defined goal discovered in the previous step, based on the given user desire. As a result a refined or parameterized goal is "found". Please notice that the refined or parameterized goal is not used in Web service discovery step but only in the service discovery step also known as service contracting. The third step *Web service Discovery* is about finding abstract Web service descriptions that matched the pre-defined goal already discovered. Different approaches to realize Web service discovery are also described in [10]. The last step, *Service Discovery* is about finding real services, concrete services whom abstract descriptions were discovered in the previous step.

The solution that we propose for WSMX distributed discovery considers only the *Web service discovery* step from the previously described framework. Semantic descriptions of request and services specified in WSMO are the entities manipulated in our solution. Our approach provides a means to bring together distributed descriptions in order to match them against the user's request. A more detailed description of our solution is provided in Section 5.

4 Distributed Discovery in WSMX - possible approaches

Distributed discovery is one of the main aspects that WSMX deals with to realize a goal. Let us look at the the following scenario where distributed discovery is essential. Given a user request, WSMX has to find services that fulfill the user requirements. In this process, WSMX discovery engine evaluates user requirements against the capabilities of the services registered locally. For convenience we call the services registered locally, *local services*. If the capabilities of the local services do not fulfill the user requirements, WSMX discovery engine should, in cooperation with other systems in the network, find services that fulfill user requirements. The cooperation between systems can be achieved by forwarding the requests to other systems that potentially can find services matching user requests. Note that the systems used during a discovery process may not necessarily be WSMXes. However, in the rest of the paper, we restrict ourselves to WSMXes only. This is purely to simplify the complexity of the problem domain. Interoperability with other different systems may be achieved by using WSMX adapters.

The services descriptions registered with WSMXes scattered all over the world, can be discovered using either of the three fundamental approaches: UDDI like centralized approach [2], P2P like approach, and Triple Space paradigm [5]. Following subsections briefly discuss about all three approaches mentioned above.

4.1 Centralized Approach

The centralized approach is the simplest way to implement a distributed discovery mechanism. In this approach a centralized known server is used as a registry for WSMX platforms. Figure 1 depicts this approach.

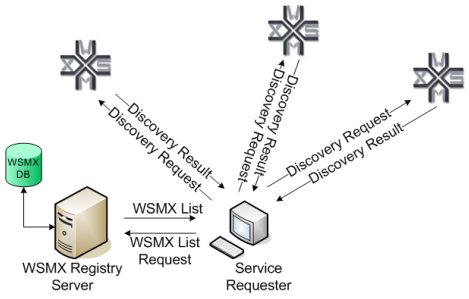

Fig. 1. Centralized Registry scenario for Distributed Discovery in WSMX

The WSMX registry server provides interfaces for requesters and WSMX platforms. A WSMX system registers over this server and a requester retrieves the list of available WSMX platforms such that requests for discovery can be sent for each. This approach is unscalable and highly inefficient since it also puts burden on the requester to send multiple requests to different WSMX platforms. This approach is thus not considered in this paper.

4.2 P2P approaches

In P2P like approach there will be no dedicated central registry. All peers in the network are functionally equal and co-operate with each other for answering a user request. Therefore, P2P approaches minimize the limitations of the centralized approaches described above. Since a P2P like approach is more plausible than a centralized approach, we envision a P2P network of WSMXes where peers make use of P2P protocols(eg. [9], [19]) to find each other. The way the peers cooperate in the network defines the structure of the P2P network. Different variants of P2P like approach are described below.

A pure P2P approach In pure P2P approach all peers behave both as a server and as a client to other peers. This approach is illustrated in Figure 2 where, each WSMX peer is essentially both a client and a server. The Service Requester will have to know at least one WSMX peer to which a request will be sent. If the goal described in such a request cannot be satisfied, then the request is forwarded to another known WSMX peer. The selection criteria of the next WSMX may be based on the following:

– A taxonomy of WSMX repositories might be used to define what categories of Web services a WSMX hosts, hence the one which is likely to provide the required service is chosen.
– The local WSMX registry may contain the history of the peer WSMX that were forwarded the similar requests in the past. Therefore, the one with high probability of answering the request will be chosen.

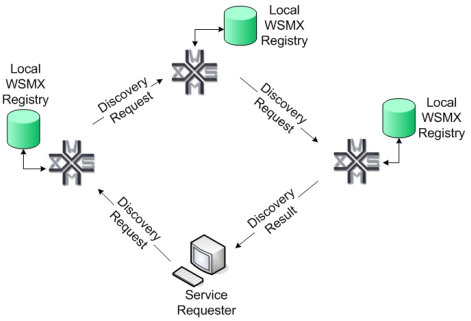

Fig. 2. A Pure Peer-to-Peer approach for Distributed Discovery in WSMX

Once the goal described is matched, a result is sent back to the client. The service can then be executed on the respective WSMX peer and results are sent back to the Service Requester. In order for this to happen, the request should also hold information about the Service Requester itself. Ideally, a list of the respective types of matches obtained from each WSMX is also maintained.

A hybrid P2P approach In this approach, features of both the centralized and the pure P2P approached are combined. The idea behind this approach is to form two peer groups: 'Child Nodes' and 'Super-Nodes' peers. Child Nodes are peers which know exactly one Super Node. Super Nodes on the other hand, besides their functionality which is the same with the one of Child Nodes, act as registries for the former ones. Super Nodes maintain two lists: one keeping the information about Child Nodes that are 'registered' with them and the other one keeping the information of its neighboring Super Nodes. Figure 3 shows this approach.

In this hybrid system, the request may be forwarded to either a Super Node or a Child Node. Lets assume a Child Node gets a request which will be executed locally. If Child Node gets no positive response then the request is forward to its Super Node where the request is executed locally. If the result is negative, the Super Node will then consult other directly connected Child Nodes for the services matching the goal. If the query fails, then the request is forwarded to the next Super Node and the process is repeated until all the nodes are consulted or the goal is matched. It is possible that no node can provide positive result. In this case, the result that is forwarded to the requester will be negative.

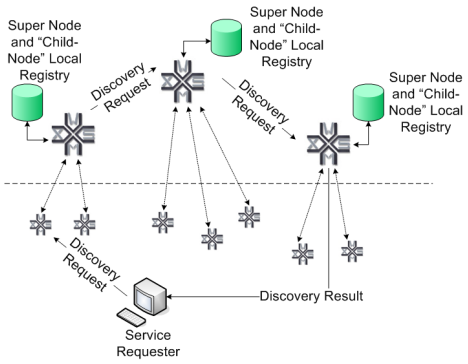

Fig. 3. A Hybrid Peer-to-Peer approach for Distributed Discovery in WSMX

4.3 Triple Space approach

In this approach, WSMXes make use of Tuple/Triple Space paradigm. Tuple Space [6] provides a simple but powerful communication and coordination mechanism. It provides a shared space where applications can write and read information in order to communicate with each other. Triple Space [5] inherits the Tuple Space communication model and projects it in the context of the Semantic Web [3]. Instead of sending messages back and forth, applications can communicate simply by writing and reading RDF [11] triples in a shared space, called Triple Space.

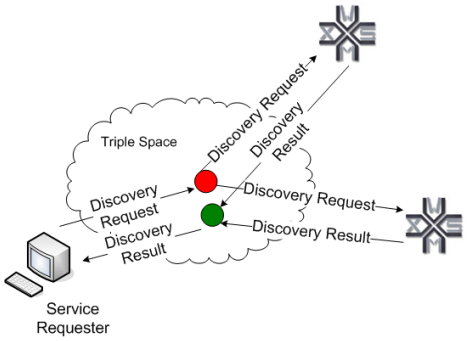

Fig. 4. A Triple Space approach for Distributed Discovery in WSMX

Tuple/Triple Space approach for distributed discovery in WSMX is illustrated in Figure 4. In this approach a requester application writes a request in the space. It might be the case that services registered with different WSMXes could fulfill the request. Therefore, WSMXes can read the request from the space and start the local discovery process. If the discovery process is successful each

WSMX writes a result back in the space. In case the local discovery process fails, WSMX, does not write anything back to the space. In this way, WSMXes independently read and evaluate the request locally. This process seems viable, however, WSMXes need to know the shared space in use.

5 Peer-To-Peer discovery solution for WSMX

From the previous possible approaches for distributed discovery in WSMX we have selected the *pure P2P approach*. The rationale behind choosing a P2P solution for WSMX distributed discovery is: *(1)* P2P approaches are *scalable* solutions compared with centralized approaches. *(2)* P2P approaches are proven workable solutions (other technologies like Triple Space although with a great potential have not been implemented and tested so far). Among possible P2P solutions, we have selected a *pure P2P approach* because we envision a network of **equal** WSMXes that host semantic annotation for services.

In our approach, equal WSMX peers which participate in the service discovery process have to match the local registered services against a broadcasted query. A major aspect that have to be considered in this context is the topology of network. For message routing the topology of the network has significant impact on the overall performance of the service discovery process. The approach that we adopt to address these aspects is the HyperCuP (Hypercube P2P) [17] approach. HyperCuP decreases the big overhead of network communication by providing a topology based on a structure called *hypercube*: a generalization of a 3-cube to n dimensions. In the resultant graph the connection between neighbored nodes can be associated with a specific dimension of the hypercube. This allows us to define a message broadcast scheme with certain guarantees: nodes receive a message exactly once and the number of messages sent is linearly dependent on the number of nodes in the network. A set of structuring ontology concepts is used to build a hypercube consisting of distinct concept clusters. A query which consists of a logical combination of service and domain ontology concepts is routed to all relevant concept clusters. Within a concept cluster the message is broadcasted to all contained peers. If the query formulation matches the conceptual description of a service the representing peer is reacting by sending an according response to the requester.

If for the distributed discovery we adopt the infrastructure offered by the HyperCuP approach, for the local discovery, we use a keyword-based approach. In our understanding local discovery is roughly the match of request description against services descriptions executed 'inside' one WSMX. The descriptions are locally available: services descriptions are already registered with the local WSMX and the request is made available by broadcasting into the network. The keyword-based discovery consists on matching keywords from the `nonfunctional properties` section of the request description against the keywords from the `nonfunctional properties` section of services descriptions. For more insights on how keyword-based local discovery works we refer the reader to [12].

From the architectural point of view the WSMX discovery component is divided in two sub-components. One which implements the local discovery functionality (local discovery sub-component) and another which manages the P2P communication with other WSMXes (communication discovery sub-component).

For a better understanding of our solution lets consider the events sequence for a simple distributed discovery scenario. A necessary condition that has to hold before considering any distributed discovery scenarios is that the WSMXes P2P network is already formed. First a Service Requester sends her/his request formalized in WSML to a known WSMX peer. Following the execution semantic of discovery in WSMX [21] the goal flows inside WSMX and is provided as input to the discovery component. The request is first internally processed by the communication discovery sub-component which in this case simply forwards it to the local discovery sub-component. The request is matched against local registered service descriptions. In case no services are found that can fulfil the the request, the communication discovery sub-component creates a request message that we will be sent to the neighbor WSMX peers. The requested message contains basically the same information as the initial message plus additional communication information: the *identifier* of the current WSMX machine and a *time value*. The time value includes a *time stamp* plus a *timeout* value specifying how long the message is considered a valid request. Once a communication discovery sub-component from a neighboring peer receives a request message it knows how to extract the extra communication information: *identifier* and *time value*. The rest of the message which contains only information about the request itself will be handed over to the local discovery component of the current WSMX. The process goes on until at least one service description registered with one WSMX from the network is found or all the services description in the P2P WSMX network were checked without success. In both cases, a response message is constructed and sent back to the originated WSMX peer.

6 Related Work

Several P2P approaches for locating services in a distributed environment already exist (cf. [1] [18], [15], [20], [14]).

For instance METEOR-S ([15], [20]) provides a solution based on a P2P network of UDDI semantically annotated registries which are used to locate services based on a given request. Requests are created by populating a predefined user request template. However, P2P network of registries envisioned in METEOR-S suffers from single point of failure as there is only single entry point defined to enter P2P network of registries. Our approach is different from METEOR-S mainly in terms of the P2P network architecture and the model used for service and request descriptions.

Another approach which uses semantic descriptions of services combined with P2P network topology is described in [14]. This approach has some drawbacks which are derived from the technologies used. First, due to Gnutella [7] underling architecture, every peer creates a high amount of communication overhead.

Second, the use of DAML-S for service description inherits the drawbacks of this model [13].

Our solution based on HyperCuP as network topology and WSMO as conceptual model for describing services overcomes many of the drawbacks of previously mentioned approaches. HyperCuP ensures a P2P infrastructure which once constructed, can guarantee a very efficient routing of messages, one important requirement for scalable distributed discovery mechanism. The use of WSMO on the other hand has clear advantages due to the complete and precise model it offers to describe services and user requests. Although for the local discovery we used a keyword-based approach, replacing this component with a logic based approach component, once available, should not be too difficult.

7 Conclusions and Future Work

In this paper we presented a P2P based scalable discovery mechanism for Web Service Execution Environment. We have developed our solution based on: *(1)HyperCup* [17] approach as network topology, *(2)WSMO* [16] as a conceptual model for services and requests description and *(3)WSMO Discovery* [10] as a conceptual model for discovery. The mechanism presented addresses one of the main challenges of Semantic Web services: *how to locate services in a distributed environment*. In the current version, keyword based discovery mechanism is used for the local discovery. In future, we will test our solution using logic based discovery approaches for local discovery. The evaluation of our solution in a larger network of WSMXes is also left for future work. In order to make discovery mechanism more dynamic and accurate we aim to investigate and implement Triple Space discovery mechanism for WSMX and compare it with our current approach.

Acknowledgements. We would like to thank all members of the WSMO and WSMX Working Groups for fruitful discussions. The work has been supported by the European Commission under the projects DIP, Knowledge Web, InfraWebs, SEKT, SWWS, ASG and Esperonto, by Science Foundation Ireland under the DERI-Lion project, by the Vienna city government under the CoOperate programme and by the FIT-IT (Forschung, Innovation, Technologie - Informationstechnologie) under the projects RW^2 and TSC.

References

1. Farnoush Banaei-Kashani, Ching-Chien Chen, and Cyrus Shahabi. WSPDS: Web Services Peer-to-Peer Discovery Service. In *Proceedings of the International Conference on Internet Computing*, pages 733–743, 2004.
2. T. Bellwood, L. Clment, D. Ehnebuske, A. Hately, Maryann Hondo, Y.L. Husband, K. Januszewski, S. Lee, B. McKee, J. Munter, and C. von Riegen. UDDI Version 3.0. `http://uddi.org/pubs/uddi-v3.00-published-20020719.htm`, July 2002.
3. T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. The Scientific American, May 2004.

4. Jos de Bruijn, Holger Lausen, and Dieter Fensel. The WSML Family of Representation Languages. Working draft, Digital Enterprise Research Insitute (DERI), November 2004.
5. Dieter Fensel. Triple-based Computing. Technical report, Digital Enterprise Research Insitute (DERI), May 2004. Available from `http://www.deri.org/TR/2004-05-31`.
6. D. Gerlernter. *Mirrorworlds*. Oxford University Press, 1992.
7. Gnutella. Gnutella RFC. http://rfc-gnutella.sourceforge.net, 2003.
8. T. Gruber. A translation approach to portable ontology specifications. In *Knowledge Acquisition*, pages 199–220, 1993.
9. JXTA. The JXTA project. `http://www.jxta.org/`.
10. Uwe Keller, Rubén Lara, and Axel Polleres (*editors*). WSMO Discovery. Working draft, Digital Enterprise Research Insitute (DERI), October 2004. Available from `http://www.wsmo.org/2004/d5.1/v0.1`.
11. G. Klyne and J. J. Carroll. Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation D2v1.0, W3C, 2004. Available from `http://www.w3.org/TR/rdf-concepts/`.
12. Rubén Lara, Holger Lausen, and Ioan Toma (*editors*). WSMX Discovery. Working Draft D10v0.2, DERI, 2005. Available from `http://www.wsmo.org/TR/d10/v0.2/`.
13. Rubén Lara, Dumitru Roman, Axel Polleres, and Dieter Fensel. A Conceptual Comparison of WSMO and OWL-S. In *European Conference on Web Services (ECOWS 2004), Erfurt, Germany*, pages 27–30, October 2004.
14. Massimo Paolucci, Katia P. Sycara, Takuya Nishimura, and Naveen Srinivasan. Using DAML-S for P2P discovery. In *ICWS*, pages 203–207, 2003.
15. Abhijit A. Patil, Swapna A. Oundhakar, Amit P. Sheth, and Kunal Verma. METEROR-S Web Service Annotation Framework. In *The Thirteenth International World Wide Web Conference Proceedings*, pages 553–562, 2004.
16. Dumitru Roman, Holger Lausen, and Uwe Keller. Web Service Modeling Ontology Standard (WSMO-standard). Working Draft D2v1.0, WSMO, 2004. Available from `http://www.wsmo.org/2004/d2/v1.0/`.
17. Mario Schlosser, Michael Sintek, Stefan Decker, and Wolfgang Nejdl. A Scalable and Ontology-Based P2P Infrastructure for Semantic Web Services. In *P2P '02: Proceedings of the Second International Conference on Peer-to-Peer Computing*, page 104. IEEE Computer Society, 2002.
18. Kaarthik Sivashanmugam, Kunal Verma, Amit P. Sheth, and John A. Miller. Adding Semantics to Web Services Standards. In *Proceedings of the International Conference on Web Services*, pages 395–401, 2003.
19. I. Stoica, R. Morris, D. Karger, M. Kaashock, and H. Balakrishman. Chord: A scalable peer-to-peer lookup protocol for internet applications. In *Proceedings of ACM SIGMOD Conference*, pages 149–160, San Diego, California, USA, 2001.
20. Kunal Verma, Kaarthik Sivashanmugam, Amit Sheth, Abhijit Patil, Swapna Oundhakar, and John Miller. METEOR-S WSDI: A Scalable P2P Infrastructure of Registries for Semantic Publication and Discovery of Web Services. *Journal of Information Technology and Management*, 2004. to be published.
21. Maciej Zaremba and Eyal Oren. WSMX Execution Semantics. Working Draft D13.2v0.2, WSMO, 2005. Available from `http://www.wsmo.org/TR/d13/d13.2/v0.2/`.