UNIVERSAL, ONTOLOGY-BASED DESCRIPTION OF DEVICES AND SERVICES

Mihail Bratoev and Boyan Bontchev
Department of Software Engineering, Faculty of Mathematics and Informatics
Sofia University5, James Bourchier blvd., 1164 Sofia, Bulgaria

Keywords: Device ontology, Service description, *OWL-S*, Service discovery.

Abstract: A great problem with existing service discovery middleware solutions (e.g. *UPnP*, *Bluetooth*, *and Jini* etc) is that services available in one platform cannot be easily accessed from services based on another platform which is known as "*hidden service problem*". One of the reasons for this incompatibility is a different method used for service description in each of these platforms. This paper highlights the importance of unified, platform-neutral, semantic-oriented device and service descriptions to achieve seamless integration between different technologies and systems for advertising, discovering and invoking services. Moreover, it presents an ontology-driven approach for describing devices and their services which will enable service-seeking peers to reason about available services and devices, and make intelligent and informed decisions regarding which services to use, and how. We argue that having separate descriptions (ontologies) for device and services that this device offers aims to be more flexible, modular and therefore better design. The proposed device ontology is to be used for description of characteristics of the device that offers a specific service. For development of semantic models of services our analysis shows that making several enhancements of *OWL-S* (using *OWL* subclassing) will enable its use as common language for service description. Finally we outline some of the future work that gives a real perspective of the proposal.

1 INTRODUCTION

The widespread deployment of inexpensive communications technology and network-enabled end devices poses an interesting problem for end users: how to locate a particular network service or device out of hundreds of thousands of accessible services and devices. Service offers have to be explicitly described and published, whereas service requests need to be paraphrased and compared with the offer descriptions. For this reason, what is needed is a dedicated service description language.

Service oriented approach to design and implement modular, loosely-coupled distributed software architectures is supported by special middleware technologies for service advertisement, discovery and control. The main problem with the existing service discovery technologies is that as a whole, these technologies remain incompatible each other. That leads to the so called isolated "islands" of interoperability (Nakazawa J., 2006), where it is possible one service to be available in general but not to be discoverable by the mechanism used by its potential clients. Since different discovery systems use different formats for service description, we believe that providing a mean for automated translation between different service description formats (languages) used by these technologies is an important step towards seamless integration and mutual understanding across platforms. Such a mean should possess good abilities for semantic expressivity required for an adequate representation of requirements and capabilities of system services. Interoperability at semantic level will significantly improve service reuse and discovery and, as well, strongly assist composition of new services and enable business process integration. Generally speaking, there are two possible ways for translation of service descriptions – *direct* and *mediated* (Nakazawa J., 2005). Mediated translation uses a common intermediate language for protocol independent description of services and is considered more practical approach where a lot of target language is available. The main disadvantage hete – a possible loss of semantics during translation

Bratoev M. and Bontchev B.
UNIVERSAL, ONTOLOGY-BASED DESCRIPTION OF DEVICES AND SERVICES.
DOI: 10.5220/0001843001050110
In *Proceedings of the Fifth International Conference on Web Information Systems and Technologies (WEBIST 2009)*, page
ISBN: 978-989-8111-81-4

– could be overcome by using comprehensive, semantically-rich description language.

Until the moment, there are published several proposals for languages and systems supporting universal service descriptions (Bandara A., 2004), (OWL-S, 2007), (Sun, 2007). Universal frameworks such as uMiddle (Nakazawa J., 2006) try to enable seamless device interaction over diverse middleware platforms. However, they fail in achieving flexible and adequate interoperability between environments based on different service technologies. Creation of languages as USQL (Unified Service Query Language) is quite prominent as this is a way to enable service requestors to interact with heterogeneous service registries, for the discovery of available services (Pantazoglou, 2006).

2 REQUIREMENTS FOR SERVICE DESCRIPTION LANGUAGE

Service descriptions can be done in various levels of detail. They range from simple keyword-based to highly complex ontology-based approaches and many different languages have been proposed in the last years. We can distinguish four clusters of description languages: keyword-based, template-based, object-based and ontology-based descriptions (Klein M., 2003). As presented in figure 1, the lower is the semantic expressiveness the higher is the ability of such a language for automated comparison. While keyword-based descriptions are far from semantic context representation, template-based approaches - such as WSDL, SLP (Bratoev M., 2006) and UPnP (Microsoft, 2007), and object-based representations - like these used in CORBA and Jini (Sun, 2007), try to structure service descriptions into templates and objects. However, they provide means for describing non-functional service aspects while functional semantics have to be still derived from textual descriptions, names and parameters. Finally, ontology-based approaches - such as RDFS, DAML-S (Defense, 2006) and OWL-S (OWL-S, 2007), succeed in providing higher expressiveness by trying to define an upper extendible ontology for all types of services where there are proposed general service descriptions for both non-functional and functional service attributes such as input/output parameters and result effect.

The definition of a powerful, protocol independent, language for service description is vitally important for implementation of mediated translation. In principle, this language should be highly expressive and able to represent every concept in a set of real service description languages. It must allow automatic comparison of service descriptions without any need for additional human analysis. It also has to be flexible and adaptable to changes in order to reflect the evaluation of real description formats.

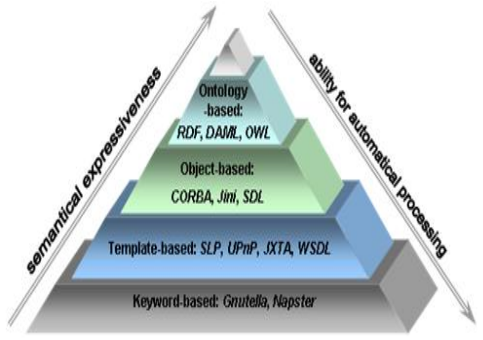

semantical expressiveness
ability for automatical processing
Ontology -based: RDF, DAML, OWL
Object-based: CORBA, Jini, SDL
Template-based: SLP, UPnP, JXTA, WSDL
Keyword-based: Gnutella, Napster

Figure 1: Classification of service description languages.

A good service description language has the following important characteristics:

- *Semantically expressive* - service descriptions must be expressive and flexible, scaling to future device types and providing support for reasoning about services.
- *Automatically comparable* – the ability for automatic comparison of service descriptions is essential and implies well structured descriptions
- *Flexible and extensible* – as the information that should be included in the service description changes the language must be able to accommodate the new reality
- *Editable* – descriptions should be easily created and modified by humans, possibly by using special-purpose editors.

Our vision for such a language is that it should be ontology-based. The particular format for service description should be able to represent various properties of the service such as its interface and attributes as well as its dynamic behavior. When examining requirements about semantic expressiveness, we naturally come to the conclusion that only ontology-based description languages can provide enough expressiveness for an automatic service trading. Describing services using ontology is superior to using other forms of data structures such as service templates etc. used in current standards, because that method provides a structure that makes it possible to reason about and derive knowledge from the given descriptions. These semantically rich descriptions enable automated machine reasoning over service and domain descriptions, thus supporting automation of service discovery, composition, and execution, and reducing manual configuration and programming efforts.

Some services specifically support, or are offered by *devices* (i.e. hardware components), whereby the characteristics/capabilities of the device may play an integral role in the description and behavior of the services it offers. When a service involves a hardware device (for example printing service, scanning service) some level of detail about the device in which it is hosted will be required for service selection purposes.

Capabilities of the device may play an integral role to both the description and behavior of the services it offers. Such information relating to the device could be included along with the service description itself (in the service ontology), but having separate ontologies to describe devices and services promotes ease of use, readability and reusability and is therefore a better design (Bandara A., 2004). By describing devices capabilities, the device ontology may become useful in cases of service composition where is needed to get knowledge about the capabilities of available devices (such as resource capability, device proximity to individual services, etc.) in order to determine a broker platform.

3 THE DEVICE ONTOLOGY

There is a great variety in the way ontologies are created, and an ongoing discussion in the ontology community about the best practices for ontology development. One of the greatest challenges in constructing an ontology is the lack of formal standards or consensual methodology. Nevertheless, there are some processes that should be addressed. In practical terms, developing an ontology includes (Noy N., 2001):

- Defining classes in the ontology
- Arranging the classes (concepts) in a taxonomic (subclass/superclass) hierarchy
- Defining slots (attributes of the classes) and describing allowed values for these slots
- Filling in the values for slots for instances

Table 1: Device concepts.

Device concept		Service Discovery System		
		Jini	Universal Plug And Play (UPnP)	Service Location Protocol (SLP)
1. Name		√	√	
2. Model	Name	√	√	
	Description		√	
	Number		√	
	URL		√	
3. Type		√	√	√
4. Vendor	Name	√	√	
	URL		√	
5. Identity	Serial Number	√	√	
	Universal Device Name		√	
	Universal Product Code (UPC)		√	

To create the device ontology we have tried to enumerate all the important terms for the device – *hardware*, *software*, *service*, *embedded device*, *serial number* etc. From this list we select the terms that describe objects having independent existence rather than terms that describe these objects. These terms will be classes in the ontology and will become anchors in the class hierarchy. On the other hand, we identified all device-related properties (table 1) that are incorporated in service description of the three real target languages. For illustrative purposes and without violating the general applicability of our approach we will take three discovery technologies (*Jini*, *UPnP* and *SLP*) and carefully examine the service and device description concepts used by each of them. These technologies are mutually orthogonal in respect to their service description approach (respectively, object oriented, template-based and keyword-based) to minimize distortion of analysis results. For each property in the list, we must determine which class it describes. If we cannot find semantically relevant class to add this property (slot) to we create a new class. To incorporate the information from table 1 in the device ontology we created one class named *Description*. This class aggregates all the information that can be used for device discovery and advertisement. This class has a number of subclasses that group a set of semantically related properties (for example, class *Vendor* has properties *Vendor Name* and *Url*).

The produced device ontology is shown on figure 2. The *Service* class provides the information about the service(s) hosted on the device concerned. The

OWL-S could be potentially used to describe these individual services.

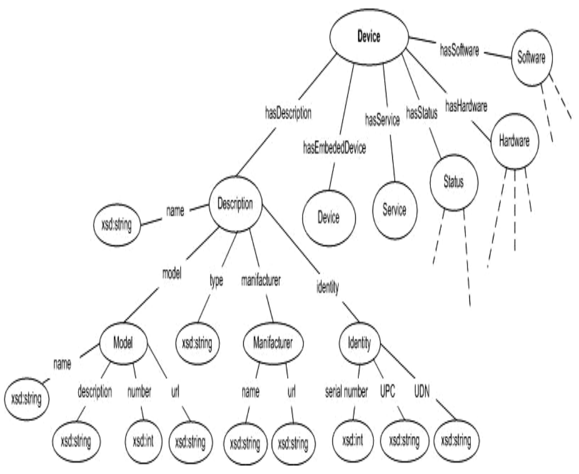


Figure 2: The proposed device ontology.

4 THE SERVICE ONTOLOGY

When starting to develop ontology, it is highly recommended to consider existing ontologies in the same or similar domain (Noy N., 2001). Most prominent examples of ontology-based service languages are OWL-S (OWL-S, 2007), i.e. OWL-based (Ontology Web Language) web service ontology (formerly DAML-S (Defense, 2006)), and SAWSDL (W3C, 2007):

- OWL-S markup of Web services facilitates the automation of Web service tasks, including automated Web service discovery, execution, composition and interoperation (OWL-S, 2007);
- SAWSDL enables semantic annotations for Web services not only for discovering Web services but also for invoking them (W3C, 2007).

The most prominent ontology-oriented language intended especially for construction of web service descriptions is OWL-S. It is on ontology of service concepts and provides a core set of markup language constructs for describing the properties and capabilities of a Web service in unambiguous, computer interpretable form. Therefore, it provides an upper ontology for all Web services, is extremely generic and organizes a service description into four conceptual areas (fig. 3):

- *Service* - what is the service itself?
- *Process model* - what does the service provide for prospective clients?
- *Profile* - how is the service used?
- *Grounding* - how does one interact with it, by which transport protocols?

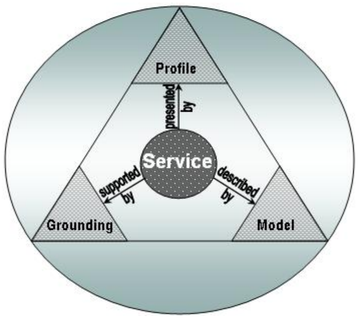
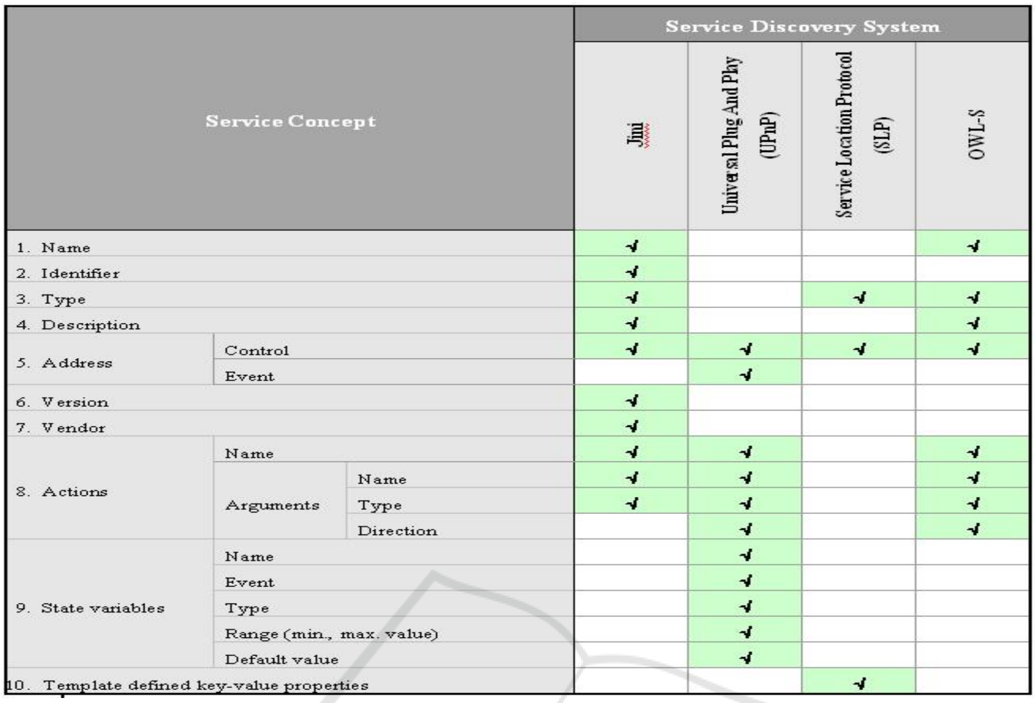


Figure 3: Top level representation of the service ontology.

An OWL-S Profile describes a service as a function of three basic types of information: provider description consisting of contact information about the service provider, functional description in terms of the transformation produced by the service (required inputs, generated outputs, preconditions and expected effects) and, finally, features of the service - category, quality rating of the service, and an unbounded list of service parameters of any type. The service profile is used to characterize a service for purposes such as advertisement, discovery, and selection. Service profiles may be published in various kinds of registries, discovered using various tools, and selected using various kinds of matchmaking techniques. OWL-S 1.2 defines a Process (i.e., a perspective on how to interact with a service) as a subclass of ServiceModel, where outputs and effects can depend on conditions that hold true of the world state at the time the process is performed. An OWL-S process may be atomic or composite, while the control constructs may be Sequence, Split, Choice, Condition, If-Then-Else, Iterate, etc. (OWL-S Coalition., 2007). While both the ServiceProfile and the ServiceModel are thought of as abstract representations, the ServiceGrounding deals with the concrete level of specification.

There are two main tasks in the development of OWL-S services: (1) to define the service's domain ontologies (in terms of OWL classes, properties, and instances), and (2) to create an OWL-S service description relating to the domain ontologies and consisting of instances of OWL-S classes such as Service, Process, Input, and Output. Unfortunately, OWL-S is not directly applicable for services accessible by all existing service discovery systems. The lack of concept for service state and events (which are available in UPnP for example) as well as for some service properties such as type, identifier

Table 2: Service concepts.



Service Concept			Service Discovery System			
			Jini	Universal Plug And Play (UPnP)	Service Location Protocol (SLP)	OWL-S
1. Name			✓			✓
2. Identifier			✓			
3. Type			✓		✓	✓
4. Description			✓			✓
5. Address	Control		✓	✓	✓	✓
	Event			✓		
6. Version			✓			
7. Vendor			✓			
8. Actions	Name		✓	✓		✓
	Arguments	Name	✓	✓		✓
		Type	✓	✓		✓
		Direction		✓		✓
9. State variables	Name			✓		
	Event			✓		
	Type			✓		
	Range (min., max. value)			✓		
	Default value			✓		
10. Template defined key-value properties					✓	

and version in OWL-S ontology impose the need of extending it in order to handle these specific modeling situations. Table 2 below summarizes the availability of a specific service description concept in each of these discovery systems. It is clear that OWL-S is not directly applicable for services accessible by all existing service discovery systems. The lack of concept for service state and events (which are available in UPnP for example) as well as for some service properties such as type, identifier and version in OWL-S ontology impose the need of extending it in order to handle these specific modeling situations. Every service concept available in one or more of the analyzed real description languages but missing in OWL-S framework should be associated and included to extended service profile or extended service model producing the following revised OWL-S service description ontology. Since OWL-S provides a good starting point for our work, by providing a set of concepts for modeling basic aspects of services there are some service concepts in the examined technologies that are not covered by OWL-S in its pure (original) form. That's the reason it cannot play the role of unified common description language that is able to describe any service without loss of information. Therefore, it is important to incorporate into OWL-S information about missing concepts. That is the reason making us to propose several enhancements to OWL-S semantics.

The proposed extensions of the Profile and Model schemes are *not* meant to provide canonical and universal ontology for use on all service discovery systems, but are meant merely to illustrate technical features of this approach. Ultimately, these enriched elements are intended to describe more precisely the service and its functionalities. It is up to service providers who create the service profiles and models to decide whether to take advantage of them and how. The unified ontology (figure 4) together with a set of particular instances of its classes represents a service description.

5 CONCLUSIONS AND FUTURE WORK

In this article we proposed an ontology-oriented perspective for device and service description in a formal, platform-independent manner. Such a description is essential for suppressing the integration crisis between different service discovery systems. The presented device ontology and OWL-S enhancements (the new Service Profile and Service Model subclasses) contain the all information for a service that can be found in Jini, UPnP or SLP and is

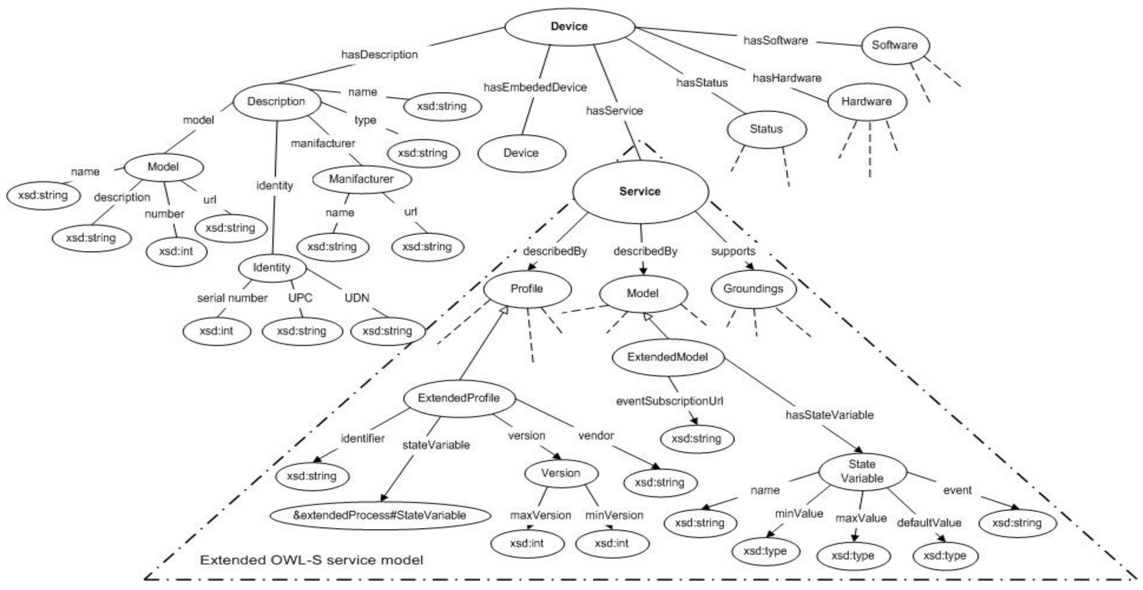


Figure 4: The unified ontology for device and service description.

important for achieving a common semantic model of a service (device). supported by the sister project (ec fp7-sp4 capacities agr. no. 205030), we are creating a prototype application (employing owl-s api) that translates a service description from jini to upnp format using our framework. the specified extensions and the device ontology are constructed using protégé and its open source ontology owl editor and demonstrate the feasibility and effectiveness of the proposed ontology. furthermore, we plan to investigate mechanisms that enable users to specify discovery preferences in an unobtrusive manner.

REFERENCES

Bandara A., Payne T., 2004, An Ontological Framework for Semantic Description of Devices (Poster), In *Proceedings of International Semantic Web Conference (ISWC)*, Hiroshima, Japan.

Bratoev M., Bontchev B., 2006, Comparison of Service Discovery Protocols, In *Journal of Information Technologies and Control*, October 2006, pp 47-52.

Defense Advanced Research Projects Agency 2006 *DARPA agents markup language – services (DAML-S)*, http://www.daml.org

Klein M., Köning-Ries B., 2003, A Process and a Tool for Creating Service Descriptions based on DAML-S, In *LNCS "Technologies for E-Services",* Vol. 2819, 2003, pp.143-154.

Microsoft Corp., 2007, *Universal Plug and Play*, http://www.upnp.org

Nakazawa J., Yura J., Tokuda H., 2006, *uMiddle: A Universal Framework for Bridging Diverse Middleware Platforms*, White paper, Georgia Institute of Technology, 2006.

Nakazawa J., Keith Edwards W., Ramachandran U., Tokuda H., 2006, A Bridging Framework for Universal Interoperability in Pervasive Systems, In *Proc. of the 26th Int. Conf. on Distributed Computing Systems (ICDCS)*, July 04-07, 2006, p.3.

Noy N., McGuinness D., 2001, Ontology Development 101: A Guide to Creating Your First Ontology, *Stanford Knowledge Systems Laboratory Technical Report KSL-01-05 and Stanford Medical Informatics Technical Report SMI-2001-0880*

Ong S., Tang Kong., 2005, Service Description and Composition Using Ontologies, Proc. of *IASTED Int. Conf. On Artificial Intelligence and Applications*, Innsbruck, Austria, 14-16 February, 2005.

OWL-S Coalition., 2007, *OWL-S 1.2 Pre-Release*, http://www.ai.sri.com/daml/services/owl-s/1.2/

Pantazoglou M., Tsalgatidou A., Athanasopoulos G., Pilioura, T., 2006, A Unified Approach towards the Discovery of Web and Peer-to-Peer Services, In *Proc. of the IEEE Int. Conf. on Web Services*, ICWS 2006, Chicago, USA, September 18-22, 2006, pp.901-902.

Sun Microsystems., 2007, *Jini*, http://www.jini.org

Uschold M., Gruninger M., 1996, Ontologies: Principles, Methods and Applications, *Knowledge Engineering Review 11(2).*

W3C, 2007 Semantic Annotations for WSDL and XML Schema, W3C Working Draft, April 10, 2007, http://www.w3.org/TR/sawsdl/