Composing Semantic Web Services with OWL-S for Location-Based Services

Ulf Rerrer
University of Paderborn, Department of Computer Science
Fuerstenallee 11, 33102 Paderborn, Germany
urerrer@upb.de

Abstract

In the area of Semantic Web Services the OWL-S specification defines a set of Ontologies through which a semantic description can be created. This work describes the semantic description of location-based Web services using OWL-S and how location-aware information can lead to an automatic service composition.

1 Introduction

User mobility led to the development and installation of numerous wireless networks and powerful wireless devices. With mobile computing activities and local-area wireless networks location identification has naturally become a critical attribute, as it has fostered a growing interest in location-aware systems and services. A key feature of so-called location-based systems is that the application information and/or interface presented to a user is a function of his or her physical location, especially within buildings. The Web offers organisations the possibility to reach potential clients through specialised services dynamically matching providers with requestors. Recent work concerning Web Services (WSs) and the Semantic Web is setting up the way for realizing dynamic discovery and utilization of Web resources.

Our goal is to introduce a semantic description for location-based Web services using the OWL-S specification. Gathering automatically location information can be used to easily compose new location-aware services. This work identifies shortcomings and offers recommendations for modeling locations and LBSs in the OWL-S language.

2 Location-Based Services

The ability to identify the exact physical location of a mobile user opens the door to a new world of innovative services, which are commonly referred to as *Location-based Services* (LBS) [Barnes, 2003]. User location is an important dimension in this new data-service world and Wireless LAN can be used for communication and (indoor) positioning simultaneously. Therefore, no additional hardware such as GPS receivers is needed. With up-to-date algorithms and techniques a sufficient positioning accuracy for LBSs can be achieved [Bahl and Padmanabhan, 2000].

2.1 LBS Workflow

In our workflow a user with a mobile device enters a building equipped with a WLAN infrastructure for communication and a LBS system. When the user moves within the range of one or more WLAN access points his wireless device is recognized by the system. Standard WLAN connectivity procedures care for authentication, IP addresses distribution, etc. A splash screen technique – well known from airport lounges or other commercial WiFi hot spots – shows a welcome screen as first homepage in a browser advertising the local services. After picking one or more services this selection is stored in the LBS system and a simple interaction via browser takes place like using other standard WSs.

For example the user wants to go to the conference room of his imminent meeting. Therefore, he uses the map service entering his target room and a map of the building appears with the shortest route from his current position to the conference room. On the way the user wants to print the latest version of his slides for his presentation in the meeting. He uses the printing service which determines the nearest available printer matching the requirements to print colored slides.

For the meeting a file sharing service is activated. During the meeting all participants in the conference room – and no one else from outside – can easily share their relevant documents. This is achieved by a service providing all users in this room a restricted, but easily accessible shared web space.

2.2 Architecture

The above scenario denotes the power and comfort of LBSs. The *central component* of the LBS architecture (Figure 1) is needed to register the users of such a system and coordinate the communication between users and services. A *location component* determines the position of a wireless device via triangulation of the received signal strength from different access points. The *map-server* for example is a basic service implementing a map service generating dynamically multi-layered vector-maps with miscellaneous objects and their positions. Rooms, floors, devices with dynamic properties and users can be displayed. Even routing between objects can be done. The *other services* have an interface for basic communication and interaction with the main component. Interfaces create service specific connections directly to the client. With the help of these interfaces services like the file sharing service are possible.

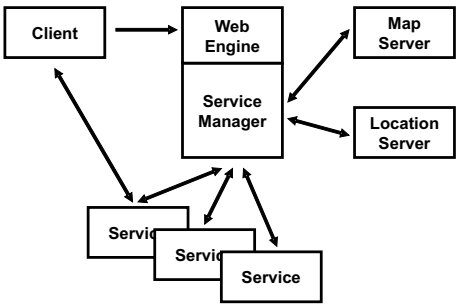

Figure 1: LBSs system architecture

3 Creating OWL-S Descriptions

A key component of the Semantic Web Services [McIlraith *et al.*, 2001] is the creation of a language for describing WSs. OWL-S [1] is such a language, adding rich typing and class information that can be used to describe and constrain the range of Web service capabilities much more effectively than XML data types do. Furthermore, it integrates rich class representations with a process model designed to capture not only the control and data flow, but also real-world side effects – preconditions and effects – of WSs. Well-defined semantics allow automated manipulation of Web service structures with a known outcome using powerful tools and reasoning techniques. This makes OWL-S well suited for LBSs.

Central to an OWL-S process model is the specification of a service's inputs, outputs, preconditions, and effects (IOPE's). To realise services like a location-service, printing-service, file-sharing-service as described above, it is necessary to define several atomic processes. In the OWL-S process model these small and mostly basic processes can be used individually or composed in a variety of ways to complex services. For the example of a location-aware file-sharing-service this could be done with five services that are used in this part of the scenario are:

- *UploadDocument* (file import service)
- *DownloadDocument* (file export service)
- *SetRepository* (defining a private web space)
- *CheckAccessRange* (checking user permissions)
- *WebSpacePortal* (presenting repository contents)

Currently to use these services it is necessary to manually invoke them individually in the desired order. Our goal is to automatically configure and compose these services and especially describe and use their location-aware nature properly. To add semantics to the Web service descriptions the four Ontologies that constitute OWL-S are used: service, profile, model, and grounding.

Service Ontology
This Ontology specifies the highest level OWL-S concept: a service. It specifies the relation of the service class to other classes. Each of these describes the service from a certain point of view using concepts defined in the corresponding OWL-S Ontology.

[1] OWL-S, `http://www.daml.org/services/owl-s/`

Profile Ontology
The profile Ontology contains the vocabulary to describe the service. It contains information of providers, a set of service characteristics and a functionality description by specifying the inputs, outputs, preconditions and effects of the service. Experiments showed the need for a location Ontology beyond just address, city and country. Our approach uses common parts of different Ontologies, due to shared location information to connect location-aware WSs. For the exemplary file sharing service the input is the file to be uploaded, it checks the location for service access, contains references and is provided as a URL. The output is handled similar.

Model Ontology
The service model is described by a process model in the model Ontology. The process model has a single process which can be atomic, simple or composite which defines how the Web service works in its interior. A composite process can be composed through various control constructs such as split, unordered, iterate, etc. Each Process has a set of IOPE's which are modeled as properties of the process and take values that describes them.

Grounding Ontology
To ground an OWL-S specification to WSDL, each atomic process defined in the process model is grounded to a WSDL operation. Also each input/output of the process becomes a message part in the input/output message of the corresponding operation. The WSDL file specifies the type of its message parts in terms of XML data types, the transport protocol, the messaging format, and the actual address of the service.

4 Conclusions

Starting from current interest in WSs to develop a modular architecture for LBSs in WLANs, we explored Semantic Web issues and the composition problem. In summary the OWL-S description language is sufficiently expressive for specifying conceptual and operational building blocks for the location-aware services in our scenario. In the presented scenario the OWL-S descriptions provided sufficient information for basic reasoning purposes. The profile models provided descriptions of the components are the conceptual level and the grounding model described the service at the implementation level. Data types were also specified at both levels through the definition of inputs and outputs and various external Ontologies. However, it would be useful to see review and possible extensions to OWL-S to support a more details location description to make automatic composing and reasoning possible.

References

[Bahl and Padmanabhan, 2000] P. Bahl and V. N. Padmanabhan. Radar: An in-building rf-based user location and tracking system. In *Proceedings of IEEE InfoCom*, pages 775–784, March 2000.

[Barnes, 2003] S. J. Barnes. Location-based services: The state of the art. *e-Service Journal*, 1:59–70, 2003.

[McIlraith *et al.*, 2001] S. McIlraith, T.C. Son, and H. Zeng. Semantic web services. *IEEE Intelligent Systems*, 16(2):46–53, April 2001.