A Universal Service Description Language

Luke Simon, Ajay Mallya, Ajay Bansal, Gopal Gupta
Department of Computer Science
University of Texas at Dallas
Richardson, TX 75083

Thomas D. Hite
Metallect Corp
2400 Dallas Parkway
Plano, TX 75093

Abstract

To fully utilize web-services, users and applications should be able to discover, deploy, compose and synthesize services automatically. This automation can take place only if a formal semantic description of the web-services is available. In this paper we present a markup language called USDL (Universal Service Description Language), for formally describing the semantics of web-services.

1 Introduction

The next milestone in the Web's evolution is making *services* ubiquitously available. A web service is a program available on a web-site that "effects some action or change" in the world (i.e., causes a side-effect). Examples of such side-effects include a web-base being updated because of a plane reservation made over the Internet, a device being controlled, etc. As automation increases, these web-services will be accessed directly by the applications themselves rather than by humans. In this context, a web-service can be regarded as a "programmatic interface" that makes application to application communication possible. For web-services to become practical, an infrastructure needs to be supported that allows users to discover, deploy, compose, and synthesize services automatically. Such an infrastructure *must* be semantics based so that applications can reason about a service's capability to a level of detail that permits their discovery, deployment, composition and synthesis.

Approaches such as WSDL are purely syntactic in nature, that is, they merely specify the format of the service. Our approach can be regarded as providing semantics to WSDL statements. We present a language called Universal Services Description Language (USDL) which service developers can use to specify formal semantics of web-services. Thus, if WSDL can be regarded as a language for formally specifying the syntax of web services, USDL can be regarded as a language for formally specifying their semantics. USDL can be thought of as formal program documentation that will allow sophisticated conceptual modeling and searching of available web services, automated composition, and other forms of automated service integration. For example, the WSDL syntax and USDL semantics of web services can be published in a directory which applications can access to automatically discover services. That is, given a formal description of the context in which a service is needed, the service(s) that will precisely fulfill that need can be determined. The directory can then be searched to check if this exact service is available, and if not available, then whether it can be synthesized by composing two or more services listed in this (or another) directory.

The design of USDL rests on two formal languages: Web Services Description Language (WSDL) [3] and Web Ontology Language (OWL) [2]. The Web Services Description Language (WSDL) [3], is used to give a syntactic description of the name and parameters of a service. The description is syntactic in the sense that it describes the formatting of services on a syntactic level of method signatures, but is incapable of describing what concepts are involved in a service and what a service actually does, i.e. the conceptual semantics of the service.

USDL can be regarded as as a merger of WSDL and OWL that yields a language capable of describing the syntax and semantics of web services. WSDL will be used to describe message formats, types, and method prototypes, while a specialized universal OWL ontology will be used to formally describe what these messages and methods mean, on a conceptual level.

2 USDL

As mentioned earlier, USDL can be regarded as the semantic counterpart to the syntactic WSDL description. WSDL documents contain two main constructs to which we want to ascribe conceptual meaning: messages and ports. These constructs are actually aggregates of service components which will actually be directly ascribed meaning. Messages consist of typed parts and ports consist of operations parameterized on messages. USDL defines OWL surrogates or proxies of these constructs in the form of classes, which have properties with values in the OWL WordNet ontology. Furthermore, the USDL surrogates for WSDL operations allow for the description of the side-effect of executing the operation, in addition to the semantics of the input and output parameters of the operation, which are also mapped to the WordNet.

Using an OWL WordNet ontology allows for our solution to use a universal, complete, and tractable framework, which lacks the semantic aliasing problem, to which we map web service messages and operations. The semantic aliasing problem occurs when the formal semantics distinguishes two identical concepts. Other approaches such as OWL-S suffer from this problem [1, 8]. As long as this mapping into the WordNet ontology is precise and sufficiently expressive, reasoning can be done within the realm of OWL by using an automated inference systems (such as, one based on description logic), and we automatically reap the wealth of semantic information embodied in the OWL WordNet ontology that describes the relationships between ontological concepts, especially subsumption (hyponym) and equivalence (synonym) relationships. Finally, USDL restricts its conceptual constructors, in order to address the problems of semantic aliasing and tractability.

At present, we presume only four side-effect specific operations (find, create, delete, update) which have an ontology member as their "target"; this set of basic operations may be extended as more experience is gained with USDL. In practice, most services, however, deal with manipulating databases and for such services these four operations are sufficient. As stated, one of the reasons for limiting the side-effect types is to safe-guard against the semantic aliasing problem. This is also one of the main reasons for restricting the combining forms in USDL to conjunction, disjunction, and negated atoms [8].

3 Applications

With a pool of USDL services at one's disposal, rapid application development (RAD) tools could be used to aid a systems integrator with the task of creating composite services, i.e., services consisting of the composition of already existing services. The service designer could use such a RAD tool by describing the desired service via a USDL document, and then the tool would query the pool of services for composable sets of services that can be used to accomplish the task as well as automatically generate boilerplate code for managing the composite service, as well as menial inter-service data format conversions and other glue. Of course these additional RAD steps would require other technologies already being researched and developed [4, 5, 7].

4 Conclusion

The current version of USDL incorporates current standards in a way to further aid markup of IT services by allowing constructs to be given meaning in terms of an OWL based WordNet ontology. Future work involves the application of USDL to formally describe commercial service repositories (for example SAP Interface Repository and services listed in UDDI), as well as to service discovery and rapid application development (RAD) in commercial environments [6].

References

[1] Semantic markup for web services. `http://www.daml.org/services/owl-s/1.0/owl-s.html`.

[2] Web ontology language reference. `http://www.w3.org/TR/owl-ref`.

[3] Web services description language. `http://www.w3.org/TR/wsdl`.

[4] P. A. Bernstein. Applying model management to classical meta data problems. In *CIDR*, pages 209–220, 2003.

[5] C. E. Gerede, R. Hull, O. H. Ibarra, and J. Su. Automated composition of e-services:lookaheads. In *ICSOC*, 2004.

[6] T. Hite. Service composition and ranking: A strategic overview. Metallect Inc., 2005.

[7] R. Hull and J. Su. Tools for design of composite web services. In *SIGMOD*, 2004.

[8] L. Simon, A. Mallya, A. Bansal, G. Gupta, and T. Hite. A universal service description language. Technical Report UTDCS-09-05, University of Texas at Dallas, 2005.