Introduction to DAML-S Groundings

June 2, 2002

This white paper accompanies DAML-S draft release 0.6, and explains how DAML-S atomic processes may be "grounded" in WSDL declarations. This material is meant to be read in connection with the DAML-S 0.6 Technical Overview, and assumes prior knowledge of the concepts introduced there, particularly in the description of atomic processes, and the high-level characterization of groundings. In future releases of DAML-S, this material will be incorporated within the Technical Overview.

1 Grounding a Service to a Concrete Realization

The grounding of a service specifies the details of how to access the service – details having mainly to do with protocol and message formats, serialization, transport, and addressing. A grounding can be thought of as a *mapping* from an *abstract* to a *concrete* specification of those service description elements that are required for interacting with the service; in particular, for our purposes, the inputs and outputs of atomic processes. Note that in DAML-S, both the *ServiceProfile* and the *ServiceModel* are thought of as abstract representations; only the *ServiceGrounding* deals with the concrete level of specification.

DAML-S does not include an *abstract* construct for describing messages. Rather, the abstract content of a message is specified, implicitly, by the input or output properties of some atomic process. Thus, atomic processes, in addition to specifying the basic actions from which larger processes are composed, can also be thought of as the communication primitives of an (abstract) process specification.

Concrete messages, however, *are* specified explicitly in a grounding. The central function of a DAML-S grounding is to show how the (abstract) inputs and outputs of an atomic process are to be realized concretely as messages, which carry those inputs and outputs in some specific transmittable format. Due to the existence of a significant body of work in the area of concrete message specification, which is already well along in terms of industry adoption, we have chosen to make use of the Web Services Description Language (WSDL), a particular specification language proposal with strong industry backing, and which we view as represenative of such efforts, in crafting an initial grounding mechanism for DAML-S.

Web Services Description Language (WSDL) "is an XML format for describing network services as a set of endpoints operating on messages containing either document-oriented or procedure-oriented information. The operations and messages are described abstractly, and then bound to a concrete network protocol and message format to define an endpoint. Related concrete endpoints are combined into abstract endpoints (services). WSDL is extensible to allow description of endpoints and their messages regardless of what message formats or network protocols are used to communicate" [1].

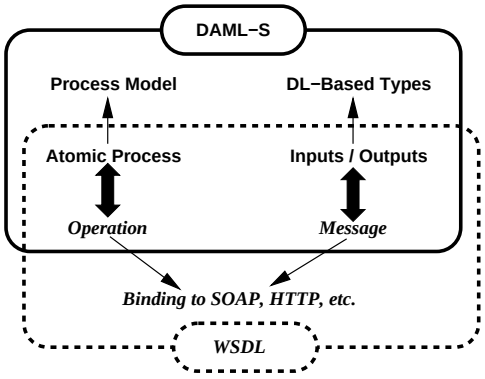

Figure 1: Mapping between DAML-S and WSDL

It may readily be observed that DAML-S' concept of grounding is generally consistent with WSDL's concept of *binding*. Indeed, by using the extensibility elements already provided by WSDL, along with one new extensibility element proposed here, it is an easy matter to ground a DAML-S atomic process. In this section, we show how this may be done, relying on the WSDL 1.1 specification.

1.1 Relationships Between DAML-S and WSDL

The approach described here allows a service developer, who is going to provide service descriptions for use by potential clients, to take advantage of the complementary strengths of these two specification languages. On the one hand (the abstract side of a service specification), the developer benefits by making use of DAML-S' process model, and the expressiveness of DAML+OIL's class typing mechanisms, relative to what XML Schema provides. On the other hand (the concrete side), the developer benefits from the opportunity to reuse the extensive work done in WSDL (and related languages such as SOAP), and software support for message exchanges based on these declarations, as defined to date for various protocols and transport mechanisms.

We emphasize that a DAML-S/WSDL grounding involves a *complementary* use of the two languages, in a way that is in accord with the intentions of the authors of WSDL. Both languages are required for the full specification of a grounding. This is because the two languages do not cover the same conceptual space. As indicated by figure 1, the two languages *do* overlap in the area of providing for the specification of what WSDL calls "abstract types", which in turn are used to characterize the inputs and outputs of services. WSDL, by default, specifies abstract types using XML Schema, whereas DAML-S allows for the definition of abstract types as (description logic-based) DAML+OIL classes [1]. However, WSDL/XSD is unable to express the semantics of a DAML+OIL class. Similarly, DAML-S has no means, as currently defined, to express the binding information that WSDL captures. Thus, it is natural that a DAML-S/WSDL grounding uses DAML+OIL classes as the abstract types of message parts declared in WSDL, and then relies on WSDL binding constructs to specify the formatting of the messages. [2]

A DAML-S/WSDL grounding is based upon the following three correspondences between DAML-S and WSDL. Figure 1 shows the first two of these.

[1] The primitive types of XML Schema can also be used in defining DAML+OIL properties.

[2] The DAML+OIL classes can either be defined within the WSDL *types* section, or defined in a separate document and referred to from within the WSDL description. In the remainder of this exposition, we describe only the latter approach.

1. A DAML-S atomic process corresponds to a WSDL *operation*. Different types of operations are related to DAML-S processes as follows:

   - An atomic process with both inputs and outputs corresponds to a WSDL *request-response* operation.
   - An atomic process with inputs, but no outputs, corresponds to a WSDL *one-way* operation.
   - An atomic process with outputs, but no inputs, corresponds to a WSDL *notification* operation.
   - A composite process with both outputs and inputs, and with the sending of outputs specified as coming before the reception of inputs, corresponds to WSDL's *solicit-response* operation.[3]

2. The set of inputs and the set of outputs of a DAML-S atomic process each corresponds to WSDL's concept of *message*. More precisely, DAML-S inputs correspond to the parts of an input message of a WSDL operation, and DAML-S outputs correspond to the parts of an output message of a WSDL operation.

   Note that WSDL allows (at most) one input, and (at most) one output message to be associated with an operation. This is in accord with a decision made independently, in DAML-S, that a grounding must map all inputs to (at most) a single message, and similarly for outputs.

3. The types (DAML+OIL classes) of the inputs and outputs of a DAML-S atomic process correspond to WSDL's extensible notion of *abstract type* (and, as such, may be used in WSDL specifications of message parts).

The job of a DAML-S/WSDL grounding is first, to define, in WSDL, the messages and operations by which an atomic process may be accessed, and then, to specify correspondences (1) and (2). Although it is not logically necessary to do so, we believe it will be useful to specify these correspondences both in WSDL and in DAML-S. Thus, as indicated in the following, we allow for constructs in both languages for this purpose.

1.2 Grounding DAML-S Services With WSDL and SOAP

Because DAML-S is an XML-based language, and its atomic process declarations and input/output types already fit nicely with WSDL, it is easy to extend existing WSDL bindings for use with DAML-S, such as the SOAP binding. In this subsection, we indicate briefly how an arbitrary atomic process, specified in DAML-S, can be given a grounding using WSDL and SOAP, with the assumption of HTTP as the chosen transport mechanism.

Grounding DAML-S with WSDL and SOAP involves the construction of a WSDL service description with all the usual parts (*message, operation, port type, binding*, and *service* constructs), except that the *types* element can normally be omitted. DAML-S extensions are introduced as follows:

[3]Since a composite process has no grounding, this construct would be grounded indirectly by means of its relationship to a simple process (by the *collapse* property), and hence to an atomic process (by the *realizedBy* property), as described in the Technical Overview. We are considering whether to create a new kind of atomic process in DAML-S, which corresponds directly to the solicit-response operation.

1. In each *part* of the WSDL *message* definition, the *daml-property* attribute is used to indicate the fully-qualified name of the DAML-S input or output property, to which this part of the message corresponds. From the property name, the appropriate DAML range class – the class of object which this message part will contain – can easily be obtained.

2. In each WSDL *operation* element, the *daml-s-process* attribute is used to indicate the name of the DAML-S atomic process, to which the operation corresponds.

3. Within the WSDL *binding* element, the *encodingStyle* attribute is given a value such as "http://www.daml.org/2001/03/daml+oil.daml", to indicate that the message parts will be serialized in the normal way for class instances of the given types, for the specified version of DAML.

Note that WSDL already allows for the use of arbitrary new attributes in message part elements, and for the use of arbitrary values for the *encodingStyle* attribute. Thus, extension (2) above is the only point on which a modification to the current WSDL specification is called for.

1.3 The *Grounding* Class

We have shown how WSDL may be used to ground a DAML-S atomic process; in particular, how WSDL may be used to specify the correspondence between a DAML-S atomic process and a WSDL operation, as well as the correspondences between the process' inputs (or outputs) and a WSDL message. Thus far, however, we have only shown how WSDL definitions may refer to the corresponding DAML-S declarations. It remains to establish a mechanism by which the relevant WSDL constructs may be referenced in DAML-S. The DAML-S *WsdlGrounding* class, a subclass of *Grounding*, serves this purpose.

A WsdlGrounding object refers to specific elements within the WSDL specification, using the following properties:

- *wsdlReference*: A URI that indicates the version of WSDL in use.
- *otherReferences*: A list of URIs indicating other relevant standards employed by the WSDL code (e.g., SOAP, HTTP, MIME).
- *wsdlDocuments*: A list of the URIs of the WSDL document(s) that give the grounding.
- *wsdlOperation*: The URI of the WSDL operation corresponding to the given atomic process.
- *wsdlInputMessage*: An object containing the URI of the WSDL message definition that carries the inputs of the given atomic process, and a list of mapping pairs, which indicate the correspondence between particular DAML-S input properties and particular WSDL message parts.
- *wsdlOutputMessage*: Similar to *wsdlInputMessage*, but for outputs.

References

[1] E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana. Web Services Description Language (WSDL) 1.1. http://www.w3.org/TR/2001/NOTE-wsdl-20010315, 2001.