Chapter 2

USDL: A Service–Semantics Description Language for Automatic Service Discovery and Composition¹

Srividya Kona
The University of Texas at Dallas, USA

Ajay Bansal
The University of Texas at Dallas, USA

Luke Simon
The University of Texas at Dallas, USA

ABSTRACT

Web services and Service-Oriented Computing is being widely adopted. In order to effectively reuse existing services, we need an infrastructure that allows users and applications to discover, deploy, compose, and synthesize services automatically. This automation can take place only if a formal description of the Web services is available. In this article we present an infrastructure using USDL (Universal Service-Semantics Description Language), a language for formally describing the semantics of Web services. USDL is based on the Web Ontology Language (OWL) and employs WordNet as a common basis for understanding the meaning of services. USDL can be regarded as formal service documentation that will allow sophisticated conceptual modeling and searching of available Web services, automated service composition, and other forms of automated service integration. A theory of service substitution using USDL is presented. The rationale behind the design of USDL along with its formal specification in OWL is presented with examples. We also compare USDL with other approaches like OWL-S, WSDL-S, and WSML and show that USDL is complementary to these approaches.

DOI: 10.4018/978-1-61350-104-7.ch002

¹The University of Texas at Dallas, USA

Gopal Gupta
The University of Texas at Dallas, USA

Thomas D. Hite
Metallect Corp., USA
INTRODUCTION

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, and so forth. The next milestone in the Web’s evolution is making services ubiquitously available. 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. An infrastructure that allows users to discover, deploy, synthesize and compose services automatically needs to be supported in order to make Web services more practical.

To make services ubiquitously available we need a semantics-based approach such that applications can reason about a service’s capability to a level of detail that permits their discovery, deployment, composition and synthesis. Several efforts are underway to build such an infrastructure. These efforts include approaches based on the semantic web (such as OWL-S (OWL-S, 2003)) as well as those based on XML, such as Web Services Description Language (WSDL (WSDL, 2001)). Approaches such as WSDL are purely syntactic in nature, that is, they merely specify the format of the service. In this article we present an approach that is based on semantics. Our approach can be regarded as providing semantics to WSDL statements. We present the design of a language called Universal Service-Semantics Description Language (USDL) which service developers can use to specify formal semantics of Web services (Bansal et al., 2005; Simon, Bansal, Mallya, Kona, Gupta, & Hite, 2005). 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 service 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 for the exact service, or two or more services that can be composed to synthesize the required service.

To provide formal semantics, a common denominator must be agreed upon that everybody can use as a basis of understanding the meaning of services. This common conceptual ground must also be somewhat coarse-grained so as to be tractable for use by both engineers and computers. That is, semantics of services should not be given in terms of low-level concepts such as Turing machines, first-order logic and their variants, since service description, discovery, and synthesis then become tasks that are practically intractable and theoretically undecidable. Additionally, the semantics should be given at a conceptual level that captures common real world concepts. Furthermore, it is too impractical to expect disparate companies to standardize on application (or domain) specific ontologies to formally define semantics of Web services, and instead a common universal ontology must be agreed upon with additional constructors. Also, application specific ontologies will be an impediment to automatic discovery of services since the application developer will have to be aware of the specific ontology that has been used to describe the semantics of the service in order to frame the query that will search for the service. The danger is that the service may not be defined using the particular domain-specific ontology that the application developer uses to frame the query, however, it may be defined using some other domain-specific ontology, and so the application