Semantic Web Services

Light-weight Annotations

Where are we?

<table>
<thead>
<tr>
<th>#</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
</tr>
<tr>
<td>2</td>
<td>Web Science</td>
</tr>
<tr>
<td>3</td>
<td>Service Science</td>
</tr>
<tr>
<td>4</td>
<td>Web services</td>
</tr>
<tr>
<td>5</td>
<td>Web2.0 services</td>
</tr>
<tr>
<td>6</td>
<td>Semantic Web</td>
</tr>
<tr>
<td>7</td>
<td>Web Service Modeling Ontology (WSMO)</td>
</tr>
<tr>
<td>8</td>
<td>Web Service Modeling Language (WSML)</td>
</tr>
<tr>
<td>9</td>
<td>Web Service Execution Environment (WSMX)</td>
</tr>
<tr>
<td>10</td>
<td>OWL-S and other</td>
</tr>
<tr>
<td>11</td>
<td>Light-weight Annotations</td>
</tr>
<tr>
<td>12</td>
<td>Applications</td>
</tr>
<tr>
<td>13</td>
<td>Mobile Services</td>
</tr>
</tbody>
</table>

Outline

- Motivation
- Technical solution
  - Overview of light-weight SWS annotations
  - WSDL
  - hRESTS
  - SAWSDL
  - MicroWSMO
  - WSMO-Lite
- Illustration by a larger example
- Extensions
- Summary
- References

Motivation
Motivation

• The vision of Semantic Web Services (SWS)
  – Automating typical Web Service usage tasks.
  – Resolving heterogeneities issues.
  – Fostering scalability.

• Existing SWS approaches
  – WSMO, OWL-S, SWSF
  – Addressing the aforementioned problems.
  , but they are considered as
    – Heavyweight solutions.
    – Introducing new languages founded on a expressive formalisms.
    – Promoting the top-down modeling approach (semantics-first).
    – Grounded usually in WSDL-based services.

Motivation

• Need for lightweight service ontologies.

• Directly built on top of the newest W3C standards
  – RDF(S), OWL, SAWSDL

• Promoting the bottom-up modeling approach
  – Augmentation of existing service specifications with semantic descriptions.

• Covering the other grounding approaches (i.e., REST)
  – WSDL-based services
    • 23757 services and 8094 providers according to Service Finder
      • Many of them are used for the intra enterprise integration
    – RESTful-based services
      • 68% RESTful services vs. 19% SOAP services

Technical Solution

Overview of the light-weight SWS annotations

1 Statistics retrieved from the Service Finder demo on Dec 17, 2008 @ http://demo.service-finder.eu/statistics
2 Statistics retrieved from the Programmable Web on Dec 17, 2009 @ http://www.programmableweb.com/apis

Motivation

1. SAWSDL
  – New W3C standard.
  – Building on top of WSDL, URIs.
  – Assuming RDF, OWL.

2. Lightweight SWS approaches
  – WSMO modularized.
  – Key pieces of service semantics.
  – Simple semantic representation.
Technical Solution

Overview

• WSMO-Lite
  - Lightweight semantic descriptions for services on the Web
  - Inspired by WSMO (Lecture #7)
  - Focusing on a subset of WSMO

• From WSDL over SAWSDL to WSMO-Lite
  - Web Service Description Language (WSDL) is the standard to describe Web service interfaces (Lecture #4)
  - Semantically Annotated WSDL (SAWSDL) introduces a set of extension attributes which can be associated to the elements of a WSDL definition
  - Attributes can point to the WSMO-Lite elements

• From hRESTS over MicroWSMO to WSMO-Lite
  - HTML for RESTful Service Descriptions (hRESTS) is a microformat used to identify basic service constructs (service definition, operations, inputs, outputs) inside HTML page which describes RESTful service.
  - MicroWSMO extends hRESTS with the elements (model, lowering, lifting) which can be associated with the elements of a hRESTS definition.
  - Elements can point to the WSMO-Lite elements
  - MicroWSMO and hRESTS are lightweight equivalents for SAWSDL and WSDL for RESTful services.
Web Service Description Language

Overview

- Web Service Description Language
  - Interface Definition Language (IDL) for Web Services
  - Current version: 2.0 @ W3C
    - Version 1.1 still in widespread use

- Interface – reusable, abstract
  - Operations with Message Exchange Patterns (MEPs)
    - In-Out, In-Only, Out-Only, Out-In

- Binding – reusable, concrete
  - Service implements an interface
    - Endpoints use bindings

WSDL Parts

- Types
  - Defining types and structures used during message exchanges

- Interface
  - Abstract view on what the operation does
  - Contains descriptions of operations and faults
  - References to the type definitions and element declarations

- Binding
  - Defines how an interface is accessed over a network
  - Specifies network details for each operation and fault

- Service
  - Main WSDL component
  - Can have a number of network accessible addresses – endpoints
  - Service implements interface

Example

```xml
  <wsdl:types>
    <xs:schema targetNamespace="http://www.w3.org/2002/ws/sawsdl/spec/wsdl/order#" elementFormDefault="qualified">
      <xs:element name="OrderRequest">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="customerNo" type="xs:integer"/>
            <xs:element name="orderItem" type="item" minOccurs="1" maxOccurs="unbounded"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      ...
      <xs:element name="OrderResponse" type="confirmation"/>
    </xs:schema>
  </wsdl:types>
  <wsdl:interface name="Order">
    <wsdl:operation name="order" pattern="http://www.w3.org/ns/wsdl/in-out">
      <wsdl:input element="OrderRequest"/>
      <wsdl:output element="OrderResponse"/>
    </wsdl:operation>
  </wsdl:interface>
  ...
</wsdl:description>
```

Technical Solution

HTML for RESTful Service Descriptions (hRESTS)
A RESTful Web service is:
- A set of Web resources
- Interacting with GET/POST/PUT/DELETE
- Interlinked
- Data-centric, not functionality-centric
- Machine-oriented

More details can be found in Lecture #5

A service is made up of interlinked resources.

Links are not only `<a href="...">`, but also forms, such as the reservation form that leads from search results to the payment resource.

nouns vs. verbs
**hRESTS Overview**

- "There's usually an HTML page"
  - There's no WSDL for Web apps
  - APIs described mostly in text
- Identifying machine-readable parts
  - Service, its operations
  - Resource address, HTTP method
  - Input/output data format
- **hRESTS** microformat

**hRESTS Definition**

- **HTML** for RESTful Service Description
- Introduces the service model structure
  - service(* label)
  - operations(* address, method)
  - input, output
- Basis for extensions:
  - MicroWSMO adds semantic annotations

**Example Description**

Description of the ACME Hotels service:

The operation `getHotelDetails` is invoked using the method GET at `http://example.com/h/{id}`, with the ID of the particular hotel replacing the parameter `id`. It returns the hotel details in an `ex:hotelInformation` document.

**Example HTML**

```html
<p>Description of the ACME Hotels service:</p>
The operation `<code>getHotelDetails</code>` is invoked using the method GET at `<code>http://example.com/h/{id}</code>`, with the ID of the particular hotel replacing the parameter `<code>id</code>`. It returns the hotel details in an `<code>ex:hotelInformation</code>` document.
```
### hRESTS

Example hRESTS

```html
<example>
    <div class="service" id="svc">
        <p>Description of the ACME Hotels service:</p>
        <div class="operation" id="op1">
            <p>The operation <code>getHotelDetails</code> is invoked using the method <span class="method">GET</span> at <code>http://example.com/h/{id}</code>, with <span class="input">the ID of the particular hotel replacing the parameter <code>id</code></span>. It returns <span class="output">the hotel details in an <code>ex:hotelInformation</code> document</span>.
        </p>
    </div>
</example>
```

### Technical Solution

**Semantically Annotated WSDL (SAWSDL)**

#### SAWSDL Overview

- How to add semantic annotations to various parts of a WSDL document:
  - Input and output message structures, interfaces and operations.
- Relying to simple extension attributes:
  - Compliant to WSDL v2.0 and v1.1, and XML Schema
- Annotations can be used for various purposes:
  - WSDL interfaces and operations with categorization information used to advertise Web services
  - XML Schema types to foster discovery and composition
  - Specifying the data mapping of XML Schema to/from an ontology used during invocation (possible mediation)
- Independent on the ontology expression language and mapping languages

#### SAWSDL Extension attributes

- Extension attributes of SAWSDL are:
  - `modelReference`:
    - Pointers to a concept in some semantic model.
    - Annotates XML Schema type definitions, element declarations, and attribute declarations, WSDL interfaces, operations, and faults.
  - `liftingSchemaMapping`:
    - Added to XML Schema element declarations and type definitions for specifying lifting mappings between semantic data and XML.
  - `loweringSchemaMapping`:
    - Added to XML Schema element declarations and type definitions for specifying lowering mappings between semantic data and XML.
SAWSDL
Extension attributes

- Multiple semantic annotations can be associated with WSDL element.
- Schema mappings and a model references can contain multiple pointers.
- Multiple schema mappings are interpreted as alternatives.
- Multiple model references all apply.
- SAWSDL does not specify any other relationship between them.

SAWSDL
Lifting/Lowering

Semantic level:

Syntactic level:

Lifting

SOAP communication

Web service

SAWSDL
An example

Technical Solution

MicroWSMO
MicroWSMO

Introduction

• hRESTS effectively creates an anologue of WSDL for RESTful services.
• hRESTS forms the basis for further extensions to annotate service descriptions for further processing.
• MicroWSMO, an extension of hRESTS adds semantics annotations through adopting the SAWSDL extensions:
  – model indicates that a link is a model reference, and
  – lifting, lowering links to the respective data transformations.
• The model attribute can point to the WSMO-Lite descriptions.

MicroWSMO

Example

<div class="service" id="svc">
  <p><span class="label">ACME Hotels</span> is a <a rel="model" href="&ex;AccommodationReservationService">hotel reservation</a> service.</p> …
  <div class="operation" id="op1">
    …
    <span class="input">A particular hotel ID replaces the param</span> <a rel="lowering" href=".../hotelID.xsparql">lowering</a>.
    …
    <a rel="lowering" href=".../hotelID.xsparql">lowering</a>.
  </div>
</div>

MicroWSMO

Semantics Implied in Web

• Hypermedia → behavioral semantics
  – Links become available through interaction
• Uniform interface → behavioral semantics
  – GET, PUT, DELETE have known effects
  – GET is safe, PUT and DELETE idempotent
  – POST has no implied semantics
• Self-description → information model
  – Operation output data can specify what it is
  • GRDDL, other semantic annotations

GRDDL is a mechanism for Gleaning Resource Descriptions from Dialects of Languages. This GRDDL specification introduces markup based on existing standards for declaring that an XML document includes data compatible with the Resource Description Framework (RDF) and for linking to algorithms (typically represented in XSLT), for extracting the data from the document.

Technical Solution

WSMO-Lite
WSMO-Lite

Introduction

- Lightweight semantic descriptions for services on the Web
  - Filling the SAWSDL annotations with concrete semantic service descriptions.
- Working draft of CMS Working Group, v0.3
- Inspired by WSMO ontology
  - Focusing on a subset.
  - Defining gradual extension of SAWSDL.
- Addresses following requirements:
  - Identifies types and simple vocabulary for semantic description of services and languages used to express descriptions.
  - Defines an annotation mechanism for WSDL.
  - Provides a bridge between WSDL, SAWSDL and existing domain-specific ontologies such as classification schemas, etc.

Types of Service Semantics

- Functional (F)
  - What the service does
- Behavioral (B)
  - How the client talks to the service
- Information model (I)
  - For handling data
  - Incl. lifting/lowering
- Nonfunctional (N)
  - Policies, QoS, price, location etc.

Semantics in Service Model

- Using SAWSDL model references, the four kinds of semantics are attached to the service model:
  - functional (F) and nonfunctional (N) properties to the service.
  - behavioral (B) properties to the operations, and
  - information model (I) properties to the input and output messages (and the fault messages as well).

Functional Semantics

- For service discovery, composition
- Category
  - Functionality categorization
  - E.g. eCl@ss, UDDI
  - Or tagging, folksonomies
- Capability
  - Precondition, Effect
  - Using WSML rule languages
WSMO-Lite
Category Example

wl:FunctionalClassificationRoot

ex:eCommerceService

subclasses

ex:TravelReservationService

ex:AccommodationReservationService

ex:TravelReservationService

rdfs:subClassOf ex:eCommerceService.

ex:AccommodationReservationService

rdfs:subClassOf ex:eCommerceService.

...

WSMO-Lite
Category Example (RDF)

ex:eCommerceService

rdf:type wl:FunctionalClassificationRoot.

ex:TravelReservationService

rdfs:subClassOf ex:eCommerceService.

ex:AccommodationReservationService

rdfs:subClassOf ex:eCommerceService.

...

WSMO-Lite
Capability Example

ex:RomaHotelReservationPrecondition

rdf:type wl:Condition;

rdf:value "[

numberOfGuests hasValue ?guests

and

city hasValue ?city

memberOf ReservationData

and

?guests <= 10

and

?city = 'Roma'

]"^^wsml:AxiomLiteral.

WSMO-Lite
Nonfunctional Semantics

- For ranking and selection
- Not constrained, any ontologies
- Example:

ex:PriceSpecification

rdfs:subClassOf wl:NonFunctionalParameter.

ex:ReservationFee

rdf:type ex:PriceSpecification;

rdf:value "15"^^ex:euroAmount.
WSMO-Lite
Behavioral Semantics

• For invocation, composition, process mediation
• Functionalities on operations
  – Capabilities, categories
  – Client selects operation to invoke next
    – Instead of being strictly guided by an explicit process
• Example functional category for operations:
  WebArch interaction safety

WSMO-Lite
Information Semantics

• For invocation, composition, data mediation
• Not constrained, any ontologies
• Refer to course Semantic Web

WSMO-Lite service ontology in RDFS, serialized in Notation 3.

@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix owl: <http://www.w3.org/2002/07/owl#> .
@prefix wsl: <http://www.wsmo.org/ns/wsmo-lite#> .
wsl:Ontology a rdfs:Class;
rdfs:subClassOf owl:Ontology.
wsl:FunctionalClassificationRoot rdfs:subClassOf rdfs:Class.
wsl:NonFunctionalParameter a rdfs:Class.
wsl:Condition a rdfs:Class.
wsl:Effect a rdfs:Class.

WSMO-Lite service ontology in RDFS - Explanation

• Semantics of the WSMO-Lite elements:
  – wsl:Ontology
    – Defines a container for a collection of assertions about the information model of a service.
  – wsl:FunctionalClassificationRoot
    – A classification (taxonomy) of service functionalities can be used for functional description of a service.
  – wsl:NonFunctionalParameter
    – specifies a place holder for a concrete domain-specific nonfunctional property.
  – wsl:Condition and wsl:Effect
    – together form a capability in a functional service description. Both are expected to use a concrete logical language to describe the logical expressions for conditions and effects.
• WSDL has more detail than our simplified service model, therefore there are more ways of attaching semantics.

• Information semantics can be anywhere in the schema inside the types section.

• Functional semantics can be either on the interface (general, reusable) or on the service (specific), and

• Nonfunctional parameters can be on a binding (general, reusable) or on the service (specific).
A telecommunication company wants to enter a highly competitive video on demand market.

The market is formed around the service provisioning platform based on WSMO-Lite service description.

The company has a WSDL service offering video on demand subscription functionality:
- The service formalizes basic concepts in the domain such as Customer, Service, Network Connection, etc. in the form of XML Schema.
- The service provides basic subscription operation.

The service can process a subscription request only if a customer already has a network connection with the appropriate bandwidth.

The company also wants to associate its pricing schema related to the service consumption.

### WSMO-Lite Description

```xml
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix wsl: <http://www.wsmo.org/ns/wsmo-lite#> .
@prefix ex: <http://example.org/onto#> .
@prefix xs: <http://www.w3.org/2001/XMLSchema#> .
@prefix wsml: <http://www.wsmo.org/wsml/wsml-syntax#> .

<> a wsl:Ontology .
ex:Customer a rdfs:Class .
ex:hasService a rdf:Property ;
rdfs:domain ex:Customer ;
rdfs:range ex:Service .
ex:Service a rdfs:Class .
ex:hasConnection a rdf:Property ;
rdfs:domain ex:Customer ;
rdfs:range ex:NetworkConnection .
ex:NetworkConnection a rdfs:Class .
ex:providesBandwidth a rdf:Property ;
rdfs:domain ex:NetworkConnection ;
rdfs:range xs:integer .
ex:VideoOnDemandService rdfs:subClassOf ex:Service .
ex:VideoOnDemandSubscriptionPrecondition a wsl:Condition ;
ex:VideoOnDemandSubscriptionEffect a wsl:Effect ;
ex:VideoOnDemandPrice a ex:PriceSpecification ;
ex:pricePerChange "30" ex:euroAmount ;
ex:installationPrice "49" ex:euroAmount .
ex:SubscriptionService a wsl:FunctionalClassificationRoot .
ex:VideoSubscriptionService rdfs:subClassOf ex:SubscriptionService .
ex:NewsSubscriptionService rdfs:subClassOf ex:SubscriptionService .
```

The description defines a simple ontology for a telecommunication service:

- a capability for a concrete Video on Demand subscription service (the condition says that the customer must have a network connection with some minimal bandwidth, the effect says that the customer is subscribed to the service);
- a nonfunctional property describing the pricing;
- a simple functionality classification with three categories;
- defines also the wsml:AxiomLiteral data type for WSML-Flight logical expressions so that a client can correctly process them according to the WSML specification.

SAWSDL Annotations - Fragments

Reference annotation linking to the class from the service information model ontology and transformation annotation referencing mapping used to lower semantic descriptions to an appropriate communication format.

Functional annotation including references to the capabilities and categories applied to the interface definition.

Non functional model reference annotation associated with the concrete service.

Summary
Summary

- Existing SWS approaches are
  - Heavyweight solutions.
  - Introducing new languages founded on expressive formalisms.
  - Promoting the top-down modeling approach (semantics-first).
  - Grounded usually in WSDL-based services.

- Light-weight annotations for SWS include
  - WSDL
    - Common and standard way to describe Web Service interface.
  - hRESTS
    - Microformat used to identify basic service constructs (service definition, operations, inputs, outputs) inside HTML page which describes RESTful service.
  - SAWSDL
    - Set of extension attributes which can be associated to the elements of a WSDL definition.
  - MicroWSMO
    - Extends hRESTS with the elements (model, lowering, lifting) which can be associated with the elements of a hRESTS definition.
  - WSMO-Lite
    - Lightweight ontology for semantic descriptions of services on the Web.

Extensions

- Implementation
  - WSMO-Lite and MicroWSMO automation
    - SOA4All Studio
    - WSMX
  - Tool support for service annotation
    - SOA4All Studio and others
  - Standardization of WSMO-Lite through CMS Group

References

- Mandatory reading:
  - SAWSDL: http://w3.org/TR/sawsdl
  - WSMO-Lite: http://w3.org/TR/WSMO-Lite
  - hRESTS & MicroWSMO: http://w3.org/TR/REST

- Further reading:
  - RDF:
    - http://w3.org/TR/PR-rdf
  - RDFS:
    - http://w3.org/TR/PR-rdfs
  - Wikipedia links:

Next Lecture

<table>
<thead>
<tr>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
</tr>
<tr>
<td>2</td>
<td>Web Science</td>
</tr>
<tr>
<td>3</td>
<td>Service Science</td>
</tr>
<tr>
<td>4</td>
<td>Web services</td>
</tr>
<tr>
<td>5</td>
<td>Web 2.0 services</td>
</tr>
<tr>
<td>6</td>
<td>Semantic Web</td>
</tr>
<tr>
<td>7</td>
<td>Web Service Modeling Ontology (WSMO)</td>
</tr>
<tr>
<td>8</td>
<td>Web Service Modeling Language (WSML)</td>
</tr>
<tr>
<td>9</td>
<td>Web Service Execution Environment (WSMX)</td>
</tr>
<tr>
<td>10</td>
<td>OWL-S and other</td>
</tr>
<tr>
<td>11</td>
<td>Light-weight Annotations</td>
</tr>
<tr>
<td>12</td>
<td>Applications</td>
</tr>
<tr>
<td>13</td>
<td>Mobile Services</td>
</tr>
</tbody>
</table>
Questions?