1 Related Work

Industry efforts to develop standards for electronic commerce, and in particular for the description of Web-based services currently revolve around UDDI, WSDL, and ebXML. There have also been company-specific initiatives to define architectures for e-commerce, most notably E-speak from Hewlett-Packard.

Nevertheless, we believe that DAML-S provides functionality that the other efforts do not. In comparison to the DAML-S characterization of services, the industry standards mostly focus on presenting a *ServiceProfile* and a *ServiceGrounding* of services (to use DAML-S terminology). *ServiceGroundings* are supported by all the standards. However, they are limited with respect to DAML-S profiles in that they cannot express logical statements, e.g. preconditions and postconditions, or rules to describe dependencies between the profile elements. Input and output types are supported to varying extents. Furthermore, DAML-S supports the description of certain functional attributes of services, which are not covered in the other standards, such as qualityGuarantees and serviceType.

With respect to the four tasks of automatic Web service discovery, automatic Web service invocation, automatic Web service interoperation and composition, and automatic Web service execution monitoring that DAML-S is meant to support, the standards primarily enable the first and the second tasks to a certain extent. These standards are still evolving and it is unclear at present to what extent composition will be addressed. At the moment, the standards do not consider the *ServiceModel* of a service and thus, they also do not support execution monitoring, as defined in this paper.

In the following sections, we look in greater detail at each of these technologies in turn and compare them to DAML-S.

1.1 UDDI

UDDI(Universal Description, Discovery and Integration) is an initiative begun by Microsoft, IBM and Ariba to develop a standard for an online registry, to enable the publishing and dynamic discovery of Web services offered by businesses [1]. UDDI allows programmers and other representatives of a business to locate potential business partners and form business relationships on the basis of the services they provide. It thus facilitates the creation of new business relationships.

The primary target of UDDI seems to be integration and at least semi-automation of business transactions in B2B e-commerce applications. It provides a registry for registering businesses and the services they offer. These are described according to an XML schema defined by the UDDI specification. A Web service provider registers its advertisements along with keywords for categorisation. A Web services user retrieves advertisements out of the registry based on keyword search. The UDDI search mechanism relies on pre-defined categorisation through keywords and does not refer to the semantic content of the advertisements. The registry is supposed to function in a fashion similar to white pages or yellow pages, where businesses can be looked up by name or by a standard service taxonomy as is already used within the industry. UDDI attempts to cover all kinds of services offered by businesses, including those that are offered by phone or e-mail and similar means; in principle, DAML-S could do this, but it has not been our focus.

Technically speaking, each business description in UDDI consists of a businessEntity element, akin to a White Pages element describing the contact information for a business. A businessEntity describes a business by name, a key value, categorisation, services offered (businessService elements) and contact information for the business. A businessService element describes a service using a name, key value, categorisation and multiple bindingTemplate elements. This can be considered to be analogous to a Yellow Pages element that categorises a business. A bindingTemplate element in turn describes the kind of access the service requires (phone, mailto, http, ftp, fax etc.), key values and tModelInstances. tModelInstances are used to describe the protocols, interchange formats that the service comprehends, that is, the technical information required to access the service. It is also used to describe the "namespaces" for the classifications used in categorisation. Many of the elements are optional, including most of the ones that would be required for matchmaking or service composition purposes.

UDDI aims to facilitate the discovery of potential business partners and the discovery of services and their groundings that are offered by known business partners. This may or may not be done automatically. When this discovery occurs, programmers affiliated with the business partners program their own systems to interact with the services discovered. This is also the model generally followed by ebXML. DAML-S enables more flexible discovery by allowing searches to take place on almost any attribute of the ServiceProfile. UDDI, in contrast, allows technical searches only on tModelKeys, references to tModelInstances, which represent full specifications of a kind of service.

UDDI, of itself, does not support semantic descriptions of services. Thus, depending on the functionality offered by the content language, although agents can search the UDDI registry and retrieve service descriptions, a human needs to be involved in the loop to make sense of the descriptions, and to program the access interface.

UDDI does not provide or specify content languages for advertisement so far. Although WSDL is most closely associated with UDDI as a content language, the specification refers to ebXML and XML/edi also as potential candidates. Content languages could be a possible bridge between UDDI and DAML-S. DAML-S is also a suitable candidate for a content language and in this sense, DAML-S and UDDI are complementary. A higher-level service or standard defined on top of UDDI could take advantage of the additional richness of content DAML-S has to offer within the UDDI registries.

1.2 WSDL

WSDL (Web Services Description Language) is an XML format, closely associated with UDDI as the language for describing interfaces to business services registered with a UDDI database. It is thus closer to DAML-S in terms of functionality than UDDI. Like DAML-S, it attempts to separate services, defined in abstract terms, from the concrete data formats and protocols used for implementation, and defines bindings between the abstract description and its specific realization [2]. However, the abstraction of services is at a lower level than in DAML-S.

Services are defined as endpoints, which are essentially sets of ports, that is network addresses associated with certain protocols and data format specifications. The abstract nature of a service arises from the abstract nature of the messages and operations mapped to a port and define its port type. Port types are reusable and can be bound to multiple ports [3]. Operations are of four basic kinds in WSDL: a one-way, a (two-way) request-response, a (two-way) solicit-response and a (one-way) notification message. A message itself is defined abstractly as a request, a response or even a parameter of a request or response and its type, as defined in a type system like XSD. They can be broken into parts to define the logical break-down of a message.

Messages and operations are defined abstractly and are thus reusable and extensible and correspond roughly to the DAML-S ServiceProfile. The service element itself incorporates both a ServiceProfile and ServiceGrounding information. WSDL service descriptions are not as expressive as DAML-S profiles. Preconditions, postconditions and effects of service access cannot be expressed within WSDL.

Like UDDI, WSDL does not support semantic description of services. WSDL focuses on the grounding of services and although it has a concept of input and output types as defined by XSD, it does not support the definition of logical constraints between its input and output parameters. Thus its support for discovery and invocation of services is less versatile than that of DAML-S.

1.3 E-speak

Hewlett-Packard is collaborating with the UDDI consortium to bring E-speak technology to the UDDI standard. It is thus fairly likely that E-speak will be subsumed by UDDI. E-speak and UDDI have similar goals in that they both facilitate the advertisement and discovery of services. E-speak is also comparable to WSDL in that it supports the description of service and data types [4]. It has a matching service to compare service request and service descriptions, which it does primarily on the basis of input-output and service types matching.

E-speak describes services (Resources in the e-speak world) as a set of attributes within several Vocabularies. Vocabularies are sets of attributes common to a logical group of services. E-speak matches lookup requests against service descriptions with respect to these attributes. Attributes take common value types such as String, Int, Boolean and Double. There is a base vocabulary which defines basic attributes such as Name, Type (of value String only), Description, Keywords and Version. Currently, there is no semantic meaning attached to any of the attributes. Any matching which takes place is done over the service description attributes which does not distinguish between any further subtypes. DAML-S had a much richer set of attributes, in DAML-S terminology, the input/output parameters, effects and additional functional attributes. In addition, dependencies between attributes and logical constraints on them are not expressible within E-speak.

Unlike UDDI, which was intended to be an open standard from the beginning, e-speak scores relatively low on interoperability. It requires that an e-speak engine be run on all participating client machines. Furthermore, although e-speak is designed to be a full platform for Web services and could potentially expose a execution monitoring interface, service processes remain a black-box for the e-speak platform and consequently no execution monitoring can be done.

1.4 ebXML

ebXML, being developed primarily by OASIS and the United Nations, approaches the problem from a workflow perspective. ebXML uses two views to describe business interactions, a Business Operational View (BOV) and a Functional Service View (FSV) [5] [6]. The BOV deals with the semantics of business data transactions, which include operational conventions, agreements, mutual obligations and the like between businesses. The FSV deals with the supporting services: their capabilities, interfaces and protocols. Although ebXML does not concentrate on only Web services, the focus of this view is essentially the same as that of the current DAML-S effort.

It has the concept of a Collaboration Protocol Profile (CPP) "which allows a Trading Partner to express their supported Business Processes and Business Service Interface requirements [such that they are understood] by other ebXML compliant Trading Partners", in effect a specification of the services offered by the Trading Partner. A Business Process is a set of business document exchanges between the Trading Partners. CPPs contain industry classification, contact information, supported Business Processes, interface requirements etc. They are registered within an ebXML registry, in which there is discovery of other Trading Partners and the Business Processes they support. In this respect, UDDI has some similarities with ebXML. However, ebXML's scope does not extend to the manner in which the business documents are specified. This is left to the

Trading Partners to agree upon a priori by the creation of a Collaboration Protocol Agreement.

In conclusion, the kind of functionality, interoperability and dynamic matchmaking capabilities provided by DAML-S is only partially supported, as the standards are currently positioned, by WSDL and UDDI. UDDI may become more sophisticated as it incorporates e-speak-like functionalities, but it will not allow automatic service interoperability until it incorporates the information provided by DAML-S.

References

[1] The UDDI Technical White Paper

`http://www.uddi.org/pubs/Iru_UDDI_Technical_White_Paper.pdf`

[2] Web Services Description Language (WSDL) 1.1

`http://www.w3.org/TR/wsdl`

[3] Uche Ogbuji, Using WSDL in SOAP applications: An introduction to WSDL for SOAP programmers:

`http://www-106.ibm.com/developerworks/library/ws-soap/?dwzone=ws`

[4] E-Speak Architectural Specification Release A.0

`http://www.e-speak.hp.com/media/a0/architecturea0.pdf`

[5] The ebXML website.

`http://www.ebxml.org/`

[6] David Webber and Anthony Dutton, Understanding ebXML, UDDI and XML/edi.

`http://www.xml.org/feature_articles/2000_1107_miller.shtml`