Semantic Web

Semantic Web Services
## 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>Semantic Web Architecture</td>
</tr>
<tr>
<td>3</td>
<td>Resource Description Framework (RDF)</td>
</tr>
<tr>
<td>4</td>
<td>Web of data</td>
</tr>
<tr>
<td>5</td>
<td>Generating Semantic Annotations</td>
</tr>
<tr>
<td>6</td>
<td>Storage and Querying</td>
</tr>
<tr>
<td>7</td>
<td>Web Ontology Language (OWL)</td>
</tr>
<tr>
<td>8</td>
<td>Rule Interchange Format (RIF)</td>
</tr>
<tr>
<td>9</td>
<td>Reasoning on the Web</td>
</tr>
<tr>
<td>10</td>
<td>Ontologies</td>
</tr>
<tr>
<td>11</td>
<td>Social Semantic Web</td>
</tr>
<tr>
<td>12</td>
<td><strong>Semantic Web Services</strong></td>
</tr>
<tr>
<td>13</td>
<td>Tools</td>
</tr>
<tr>
<td>14</td>
<td>Applications</td>
</tr>
</tbody>
</table>
1. Motivation
2. Technical solution
   2.1 What is a service?
   2.2 Web services
   2.3 Semantic Web services
   2.4 SWS Frameworks
3. Extensions
4. Summary
5. References
MOTIVATION
Motivation

http://www.sti-innsbruck.at/dip-movie
Motivation

• The Web is moving from static data to dynamic functionality
  – Web services: a piece of software available over the Internet, using standardized XML messaging systems
  – Mashups: The compounding of two or more pieces of web functionality to create powerful web applications
Motivation
Limitations of the current Web Processes

• Web services and mashups are limited by their syntactic nature

• As the amount of services on the Web increases it will be harder to find Web services in order to use them in mashups

• The current amount of human effort required to build applications is not sustainable at a Web scale
What is needed?

- Formal, machine processable descriptions of processes on the Web that allows easy integration, configuration and reuse
- Semantic support for finding, composing and executing these processes and all the other related tasks

**Solution:** Combine Semantics and Web processes/services that enables the automation of many of the currently human intensive tasks around Web processes/services
TECHNICAL SOLUTION
WHAT IS A SERVICE?
The word service is used in several contexts:
- Communication Service
- Ticket Reservation Service
- Transport Service
- Information Service
- Finance Service
- E-government Service
- ...

But what is a Service?
What is a service?

Main Entry: ser·vice
Function: noun
Etymology: Middle English, from Anglo-French servise, from Latin servitium condition of a slave, body of slaves, from servus slave
1 a: the occupation or function of serving <in active service> b: employment as a servant <entered his service>
2 a: the work performed by one that serves <good service> b: help, use, benefit <glad to be of service> c: contribution to the welfare of others d: disposal for use <I'm entirely at your service>
3 a: a form followed in worship or in a religious ceremony <the burial service> b: a meeting for worship —often used in plural <held evening services>
4: the act of serving: as a: a helpful act <did him a service> b: useful labor that does not produce a tangible commodity —usually used in plural <charge for professional services> c: serve
5: a set of articles for a particular use <a silver tea service>
6 a: an administrative division (as of a government or business) <the consular service> b: one of a nation's military forces (as the army or navy)
7 a: a facility supplying some public demand <telephone service> <bus service> b: a facility providing maintenance and repair <television service>
8: the materials (as spun yarn, small lines, or canvas) used for serving a rope
9: the act of bringing a legal writ, process, or summons to notice as prescribed by law
10: the act of a male animal copulating with a female animal
11: a branch of a hospital medical staff devoted to a particular specialty <obstetrical service>

Merriam-Webster Online, http://www.m-w.com
• For different people the term Service has different meaning
• **In Business and Economics**
  – a service is seen as a business activity that often results in intangible outcomes or benefits
  – a service is the non-material equivalent of a good. Service provision has been defined as an economic activity that does not result in ownership, and this is what differentiates it from providing physical goods.
  – a process that creates benefits by facilitating either a change in customers, a change in their physical possessions, or a change in their intangible assets.
What is a service?

- **In Computer Science**
  - the terms service and Web service are often regarded as interchangeable to name a software entity accessible over the Internet.
  - a (Web) service is seen software system designed to support interoperable machine-to-machine interaction over a network.
Service vs. Web Service

- **Service**
  - A provision of value in some domain (not necessarily monetary, independent of how service provider and requestor interact)

- **Web Service**
  - Computational entity accessible over the Internet (using Web Service Standards & Protocols), provides access to (concrete) services for the clients.
Web Service properties

• **Functional**
  – contains the formal specification of **what** exactly the service can do.

• **Behavioral**
  – **how** the functionality of the service can be achieved in terms of interaction with the service and as well in terms of functionality required from the other Web services.

• **Non-functional properties**
  – captures constraints over the previous mentioned properties
Web Services

Dynamic

Web Services
UDDI, WSDL, SOAP

Static

WWW
URI, HTML, HTTP

Semantic Web
RDF, RDF(S), OWL

Bringing the computer back as a device for computation
Web Services: Definition

1) “Loosely coupled, reusable software components that encapsulate discrete functionality and are distributed and programmatically accessible over standard Internet protocols”, *The Stencil Group*

2) Web service applications are encapsulated, loosely coupled Web “components” that can bind dynamically to each other, *F. Curbera*

3) “Web Services are a new breed of application. They are self-contained, self-describing, modular applications that can be published, located, and invoked across the Web. Web Services perform functions, which can be anything from simple request to complicated business processes”, *The IBM Web Services tutorial*

**Common to all definitions:**
- Components providing functionality
- Distributed
- Accessible over the Web
Definitions

Def 1. Software Architecture

Def 2. New concept for eWork and eCommerce

Def 3. New programming technology
Def 1. Software architecture

- Web Services connect computers and devices with each other using the Internet to exchange data and combine data in new ways.

- The key to Web Services is on-the-fly software creation through the use of loosely coupled, reusable software components.

- Software can be delivered and paid for as fluid streams of services as opposed to packaged products.
Def 2. Web Services as a new Concept for eWork and eCommerce

- Business services can be completely decentralized and distributed over the Internet and accessed by a wide variety of communications devices.
- The internet will become a global common platform where organizations and individuals communicate among each other to carry out various commercial activities and to provide value-added services.
- The dynamic enterprise and dynamic value chains become achievable and may be even mandatory for competitive advantage.
Def 3. Web Services as a programming technology
Web Services are Remote Procedure Calls (RPC) over HTTP
Web Services

- UDDI Registry
  - Finds Services

- WSDL
  - Points to Description
  - Describes Service

- Service Consumer
  - Communicate with XML Messages

- Web Service
  - SOAP
WSDL 1.0

- Web Service Description Language

describes interface for consuming a Web Service:
- Interface: operations (in- & output)
- Access (protocol binding)
- Endpoint (location of service)
• Simple Object Access Protocol
• W3C Recommendation

XML data transport:
- sender / receiver
- protocol binding
- communication aspects
- content
• Universal Description, Discovery, and Integration Protocol
• OASIS driven standardization effort

Registry for Web Services:
- provider
- service information
- technical access
Restful services

• Another way of realizing services, other than SOAP/WSDL/UDDI approach
• Follows the Web principles (REST principles)

• Services expose their data and functionality through resources identified by URI
• Services are Web pages that are meant to be consumed by an autonomous program
• Uniform interfaces for interaction: GET, PUT, DELETE, POST
• HTTP as the application protocol instead of SOAP

• Used by Amazon, Google, Flickr, and many others
RESTful WS Definition

• A RESTful Web service is:
  – A set of Web resources
  – Interlinked
  – Data-centric, not functionality-centric
  – Machine-oriented

• Like Web applications, but for machines
• Like WS-*, but with more Web resources
Technologies

• REST: the architectural style of the Web
• HTTP: the basis
• XML, JSON, Microformats for data exchange
• Atom/RSS, AtomPub
  – Feeds, publishing, syndication
• Javascript programming the browser, AJAX
Example: Flickr

- Example operations (methods):
  - `flickr.photos.addTags`
  - `flickr.photos.delete`
  - `flickr.contacts.getList`
  - `flickr.photos.comments.editComment`
  - ...

- HTTP GET or POST
  @ http://api.flickr.com/services/rest/?method=method&parameters

- Special authentication method
Flickr API Authentication

• Application needs an API key
  – API key requested by application developer
  – Application has a shared secret with Flickr
• Every method needs API key
• Application lets user log in, gets auth token
• Authenticated methods need auth token and signature
  – Signature uses shared secret and all parameters

More at http://www.flickr.com/services/api/misc.userauth.html
SEMANTIC WEB SERVICES
Semantic Web Services

Bringing the web to its full potential

Dynamic

Web Services
UDDI, WSDL, SOAP

Static

WWW
URI, HTML, HTTP

Semantic Web
Services
RDF, RDF(S), OWL
Deficiencies of WS Technology

Syntax only!
Deficiencies of WS Technology

- current technologies allow usage of Web Services
- but:
  - only syntactical information descriptions
  - syntactic support for discovery, composition and execution

=> **Web Service usability, usage, and integration needs to be inspected manually**
  - no semantically marked up content / services
  - no support for the Semantic Web

=> current Web Service Technology Stack failed to realize the promise of Web Services
So what is needed?

- **Mechanized support** is needed for
  - Annotating/designing services and the data they use
  - Finding and comparing service providers
  - Negotiating and contracting services
  - Composing, enacting, and monitoring services
  - Dealing with numerous and heterogeneous data formats, protocols and processes, i.e. mediation

=> Conceptual Models, Formal Languages, Execution Environments
Semantic Web Services

Semantic Web Technology

- allow machine supported data interpretation
- ontologies as data model

Web Service Technology

automated discovery, selection, composition, and web-based execution of services

=> Semantic Web Services as integrated solution for realizing the vision of the next generation of the Web
• define exhaustive description frameworks for describing Web Services and related aspects (**Web Service Description Ontologies**)

• support ontologies as underlying data model to allow machine supported data interpretation (**Semantic Web aspect**)

• define semantically driven technologies for automation of the Web Service usage process (**Web Service aspect**)
Tasks to be automated

- Service Description
  - Describe the service explicitly, in a formal way
- Service Publishing
  - Make available the description of the service
- Service Discovery
  - Locate different services suitable for a given goal
- Service Negotiation & Contracting
  - Choose the most appropriate services among the available ones
- Service Composition
  - Combine services to achieve a goal
- Service Enactment & Monitoring
  - Invoke & Monitor services following programmatic conventions
SWS FRAMEWORKS
Some of the most popular approaches for SWS are:

- **WSMO**: Ontologies, Goals, Web Services, Mediators
- **OWL-S**: WS Description Ontology (Profile, Service Model, Grounding)
- **Meteor-S (WSDL-S)**: Bottom-up semantic annotation of WSDL descriptions
- **SWSF**: Process-based Description Model & Language for WS
- **IRS-III**: an implementation of WSMO framework
The WSMO Approach

- Conceptual Model & Axiomatization for SWS
- Formal Language for WSMO
- Execution Environment for WSMO
- Ontology & Rule Language for the Semantic Web

STI2 CMS WG

OASIS

SEE TC
Web Service Modeling Ontology (WSMO)

- Conceptual Model & Axiomatization for SWS
- Formal Language for WSMO
- Execution Environment for WSMO

Ontology & Rule Language for the Semantic Web

STI2 CMS WG

OASIS SEE TC
Objectives that a client wants to achieve by using Web Services

- Provide the formally specified terminology of the information used by all other components

Semantic description of Web Services:
- **Capability** (functional)
- **Interfaces** (usage)

Connectors between components with mediation facilities for handling heterogeneities
In WSMO, Ontologies are the key to linking conceptual real-world semantics defined and agreed upon by communities of users.

Class ontology sub-Class wsmoElement
importsOntology type ontology
usesMediator type ooMediator
hasConcept type concept
hasRelation type relation
hasFunction type function
hasInstance type instance
hasRelationInstance type relationInstance
hasAxiom type axiom

Examples:
- The Location Ontology (http://www.wsmo.org/ontologies/location) contains the concepts “Country” and “Address”
- The Location Ontology (http://www.wsmo.org/ontologies/location) contains the “Austria” and “Germany” instances
WSMO – the Web Service Element

- WSMO Web service descriptions consist of non-functional, functional, and the behavioral aspects of a Web service
  - A Web service is a computational entity which is able (by invocation) to achieve a users goal. A service in contrast is the actual value provided by this invocation
WSMO Goals

• Goals are representations of an objective for which fulfillment is sought through the execution of a Web service. Goals can be descriptions of Web services that would potentially satisfy the user desires.

Class goal sub-Class wsmoElement
importsOntology type ontology
usesMediator type {ooMediator, ggMediator}
hasNonFunctionalProperties type nonFunctionalProperty
requestsCapability type capability multiplicity = single-valued
requestsInterface type interface

Example:
• A person wants to book a flight from Innsbruck to New York
WSMO Mediators

• Mediation
  – Data Level - mediate heterogeneous Data Sources
  – Protocol Level - mediate heterogeneous Communication Patterns
  – Process Level - mediate heterogeneous Business Processes
• Four different types of mediators in WSMO
  – ggMediators: mediators that link two goals. This link represents the refinement of the source goal into the target goal or state equivalence if both goals are substitutable
  – ooMediators: mediators that import ontologies and resolve possible representation mismatches between ontologies
  – wgMediators: mediators that link Web services to goals, meaning that the Web service (totally or partially) fulfills the goal to which it is linked. wgMediators may explicitly state the difference between the two entities and map different vocabularies (through the use of ooMediators)
  – wwMediators: mediators linking two Web services
Web Service Modeling Language (WSML)

Conceptual Model & Axiomatization for SWS

Formal Language for WSMO

Execution Environment for WSMO

Ontology & Rule Language for the Semantic Web
A set of concrete languages for the various tasks:

- Ontology / Rule Languages (static view)
  - WSML Core
    - efficiency and compatibility
  - WSML DL
    - decidability, open world semantics
  - WSML Rule
    - efficient existing rule engines
  - WSML Full
    - unifying language, theorem proving

- Languages for dynamics
  - Transaction Logic over ASMs

- Mapping languages
  - for dynamics (process mediation)
  - or data (data mediation)
WSML Variants

- WSML Variants - allow users to make the trade-off between the provided expressivity and the implied complexity on a per-application basis
WSML Variants (cont’)

- **WSML-Core** - defined by the intersection of Description Logic and Horn Logic, based on **Description Logic Programs**
  - It has the least expressive power of all the languages of the WSML family and therefore has the most preferable computational characteristics
  - The main features of the language are the support for modeling classes, attributes, binary relations and instances
  - Supports **class hierarchies**, as well as **relation hierarchies**
  - Provides support for **datatypes** and **datatype predicates**
WSML Variants (cont’)

• **WSML-DL** - an extension of WSML-Core which fully captures the Description Logic $SHIQ(D)$, which captures a major part of the (DL species of the) Web Ontology Language OWL

• **Differences** between WSML-DL and OWL-DL:
  – No support for nominals (i.e. enumerated classes) as in OWL
  – Allows to write enumerations of individuals
    
    oneOf(Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday)
  – Support for Qualified Cardinality Restrictions (QCR)
    • Used to constrain the number of values of a particular property
    • ”The normal hand has exactly five fingers of which one is a thumb”
    • OWL does not support QCR: Class(NormalHand restriction (hasFinger cardinality (5)))
    • No possibility to model the fact that one finger is a thumb in OWL
• **WSML-Flight** - an extension of WSML-Core with meta-modeling, constraints and nonmonotonic negation features
  – Based on a logic programming variant of F-Logic
  – Semantically equivalent to Datalog with inequality and (locally) stratified negation
  – Provides a powerful rule language

• **WSML-Rule** - an extension of WSML-Flight in the direction of Logic Programming
  – Captures several extensions such as the use of function symbols and unsafe rules, and does not require stratification of negation
  – The semantics for negation is based on the Stable Model Semantics
• **WSML-Full** - unifies WSML-DL and WSML-Rule under a First-Order syntactic umbrella with extensions to support the nonmonotonic negation of WSML-Rule
  – Allows the full syntactic freedom of a First-Order logic and the full syntactic freedom of a Logic Programming language with default negation in a common semantic framework

• Compared to WSML-DL, WSML-Full adds full first-order modeling: n-ary predicates, function symbols and chaining variables over predicates

• Compared to WSML-Rule, WSML-Full adds disjunction, classical negation, multiple model semantics, and the equality operator
Web Service Modeling Execution Environment (WSMX)

Conceptual Model & Axiomatization for SWS

Formal Language for WSMO

Ontology & Rule Language for the Semantic Web

Execution Environment for WSMO

STI2 CMS WG

OASIS SEE TC
• ... is comprehensive software framework for runtime binding of service requesters and service providers,
• ... interprets service requester’s goal to
  – discover matching services,
  – select (if desired) the service that best fits,
  – provide data/process mediation (if required), and
  – make the service invocation,
• ... is reference implementation for WSMO,
• ... has a formal execution semantics, and
• ... is service oriented, event-based and has pluggable architecture
  – Open source implementation available through Source Forge,
  – based on microkernel design using technologies such as JMX.
WSMX - Design principles

- **Service-oriented principle**
  - Service reusability, loose coupling, abstraction, composability, autonomy, discoverability,

- **Semantic Principle**
  - Rich and formal description of information and behavioral models enabling automation of certain tasks by means of logical reasoning,

- **Problem-solving principle**
  - Goal-based discovery and invocation of services, and

- **Distributed principle**
  - Executing process across a number of components/services over the network, thus promoting scalability and quality of process.
Lifecyle

1. Discovery - determines usable services for a request,
2. Composition - combine services to achieve a goal,
3. Selection - chooses most appropriate service among the available ones,
4. Mediation - solves mismatches (data, protocol, process) hampering interoperation,
5. Choreography – interactions and processes between the service providers and clients,
6. Grounding – lifting and lowering between the semantic and syntactic data representations, and
7. Invocation - invokes Web service following programmatic conventions.
WSMX
Current middleware status

Web Service Execution Environment
Core Management
Execution Semantics

Communication Manager
Discovery
Selection and Ranking
Mediation
Choreography
Invoker
Grounding

Resource Manager Interface
Goals
Services
Ontologies
Mediators
Reasoner

Network
Client

Provider X
Provider Y
WSMX Components
Communication Manager, Invoker and Grounding

- Responsible for interaction with services and entities that are external to WSMX.
- Should be open to support as many transport and messaging protocols as possible (transparently to WSMX).
- WSMX uses
  - The SOAP implementation from Apache AXIS, and
  - The Apache Web Service Invocation Framework (WSIF).
- Both RPC and Document style invocations possible
• WSMO service descriptions are grounded to WSDL by the means of XSLT lifting and lowering

An example of lifting XML data to RDF
WSMX Components
Discovery

• Responsible for finding appropriate Web Services capable of fulfilling a goal
• Different techniques available
  – trade-off: ease-of-provision vs. accuracy
  – resource descriptions & matchmaking algorithms

Key Word Matching
  - match natural language key words in resource descriptions,

Controlled Vocabulary
  - ontology-based key word matching, and

Semantic Matchmaking
  - what Semantic Web Services aim at.
WSMX Components
Discovery – Key Word Matching

- Allows for a fast filtering and ranking of the huge number of available services rather quickly.
- Nonfunctional properties from the Dublin Core namespace (e.g. dc#description) are candidates for indexing and querying.
- Dictionaries of synonyms (WordNet) can be used to discover more services.

```
wsmlVariant _"http://www.wsmo.org/wsml/wsml-syntax/wsml-rule"

namespace {"http://www.wsmo.org/sws-challenge/WSMuller#",
    dc _"http://purl.org/dc/elements/1.1#"}

webService WSMuller
    nfp
        dc#title hasValue "Muller Web Service"
        dc#description hasValue "We ship to Africa, North America, Europe, Asia (all countries)."
        dc#contributor hasValue "Maciej Zaremba, Matt Moran, Tomas Vitvar, Thomas Haselwanter"
    endnfp

capability WSMullerCapability
...
```
WSMX Components
Discovery – Simple Semantic Description

Exact Match:
\[ G, WS, O, M \models \forall x. (G(x) \iff WS(x)) \]

PlugIn Match:
\[ G, WS, O, M \models \forall x. (G(x) \implies WS(x)) \]

Subsumption Match:
\[ G, WS, O, M \models \forall x. (G(x) \subseteq WS(x)) \]

Intersection Match:
\[ G, WS, O, M \models \exists x. (G(x) \land WS(x)) \]

Non Match:
\[ G, WS, O, M \models \neg \exists x. (G(x) \land WS(x)) \]

WSMX Components
Discovery – Simple Semantic Description - Example

Goal_TripAT2DE ≡ ∃hasPostCondition.(Trip\n ∃start.AustrianCity \n∃end.GermanCity)

Goal_TripAustria ≡ ∃hasPostCondition.(Trip\n ∃start.AustrianCity \n∃end.AustrianCity)

Goal_TripEurope ≡ ∃hasPostCondition.(Trip\n ∃start.EuropeanCity \n∃end.EuropeanCity)

Generic goals

Goal_TripIBK2FRA ≡ ∃hasPostCondition.(Trip\n ∃start.{Innsbruck} \n∃end.{Frankfurt})

Goal_TripIBK2SZG ≡ ∃hasPostCondition.(Trip\n ∃start.{Innsbruck} \n∃end.{Salzburg})

Specific goals

Service_VTA ≡ ∃hasPostCondition.(Trip\n ∃start.(GermanCity \nAustrianCity)\n∃end.(GermanCity \nAustrianCity))

Service_DB ≡ ∃hasPostCondition.(Trip\n ∃start.GermanCity \n∃end.GermanCity)

Service_OEBB ≡ ∃hasPostCondition.(Trip\n ∃start.AustrianCity \n∃end.AustrianCity)

Service_DBNight ≡ ∃hasPostCondition.(NightTrip\n ∃start.{Innsbruck \nSalzburg} \n∃end.GermanCity)

Web services

WSMX Components
Discovery – Simple Semantic Description - Example

• Exact match
  
  (1) Goal_TripAustria \equiv Service_OEBB
  
  (2) Goal_TripAT2DE \subseteq Service_VTA
  
  (3) Goal_TripIBK2FRA \subseteq Service_VTA
  
  (4) Goal_TripAustria \subseteq Service_VTA
  
  (5) Goal_TripIBK2SZG \subseteq Service_VTA
  
  (6) Goal_TripIBK2SZG \subseteq Service_OEBB
  
  (7) Service_VTA \subseteq Goal_TripEurope
  
• Plug-in match
  
• Subsumption match
  
  (8) Service_DB \subseteq Goal_TripEurope
  
  (9) Service_OEBB \subseteq Goal_TripEurope
  
  (10) Service_DBNight \subseteq Goal_TripEurope
  
• Intersection match
  
  (11) \neg (Goal_TripIBK2FRA \cap Service_DBNight \subseteq \bot)
WSMX and SESA

- WSMX is an implementation of SESA.

- SESA represents SOA empowered by adding semantics as a means to deal with heterogeneity and mechanization of service usage.

- Application of SESA offers a scalable integration, more adaptive to changes
  - service offerings and required capabilities are described by semantically rich and formal service models,
  - exchanged data is also semantically described, and
  - reasoning provides total or partial automation of tasks.

- A SESA implementation should build a layer on top of the existing technologies (e.g. Web Services).
Semantically Enabled SOA (SESA)

Semantic Execution Environment

- Discovery
- Ranking
- Selection
- Mediation
- Process Execution
- Lifting & Lowering
SESA Architecture

• Middleware for Semantic Web Services
  – Allows service providers to focus on their business,
• Environment for goal based discovery and invocation
  – Run-time binding of service requesters and providers,
• Provide a flexible Service Oriented Architecture
  – Add, update, remove components at run-time as needed,
• Keep open-source to encourage participation
  – Developers are free to use in their own code, and
• Define formal execution semantics
  – Unambiguous model of system behavior.
Other SWS Frameworks

- WSMO is not the only initiative aimed towards Semantic Web services
- Other major initiatives in the area are documented by recent W3C member submissions:
  - OWL-S,
  - SWSF, and
  - WSDL-S/SAWSDL (i.e. METEOR-S).
- Other implementations of WSMO:
  - IRS-III.
• **Semantic Annotations** for **WSDL** and XML Schema
• W3C Recommendation, August, 2007
• Largely based on WSDL-S
• A simple, incremental approach
  – Builds naturally on the WSDL-centric view of Web services
• 3 extensibility elements
  – `modelReference`
  – `liftingSchemaMapping`
  – `loweringSchemaMapping`
• Can be used in both WSDL and XML Schema documents
• Values are lists of URIs
• No Preconditions and Effects

```xml
<wsdl:description>
  <wsdl:types>
    <xs:schema elementFormDefault="qualified">
      <xs:element name="OrderRequest"

        sawsdl:modelReference="..."
        sawsdl:liftingSchemaMapping="..."

        sawsdl:loweringSchemaMapping="...">

        ...
    </xs:element>
  </xs:schema>
</wsdl:types>

<wsdl:interface name="Order">

  sawsdl:modelReference="...">

  <wsdl:operation name="order"

    pattern="...

    sawsdl:modelReference="...">

  <wsdl:input element="OrderRequest" />
  <wsdl:output element="OrderResponse" />
</wsdl:operation>
</wsdl:interface>
</wsdl:description>
```
• May be used with every element within WSDL
• “However, SAWSDL defines its meaning only for
  – wsdl:interface
  – wsdl:operation
  – wsdl:fault
  – xs:element
  – xs:complexType
  – xs:simpleType
  – xs:attribute.”

```xml
<wsdl:description>
  <wsdl:types>
    <xs:schema elementFormDefault="qualified">
    </xs:schema>
  </wsdl:types>
  <wsdl:interface name="Order" sawsdl:modelReference="http://.../products/electronics">  
    <wsdl:operation name="order" pattern="..." sawsdl:modelReference="http://ontology/po#RequestPurchaseOrder">  
      <wsdl:input element="OrderRequest" />
      <wsdl:output element="OrderResponse" />
    </wsdl:operation>
  </wsdl:interface>
</wsdl:description>
```
Schema Mapping Attributes

- **liftingSchemaMapping**
  - *lift* data from XML to a semantic model
- **loweringSchemaMapping**
  - *lower* data from a semantic model to XML
- Can map to XSLT script

```xml
<wsdl:description>
  <wsdl:types>
    <xs:schema elementFormDefault="qualified"> 
      <xs:element name="OrderRequest" 
        sawsdl:liftingSchemaMapping="http://.../mapping/Response2Ont.xslt" 
        sawsdl:loweringSchemaMapping="http://.../mapping/Ont2Request.xml">
        </xs:element>
    </xs:schema>
  </wsdl:types>
  <wsdl:interface name="Order" 
    sawsdl:modelReference="..."> 
    <wsdl:operation name="order" 
      pattern="..." 
      sawsdl:modelReference="..."> 
      <wsdl:input element="OrderRequest" />
      <wsdl:output element="OrderResponse" />
    </wsdl:operation>
  </wsdl:interface>
</wsdl:description>
```
But: no predefined semantics!
WSMO-Lite

WSMO-Lite Ontology

Annotations point to

SAWSDL
extends
WSDL
layer of semantic annotations

MicroWSMO
extends
hRESTS
service description layer
Semantics in Service Model

Web service

Operation 1

Operation 2

\[ \ldots \]

Operation N

- F
- N
- B
- I

input
output
input
output
input
output
Functional Semantics

- For service discovery, composition

- **Category**
  - Functionality categorization
  - E.g. eCl@ss, UDDI
  - Or tagging, folksonomies

- **Capability**
  - Precondition, Effect
  - Using WSML rule languages
Category Example

wl:FunctionalityClassificationRoot

ex:eCommerceService

subclasses

ex:Travel ReservationService

ex:Accommodation ReservationService

type
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 .
```
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
Information Semantics

- For invocation, composition, data mediation
- Not constrained, any ontologies
- Refer to course *Semantic Web (WS)*
WSMO-Lite in SAWSDL

 Ontology element
 Lifting, Lowering
 Capability, Category
 Capability, Category
 Nonfunctional property

 WSDL Description
 Types
 Element Declaration
 Type Definition
 Interface
 Operation
 Operation
 Binding
 Operation
 Fault
 Fault
 Fault
 Service
 Endpoint
<wsdl:description>
<wsdl:types>  <xs:schema>
<xs:element name="ReservationRequest"
  sawsdl:modelReference="&ex;Reservation"
  sawsdl:loweringSchemaMapping="&ex;ResMapping.xsparql" … />
</xs:schema>  </wsdl:types>
<wsdl:interface name="HotelReservations"
  sawsdl:modelReference= "&ex;AccommodationReservationService">
<wsdl:operation name="searchForRooms"
  sawsdl:modelReference="&wsdlx;SafeInteraction">
  …
</wsdl:operation>
  …
</wsdl:interface>
  …
<wsdl:service name="RomaHotels" interface="HotelReservations"
  sawsdl:modelReference="&ex;RomaHotelReservationPrecondition
dec;ReservationFee" … />
</wsdl:description>
• Extends hRESTS
  – model for model references
  – lifting, lowering
• Applies WSMO-Lite semantics
<div class="service" id="svc">
  <p><span class="label">ACME Hotels</span> is a
      <a rel="model" href="../ecommerce/hotelReservation">
          hotel reservation</a> service.</p>
  <div class="operation" id="op1">
      <p>…</p>
      <span class="input">A particular hotel ID replaces the param
          <a rel="model" href="../onto.owl#Hotel"><code>id</code></a>
          (<a rel="lowering" href="../hotelID.xslt">lowering</a>).<br/>
      </span>
  </div>
</div>
Semantics Implied in Web

- Hypermedia → behavioral semantics
  - Links become available through interaction
- Uniform interface → functional semantics
  - GET, PUT, DELETE have known effects
  - GET is safe, PUT and DELETE idempotent
- Self-description → information model
  - Operation output data can specify what it is
    - GRDDL, other semantic annotations
EXTENSIONS
Google – Unified Cloud Computing

- An attempt to create an open and standardized cloud interface for the unification of various cloud API’s
- Key drivers of the unified cloud interface is to create an api about other API's
- Use of the resource description framework (RDF) to describe a semantic cloud data model (taxonomy & ontology)
“People as a service”

- Amazon Mechanical Turk
  - An API to Human Processing Power
  - The Computer Calls People
  - An Internet Scale Workforce
  - Game-Changing Economics
Amazon – S3 & EC2

“Infrastructure as a service”

- **Amazon Simple Storage Service (S3)**
  - Write and read objects up to 5GB
  - 15 cents GB / month to store
  - 20 cents GB / month to transfer

- **Amazon Elastic Compute Cloud (EC2)**
  - allows customers to rent computers on which to run their own computer applications
  - virtual server technology
  - 10 cents / hour
Summary

Why Semantic Web services?

- To overcome limitations of traditional Web-Services Technology by integrating it with Semantic Technology;
- To enable automatic and personalized service discovery;
- To enable automatic service invocation and execution monitoring;
- To enable automatic service integration;
- To enable semantic mediation of Web-Services.
REFERENCES
• Mandatory reading

• Further reading
  – Steve Battle et al., *Semantic Web Services Framework (SWSF)*, W3C Member Submission 9 September 2005, http://www.w3.org/Submission/SWSF
References

• Wikipedia and other links
  – http://en.wikipedia.org/wiki/Web_2.0
  – http://www.wsmo.org/
  – http://cms-wg.sti2.org/
  – http://www.wsmo.org/wsml
  – http://www.wsmx.org/
  – http://www.oasis-open.org/committees/semantic-ex/
<table>
<thead>
<tr>
<th>#</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introduction</td>
</tr>
<tr>
<td>2</td>
<td>Semantic Web Architecture</td>
</tr>
<tr>
<td>3</td>
<td>Resource Description Framework (RDF)</td>
</tr>
<tr>
<td>4</td>
<td>Web of data</td>
</tr>
<tr>
<td>5</td>
<td>Generating Semantic Annotations</td>
</tr>
<tr>
<td>6</td>
<td>Storage and Querying</td>
</tr>
<tr>
<td>7</td>
<td>Web Ontology Language (OWL)</td>
</tr>
<tr>
<td>8</td>
<td>Rule Interchange Format (RIF)</td>
</tr>
<tr>
<td>9</td>
<td>Reasoning on the Web</td>
</tr>
<tr>
<td>10</td>
<td>Ontologies</td>
</tr>
<tr>
<td>11</td>
<td>Social Semantic Web</td>
</tr>
<tr>
<td>12</td>
<td>Semantic Web Services</td>
</tr>
<tr>
<td>13</td>
<td><strong>Tools</strong></td>
</tr>
<tr>
<td>14</td>
<td>Applications</td>
</tr>
</tbody>
</table>
Questions?