Federated Directories of Semantic Web Services

Michael Schumacher, Tim van Pelt, Ion Constantinescu,
Alexandre de Oliveira e Sousa and Boi Faltings
Artificial Intelligence Lab, École Polytechnique Fédérale de Lausanne (EPFL), Switzerland
URL: http://liawww.epfl.ch
FirstName.LastName@epfl.ch

ABSTRACT

This paper presents a federated directory system called WSDir[1], which allows registration and discovery of semantic web services. The system is designed and implemented as a federation: directory services form its atomic units, and the federation emerges from the registration of directory services in other directory services. Directories are virtual clusters of service entries stored in one or more directory services. To create the topology, policies are defined on all possible operations to be called on directories.

1. INTRODUCTION

This paper presents a federated directory system called WSDir, which allows registration and discovery of semantic web services. Its main functionality is to let heterogeneous service descriptions be registered and searched by certain clients. As such, it realizes a lookup function with basic retrieval schemes. The typical use of our directory system is in a context where ubiquitous application services should be flexibly coordinated and pervasively provided to the mobile user by intelligent agents in dynamically changing environments [1]. We modeled, designed and implemented our system as a backbone directory system to be searched by an infrastructure made up by such kind of agents coordinating web services (WS). This agent infrastructure therefore is deployed on mobile users: it queries our directory system for semantic WS descriptions, composes them to achieve a target higher functionality and executes them.

After introducing the WS entries that can be stored in WSDir, we explain our architecture. We then explain how to apply WSDir to setup a concrete network example.

2. SERVICE ENTRIES

We describe services using OWL-S[2] [3]. However, internally, WSDir stores service entries in the FIPA SL0 description language by translating an OWL-S service into SL0. The internal representation contains a subset of the information provided in the original service description. This information can be used to find matching services in the directory. In addition, the original service description in OWL-S is stored in a separate slot. This field is used to retrieve the original description.

A service entry in WSDir contains the following slots: i) *ServiceCategories*, which refers to an entry in some ontology or taxonomy of services; ii) *ServiceProfileURIs*, which contains a set of profile URIs that is referred to in the service description; iii) *ServiceProcessURI*, which is a process URI that is defined in the service description, and points to an externally stored, retrievable service process; iv) *ServiceGroundings*, which contains a set of full-text service groundings for the service; v) *OWLSServiceDescription*, which contains the original OWL-S service description as a full-text entry.

3. DIRECTORIES

A directory comprises a set of service entries which are managed by a collection of one or more directory services (DS). All service entries, including DS entries, are registered at a DS as belonging to a specific directory. Such, DSs can form an arbitrary organisational structure (peer-to-peer, hierarchy etc.). Specifically, a directory can contain other directories, and it is supported by one or more DSs.

Two different types of interactions exist: i) *Client-Directory* interactions, in which clients registering, deregistering, and querying the directory interact with DSs supporting the directory. They may or may not have any idea about the internals of the directory. ii) *Director-Directory* interactions, in which DSs supporting the directories interact with one another to perform the internal management of the directory (data propagation, federated queries).

The *Client-Directory* interactions are part of the base for the creation and maintenance of a *domain directory.* The *Directory-Directory* interactions are the base for the creation and maintenance of a network directory.

4. DIRECTORY SERVICES

Directory services provide a WS interface to a repository that holds service entries. The service entries in this store are all registered as belonging to a certain directory. The DS forms the atomic unit of the directory federation. It allows clients to register, deregister, modify and search registrations in its repository. These registrations include service

[1]This work has been supported in part by the European Commission under the project grant FP6-IST-511632-CASCOM.
[2]WSDir can be easily adapted for other WS descriptions.

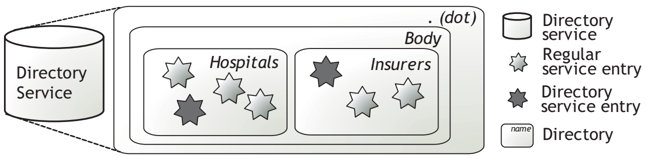

Figure 1: Directory system concepts

descriptions of services offered by clients as well as profiles of other DSs. By registering DSs in other DS stores, the system becomes federated. Figure 1 visually summarizes those concepts.

5. DIRECTORY OPERATIONS

The directory is able to handle five types of operations, which are are accessible remotely through a WS interface. The *register* operation enables a client to register a service description entry into a directory for a time period. The DS can also return a redirect message pointing to another directory where the client could try to register its object. The authentication of the requestor of the register operation can be guaranteed by the use of HTTPS in the invocation. The *deregister* operation de-registers a service that previously has been registered identified. The *modify* operation allows modifying a registered service. The *search* operation looks in the directory for services that match a template. The request can possibly be forwarded to supporting DSs, depending on the implemented search policy. As the internal service descriptions are expressed as SL0 expressions, the template used in the operation is also an SL0 expression. Search constraints can be specified in order to restrain the search: i) a deadline by which the constrained search should return the results; ii) a maximum depth of propagation of the search to federated directories, iii) a maximum number of results to be returned. Finally, the *get-profile* operation returns meta information on a DS.

6. POLICIES

DSs employ *directory policies* to regulate the operation of directories. Policies are defined per DS in the *DS profile* and determine the behaviour of a specific directory. Two types of policies can be distinguished: i) *Pro-active policies* are used for internal management of the directories. ii) *Reactive policies* assign a behaviour to combinations of directories and operations. They are executed whenever a bound operation is called. They are defined as a triple: *(directory name, operation, policy)*.

From the consequent application of policies, the network topology emerges. Policies can for example define how much entries can be registered per directory, which directories can be searched by which clients, and which types of services will be accepted. Each DS can define its own policies or use one of the pre-defined policies. A straightforward example of a pre-defined policy is the *child/sibling* policy: this policy forwards all operations to both the known children (DSs registered in the service store) as well as its known siblings.

7. A NETWORK ARCHITECTURE

WSDir allows to setup flexible distributed directory systems, especially thanks to the mechanism of policies. We used WSDir to build a specific network of DSs which are modelled as a virtual tree with multiple roots (see Fig. 2). This topology is currently being applied in the trial of a medical emergency usecase scenario.

The nodes of the tree are made up by the individual DSs. In the network, we have two types of directories: network directories and domain directories. *Network directories* are a reserved set of directories that are used for the construction of the network. In a directory federation, we can distinguish three different network directories: i) the *Hidden* network directory, which forms the root of the federation by registering the top-level nodes of the network; ii) the *Top* network directory, which is visible to the network as being one of the roots of the federation multi-rooted tree; iii) the *Body* network directory, which holds regular DSs that form the body of the multi-rooted tree.

Domain directories emerge from the registrations of service descriptions at the DSs that make up the directory system.

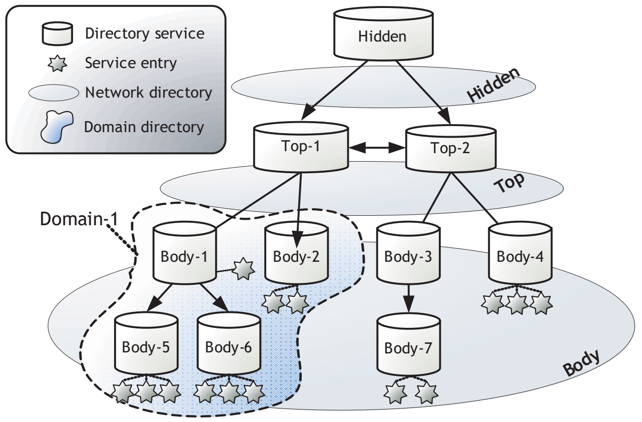

Figure 2: Network Topology

8. CONCLUSION

WSDir has been tested thoroughly in a real distributed setting spread over different countries. The system has proven to be scalable and very stable. We are currently working on a use case scenario in the pervasive ehealth domain that should use WSDir as its backbone. Future work could enhance security and finer-grained search queries by using a semantic matchmaker such as OWLS-MX [2].

9. REFERENCES

[1] C. Caceres and al. An abstract architecture for semantic service coordination in agent-based intelligent peer-to-peer environments. In *Proceedings of the 20th ACM 2006 Annual Symposium on Applied Computing (SAC-2006)*, 2006.

[2] M. Klusch and al. Owls-mx: Hybrid semantic web service retrieval. In *Proceedings 1st Intl. AAAI Fall Symposium on Agents and the Semantic Web*, Arlington VA, USA, 2005.

[3] D. Martin and al. Bringing semantics to web services: The owl-s approach. In *Proceedings of the First International Workshop on Semantic Web Services and Web Process Composition (SWSWPC 2004)*, 2004.