

Semantic Web Technologies

Semantic Web Services

Jos de Bruijn
debruijn AT inf.unibz.it

KRDB Research Group
Free University of Bolzano, Italy

12 December 2008

Outline

Semantic Web Services Basics

- The Vision

- Web Services

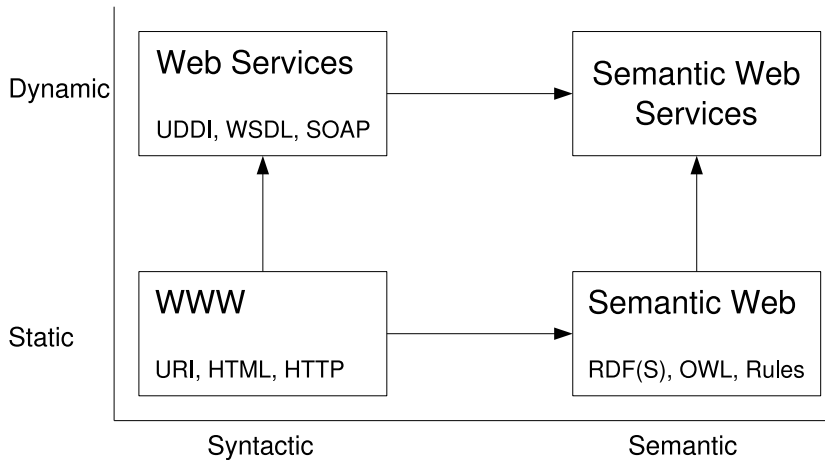
- Service Oriented Architecture

Challenges in Web Services

Semantics in Web Services

Web Service Modeling Ontology

The Vision of Semantic Web Services



The World Wide Web

- ▶ Largest document repository ever (> 25 billion Web pages indexed by Google)
- ▶ Highly distributed
 - ▶ Millions of publishers
 - ▶ No control over consistency of published content
- ▶ Web Technologies
 - ▶ HTTP for transferring documents
 - ▶ HTML for marking up documents
 - ▶ URI for addressing documents
- ▶ Most content on the Web is in natural language (HTML)
 - ▶ Natural language not suitable for machine reading
 - ▶ Current Web is “syntactic”
 - ▶ Problems in automatically:
 - ▶ Retrieving documents
 - ▶ Extracting relevant information from retrieved documents
 - ▶ Combining information from different sources

The Semantic Web

- ▶ Making the Web machine-readable
- ▶ Publishing data in machine-readable format
- ▶ Relating data on the Web to established vocabularies (ontologies)
- ▶ Ontologies specified in formal language to allow reasoning
- ▶ Ontologies enable automation in:
 - ▶ Retrieval of relevant information
 - ▶ Extracting relevant information from retrieved document
 - ▶ Combination of information from different sources (as long as they are related to the same ontology)

Web Services

- ▶ Next step in software engineering:
 - ▶ 1960s: Procedural languages
 - ▶ 1980s: Object Orientation
 - ▶ 1990s: Component Orientation
 - ▶ 2000s: Web Services
- ▶ Loosely coupled, reusable components
- ▶ Add new level of functionality to the Web
- ▶ Web Service Technologies
 - ▶ SOAP for accessing Web Services
 - ▶ WSDL for describing Web Services
 - ▶ UDDI for publishing and looking up Web Services

Web Services are not enough

- ▶ Like the current Web, Web Services are “syntactic”
- ▶ No automation in:
 - ▶ Finding services
 - ▶ Selecting services
 - ▶ Negotiation with service provider
 - ▶ Composing services
 - ▶ Executing services

Combining Semantic Web and Web Services

Semantic Web Services

- ▶ Semantic Web + Web Services = Semantic Web Services
- ▶ Using Semantic Web technologies to describe Web Services
- ▶ Enable automation in:
 - ▶ Publication
 - ▶ Discovery
 - ▶ Selection
 - ▶ Composition
 - ▶ Mediation
 - ▶ Execution

Business Vision of Web Services

- ▶ 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.
- ▶ Dynamic enterprise and dynamic value chains become achievable and possibly even mandatory for competitive advantage.

Current Web Service Technologies

- ▶ UDDI
 - ▶ Registry of services
- ▶ WSDL
 - ▶ Service description
- ▶ SOAP
 - ▶ Message format

UDDI

- ▶ UDDI provides a mechanism for clients to find web services.
- ▶ A UDDI registry is similar to a CORBA trader, or it can be thought of as a DNS service for business applications.
- ▶ **UDDI consists of:**
 - ▶ **White pages:** Who is the service provider?
 - ▶ **Yellow pages:** What is the service providing?
 - ▶ **Green pages:** How can I make use of the service?
- ▶ Public UDDIs
 - ▶ Microsoft, IBM, others
 - ▶ Multitude of services is registered
 - ▶ Most services no longer available
 - ▶ Not used in practice
 - ▶ Private UDDIs

WSDL

▶ Port type

- ▶ Collection of **operations**
 - ▶ Name
 - ▶ Input/output

▶ Message formats

- ▶ Content defined through **XML Schema**

▶ Services

▶ Ports

- ▶ Endpoint (network addresses)
- ▶ Protocol (e.g. SOAP, RMI)
- ▶ Port type
- ▶ Message format

SOAP

- ▶ SOAP
 - ▶ used to stand for Simple Object Access Protocol
 - ▶ is a message layout specification that defines a uniform way of passing XML-encoded data.
- ▶ Binding to HTTP as communication protocol.
- ▶ SOAP is basically a technology to allow for “RPC over the web”.
- ▶ **SOAP Message**
 - ▶ **Envelope**
 - ▶ Sender address
 - ▶ Receiver address
 - ▶ Intermediate nodes
 - ▶ Security information
 - ▶ Format of the body
 - ▶ **Body**
 - ▶ Message content
 - ▶ Typically XML

Other Web Service Standards

- ▶ WS-Addressing
 - ▶ Defining endpoints of services
- ▶ WS-Security
 - ▶ Comprehensive security framework for web services
- ▶ WS-Policy
 - ▶ Specification of policies for access, pricing, etc...
- ▶ WS-RF (Web Service Resource Framework)
 - ▶ Specification for **Grid computing**
 - ▶ Accessing **resources** (e.g. data, processor time, disk space)
- ▶ *etc...etc...etc...*
 - ▶ Responsible: Microsoft, IBM, BEA, Sun, Netscape, ...

Current use of Web Services

- ▶ On the Web
 - ▶ Amazon
 - ▶ Finding products
 - ▶ Adding, deleting, updating shopping cart
 - ▶ Payment still over the Web
 - ▶ Google
 - ▶ Search Google via Web Service API
 - ▶ Accessing other Google services
 - ▶ Weather, stock tickers, etc...
 - ▶ Most public services use HTTP/GET, not SOAP
- ▶ In Enterprises
 - ▶ Many companies started implementing Web Services
 - ▶ Example: Verizon
 - ▶ Before WSs: information about telephone numbers stored in 50 places
 - ▶ With WSs: one Web Service to retrieve telephone numbers
 - ▶ 1500 Business functions implemented using Web Services
 - ▶ Custom registry with services

Advertising Services

- ▶ Current repositories (UDDI)
 - ▶ “polluted”
- ▶ Advertise functionality of the service
- ▶ Non-functional aspects
 - ▶ Security
 - ▶ Accessibility
 - ▶ Cost

Discovering Services

- ▶ Finding services which provide **required functionality**
- ▶ Requester needs to **specify** requirements
- ▶ Mechanism for matching **required** with **advertised** functionality
- ▶ Currently no way of specifying requirements
- ▶ Discovery currently **manual**

Selecting Services

- ▶ Given a set of services which meet requirements, select service with
 - ▶ **best** availability,
 - ▶ **lowest** cost,
 - ▶ **best** security measures,
 - ▶ etc..

- ▶ Select best service wrt. set of preferences
 - ▶ Specification of requester's preferences
 - ▶ Specification of non-functional properties of service

- ▶ Selection currently **manual**

Service Composition

- ▶ Required functionality requires **multiple** services to be invoked in **some combination**
 - ▶ Sequential execution
 - ▶ Parallel execution
 - ▶ loops
- ▶ Specifying requirements
 - ▶ Single requirement specification vs.
 - ▶ workflow description with identified tasks
- ▶ Single requirement specification
 - ▶ **Find** composition which fulfills requirement
- ▶ Workflow description
 - ▶ Discovery of services for each task
 - ▶ Optimizing preferences when selecting services

- ▶ Composition currently manual process specification (e.g. BPEL, WSFL)

Service Execution

- ▶ How to invoke a service?
- ▶ If requester and provider use same format, current technologies suffice.
- ▶ What if actual formats differ?
- ▶ ⇒ Translation of message content

Semantics in Web Services

- ▶ Description of
 - ▶ Functionality of services
 - ▶ e.g., searching, selling books, booking flight
 - ▶ User requirements
 - ▶ e.g., finding, buying books, booking flight
 - ▶ Web Service policies
 - ▶ Permissions
 - ▶ Cost
 - ▶ etc...
 - ▶ User preferences
 - ▶ Data in messages
- ▶ Matching descriptions
 - ▶ functionality of service vs. user requirements
 - ▶ policies vs. user preferences
- ▶ Data translation
- ▶ Web Service Composition
 - ▶ Finding compositions
 - ▶ Matching composition with user requirements

Functional Description of Services

- ▶ Describe functionality of Services
- ▶ Similar to software specification
- ▶ View web service as a function: $input \longrightarrow output$
- ▶ Coarse-grained approach:
 - ▶ Functionality described as single task
 - ▶ e.g., “book sales”, “flight reservation”
 - ▶ Organizing tasks in hierarchy
 - ▶ Interpreting task as set \Rightarrow subsumption reasoning
 - ▶ No clear relation between input and output
- ▶ Fine-grained approach:
 - ▶ Specify conditions on the input: **preconditions**
 - ▶ Specify relation between input and output: **postconditions**
 - ▶ Intuitively: $(\forall)preconditions \Rightarrow postconditions$

DLs (OWL DL) for functional description

- ▶ Model service as OWL concept S
- ▶ S represents set of possible service executions
- ▶ Model required functionality as DL concept R
- ▶ R represents set of service executions which fulfill requirements
- ▶ Relation between C, D determines match

Notions of matching

- ▶ **Exact match:** $S \equiv R$
 - ▶ Service provides **exactly** the requested functionality
- ▶ **PlugIn:** $R \sqsubseteq S$
 - ▶ Service provide **more** than the requested functionality
- ▶ **Subsume:** $S \sqsubseteq R$
 - ▶ Service provides **some** of the requested functionality, and nothing more
- ▶ **Intersection:** $\neg(S \sqcap R \sqsubseteq \perp)$
 - ▶ Service provides **some** of the requested functionality, and perhaps more
- ▶ **Disjoint:** $S \sqcap R \sqsubseteq \perp$
 - ▶ Service provides **none** of the requested functionality

Rules for Policy Description / Contracts

- ▶ Policies are often in the form of rules
 - ▶ e.g., “if user is recognized business partner, then give discount”
 - ▶ “if used encryption method is at least DES-256, then credit card detail may be transferred”
- ▶ Use Semantic Web rules language (RIF) for specifying policies
- ▶ Initiatives for using rules to specify web service policies (e.g., policy RuleML)

Semantic Web languages for data description

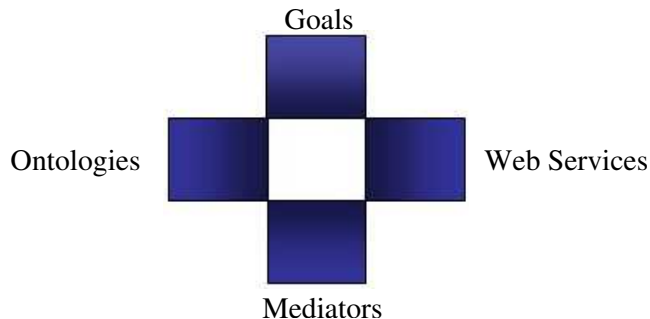
- ▶ Transfer data in RDF format
 - ▶ Advantage: More flexible than XML
 - ▶ Advantage: using ontology for describing structure of data
- ▶ Use ontologies for describing structure of data
 - ▶ Problem: which data to send over the wire
- ▶ If both parties use same ontology, no translation necessary
- ▶ Data transformation based on ontology mapping (rules)

The Web Service Modeling Ontology WSMO

Introduction

- ▶ An ontology for Semantic Web Services
- ▶ Provides conceptual model for SWS
- ▶ Principles of WSMO:
 - ▶ Ontology-based descriptions
 - ▶ Strict decoupling of components
 - ▶ Mediation between components
 - ▶ Interface, not implementation

The Web Service Modeling Ontology WSMO



Ontologies

- ▶ Provide terminology for:
 - ▶ Data exchanged between service requesters and providers
 - ▶ Description of other WSMO elements
- ▶ Ontologies consist of:
 - ▶ Concepts
 - ▶ Relations
 - ▶ Functions
 - ▶ Instances
 - ▶ Axioms

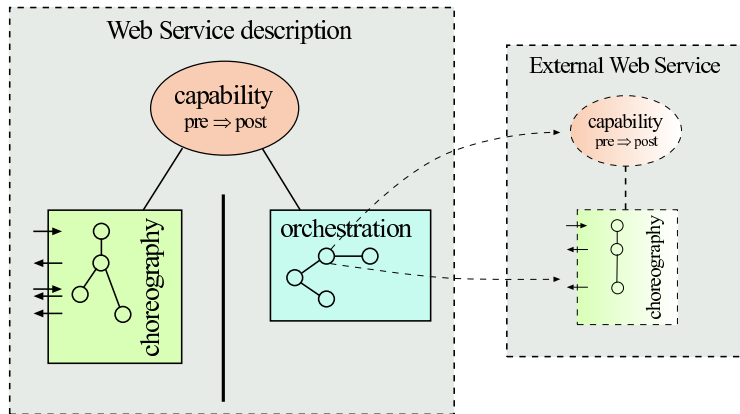
Web Service descriptions

- ▶ Functionality offered by the Web Service
- ▶ Functional description, in the form of a **capability**:
 - ▶ Assumptions
 - ▶ Cannot be checked
 - ▶ Usually indicate dependency on real world
 - ▶ Preconditions
 - ▶ Conditions over the input
 - ▶ Effects
 - ▶ Changes in the real world as a result of execution of the Web Service
 - ▶ Postconditions
 - ▶ Relation between the input and the output

Web Service descriptions (cont'd)

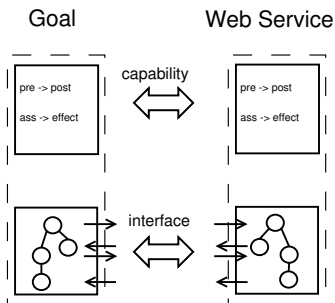
- ▶ Behavioral description, in the form of an **interface**:
 - ▶ Choreography
 - ▶ How to interact with the service
 - ▶ Orchestration
 - ▶ Use of external Web Service to realize the functionality
 - ▶ Both choreography and orchestration are decompositions of the capability

Web Service descriptions (cont'd)



Goals

- ▶ Functionality requested from the Web Service
- ▶ Description symmetric to Web Service description:
 - ▶ Capability
 - ▶ Interface



WSMO Process

