Towards Semantic Service-Oriented Systems on the Web
~ An Overview ~

Dumitru Roman

Digital Enterprise Research Institute (DERI)
Recent trends around the Web

Web Pages
- HTTP
- HTML
- URL
- StyleSheet

Web Applications
- Applet
- Servlet
- JSP/ASP
- JDBC/ODBC

Web Services
- XML
- SOAP
- WSDL
- UDDI

Web 2.0
- AJAX
- Folksonomy
- RSS / ATOM

Semantic Web
- XML
- RDF/S
- OWL

Content
- Information

Knowledge

Making Semantic Web real.
Outline

• Semantic Web
• Web Services
• Semantic Web Services (SWS)
  – Tasks to be Automated
• Existing Approaches to SWS
  – OWL-S, SWSF, IRS-III, WSDL-S
  – The WSMO Approach: WSMO, WSML, WSMX
• Conclusions
• Proposed Challenge for Measuring Success of SWS
Outline

• **Semantic Web**
• Web Services
• Semantic Web Services (SWS)
  – Tasks to be Automated
• Existing Approaches to SWS
  – OWL-S, SWSF, IRS-III, WSDL-S
  – The WSMO Approach: WSMO, WSML, WSMX
• Conclusions
• Proposed Challenge for Measuring Success of SWS
First some definitions from W3C …

• The Semantic Web provides a common framework that allows data to be shared and reused across application, enterprise, and community boundaries
  – The Semantic Web is a web of data

• The Semantic Web is about two things
  – It is about common formats for integration and combination of data drawn from diverse sources, where on the original Web mainly concentrated on the interchange of documents
  – It is also about language for recording how the data relates to real world objects
The Web…and it’s problems

Billions of diverse documents online; problems in:
  – Retrieving documents
  – Extracting relevant data from retrieved documents
  – Combining information from different sources to achieve a particular goal
Retrieving documents

Which one is my picture?
Extracting information

Which book is about the Web?


What is the price of the book?

Charlotte’s Web (Widescreen Edition) by E.B. White, Robert McCloskey, and Audie Cole

Buy new: $29.95 $16.99 Used it new from $7.49

Get it by Thursday, May 10, if you order in the next 5 hours and 41 minutes.

Eligible for FREE Super Saver Shipping.

Also Available For Download From Amazon Unbox


Buy new: $29.95 $25.17 Used it new from $22.89

Get it by Thursday, May 10, if you order in the next 5 hours and 41 minutes.

Eligible for FREE Super Saver Shipping.

Books: See all 335,356 items


Buy new: $24.95 $16.49 Used it new from $7.15

Get it by Thursday, May 10, if you order in the next 7 hours and 41 minutes.

Eligible for FREE Super Saver Shipping.

Books: See all 335,356 items

Making Semantic Web real.
Combining information

I want the cheapest copy of the book “A Semantic Web Primer”, taking into account the price for shipping the book!

On average 10 clicks to find out what the shipping rate is!!!
The solution!

- Instead of publishing natural language, publish machine-processable data
- Publish information in terms understandable for a machine
- Ask questions in terms understandable for a machine
- And: make sure all machines understand your terms!

=> The Semantic Web!
Publishing and querying machine processable data (cont’d)

• Publishing (related-to is transitive):

  B related-to A
  C related-to A
  D related-to C


• Querying (give me all things related to A):

  ?x related-to A

  Answer:
  ?x = B
  ?x = C
  ?x = D
What is an **ontology**?

- **Formal**, explicit specification of a shared conceptualization of a domain.

- Meaning of ontology is **unambiguous**
- Avoids misunderstanding
- Specification using **formal language**
- Enables **reasoning**: making implicit information explicit
- Hampers **consensus**

- Make domain assumptions **explicit**
  - For **reasoning**
  - For **clarifying** understanding of domain
- Minimal ontological commitment
  - Too much explicit => **no consensus**
  - Too little explicit => ontology **unusable**
  - Minimal ontological commitment = “make as little as explicit as possible, while keeping ontology useful”

- Domain: specific part of the world
- Conceptualization
  - Forming idea of domain in the **minds** of people
- Shared among its users
  - Facilitates accepting the ontology

-Digital Enterprise Research Institute www.deri.org

-Making Semantic Web real.
Elements of Ontologies

- **Classes**
  - Grouping of individuals with common properties
  - e.g. Persons, Cars, Universities, ...

- **Relations**
  - Connections between individuals
  - May be attached to classes
  - e.g. hasName, hasAge, owns, ...

- **Individuals**
  - Objects in the domain
  - May be instances of classes

- **Axioms**
  - Additional statements about the domain
  - Specified in logical language
  - e.g. “hasName has one value”

Ontologies and the Semantic Web

- Form the backbone of the Semantic Web
- Define the basic vocabulary for the annotations
- Enable reasoning with background knowledge, based on formal languages
- Interweave meaning for humans and machines
- Are shared
A wide variety of languages for Ontologies

- **Graphical**: Semantic Networks, Topic Maps, UML, RDF

- **Logical**: Description Logics, First Order Logic, Rules, Conceptual Graphs

<table>
<thead>
<tr>
<th>DL Syntax</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_1 \cap \ldots \cap C_n$</td>
<td>Human (\cap) Male</td>
</tr>
<tr>
<td>$C_1 \cup \ldots \cup C_n$</td>
<td>Doctor (\cup) Lawyer</td>
</tr>
<tr>
<td>$\neg C$</td>
<td>(\neg)Male</td>
</tr>
<tr>
<td>$\forall P.C$</td>
<td>(\forall)hasChild.Doctor</td>
</tr>
<tr>
<td>$\exists P.C$</td>
<td>(\exists)hasChild.Lawyer</td>
</tr>
<tr>
<td>$\leq n P$</td>
<td>(\leq)1hasChild</td>
</tr>
<tr>
<td>$\geq n P$</td>
<td>(\geq)2hasChild</td>
</tr>
</tbody>
</table>

**Brothers are siblings**

\[
\forall x, y \quad \text{Brother}(x, y) \Rightarrow \text{Sibling}(x, y).
\]

“Sibling” is symmetric

\[
\forall x, y \quad \text{Sibling}(x, y) \Leftrightarrow \text{Sibling}(y, x).
\]

One’s mother is one’s female parent

\[
\forall x, y \quad \text{Mother}(x, y) \Leftrightarrow (\text{Female}(x) \land \text{Parent}(x, y)).
\]

A first cousin is a child of a parent’s sibling

\[
\forall x, y \quad \text{FirstCousin}(x, y) \Leftrightarrow \exists p, s \quad \text{Parent}(p, x) \land \text{Sibling}(p, s) \land \text{Parent}(p, s, y).
\]
The Evolution of the Semantic Web

2001

(Tim Berners-Lee)

2006

(Tim Berners-Lee)
RDF and RDF(S) – example
### RDFS Entitlement - example

<table>
<thead>
<tr>
<th>Statement 1</th>
<th>Statement 2</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;http://example.org/#Student&gt;</code> rdfs:subClassOf <code>&lt;http://example.org/#Person&gt;</code></td>
<td></td>
</tr>
</tbody>
</table>

### entails

<table>
<thead>
<tr>
<th>Statement 1</th>
<th>Statement 2</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;http://example.org/#hasName&gt;</code> rdfs:domain <code>&lt;http://example.org/#Student&gt;</code></td>
<td><code>&lt;http://example.org/#mary&gt;</code> <code>&lt;http://example.org/#hasName&gt;</code> “Mary”</td>
</tr>
</tbody>
</table>

### entails

<table>
<thead>
<tr>
<th>Statement 1</th>
<th>Statement 2</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>&lt;http://example.org/#mary&gt;</code> rdf:type <code>&lt;http://example.org/#Student&gt;</code></td>
<td></td>
</tr>
</tbody>
</table>

### entails

<table>
<thead>
<tr>
<th>Statement 1</th>
<th>Statement 2</th>
</tr>
</thead>
</table>
Outline

• Semantic Web
• **Web Services**
• Semantic Web Services (SWS)
  – Tasks to be Automated
• Existing Approaches to SWS
  – OWL-S, SWSF, IRS-III, WSDL-S
  – The WSMO Approach: WSMO, WSML, WSMX
• Conclusions
• Proposed Challenge for Measuring Success of SWS
Web Services – distributed computing background

- Evolution of distributed computing

- Problems with existing models
  - Proprietary protocols
  - Interoperability
  - Platform lock-in
  - Flexibility

Downsize
- Client/Server

Components
- CORBA
- DCOM
- EJB

Messaging
- MOM

SOA
Web Services
Web Services

- W3C: “The World Wide Web is more and more used for application to application communication. The programmatic interfaces made available are referred to as Web services”

- A multitude of Web services specifications “WS*-”:

  - SOAP ASAP  WS-Addressing
  - MTOM  WS-Enumeration WS-Eventing
  - WS-Transfer
  - WSDL WSRF
  - WS-Remote Portlet
  - WS-Reliability WS-ReliableMessaging
  - WS-Acknowledgement WS-Coordination
  - WS-Atomic Transaction
  - WS-BusinessActivity
  - WS-TransactionManagement
  - WS-Reliability WS-Remote Portlet
  - UDDI WS-Discovery
ebXML Registry
  - WSIL WS-Notification
  - WS-Management WS-Management Catalog
  - WS-Manageability WS-Provisioning
  - WS-Distributed Management (WSDM)
  - WS-Security : SOAP Message Security
  - WS-Security: Username Token Profile
  - WS-Security: X.509 Certificate Token Profile
  - WS-Security Kerberos Binding
  - WS-Security Minimalist Profile
  - WS-SecureConversation WS-Authorization SAML
  - WS-Security Policy
  - WS-Encryption WS-Signature
  - WS-Trust WS-Federation
  - WS-BPEL
  - WS-CDL WSCI
  - WS-Business Process Execution Language
  - WS-Coordination
  - WS-TransactionManagement

Making Semantic Web real.
Basic Web Services

Syntax only!
Some 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 did not realize the promise of Web Services
Outline

• Semantic Web
• Web Services
• Semantic Web Services (SWS)
• Existing Approaches to SWS
  – OWL-S, SWSF, IRS-III, WSDL-S
  – The WSMO Approach: WSMO, WSML, WSMX
• Conclusions
• Proposed Challenge for Measuring Success of SWS
Semantic Web and Web Services - SWS

It’s all about automation !!!

Dynamic

Web Services
UDDI, WSDL, SOAP

Semantic Web Services

Static

WWW
URI, HTML, HTTP

Semantic Web
RDF, RDF(S), OWL, etc.

Making Semantic Web real.
SWS – 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 Enactment & Monitoring**
- Invoke & Monitor services following programmatic conventions

**Service Mediation**
- Heteroginity is Everywhere

**Service Composition**
- Combine services to achieve a goal
An Example of a SWS Usage Process

1. **Request**
   - *Submission*
   - *if: directly usable*
   - *if: composition needed*
   - *if: compatible*
   - *if: successful*
   - *if: execution error*

2. **Discoverer**
   - *uses*
   - *if: composition needed*
   - *uses*
   - *if: compatible*
   - *uses*
   - *matchmaking R with all WS*
   - *else: try other WS*

3. **Composer**
   - *uses*
   - *composition (executable)*
   - *uses*
   - *information lookup for particular service*
   - *else: try other WS*

4. **Executor**
   - *uses*
   - *uses*
   - *else: try other WS*

5. **Data Mediator**
   - *uses*
   - *else: try other WS*

6. **Process Mediator**
   - *uses*
   - *else: try other WS*

7. **Service Repository**
So what is needed?

• **Mechanized support** is needed for
  – Annotating/designing services and the date 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
Outline

- Semantic Web
- Web Services
- Semantic Web Services (SWS)
  - Tasks to be Automated
- **Existing Approaches to SWS**
  - OWL-S, SWSF, IRS-III, WSDL-S
  - The WSMO Approach: WSMO, WSML, WSMX
- Conclusions
- Proposed Challenge for Measuring Success of SWS
OWL-based Web service ontology (OWL-S)

• Conceptual Model
  – A set of ontologies used to describe different aspects SWS

• Language: OWL

• Some OWL-S drawbacks
  – OWL not sufficiently expressive for all aspects of a service
    • more expressive languages have been syntactically integrated: SWRL, KIF, DRS, and PDDL – how do these languages interoperate?
  – Inherits some of the drawbacks of OWL (e.g. lack of proper layering, improper use of OWL for describing and reasoning about processes)
  – No explicit support for Mediation in the language
Semantic Web Services Framework (SWSF)

- Two major components: an ontology and a language used to axiomatize it
- Semantic Web Services Ontology (SWSO) – an extension of OWL-S conceptual model, e.g. a rich behavioural process model based on PSL
  - FLOWS – First-Order Logic Ontology for Web Services
  - ROWS - Rule Ontology for Web Services
- The Semantic Web Services Language (SWSL)
  - SWSL-FOL - based on First Order Logic; includes features from HiLog and F-Logic
  - SWSL-Rules - a logic programming language; includes features from Courteous logic programs, HiLog, and F-Logic
- Some SWSF drawbacks
  - unclear how all the paradigms part of this approach work together
  - first-order logic ontology for Web services, but not a Web language
Internet Reasoning Service (IRS-III)

- A platform which acts as a broker mediating between the goals of a user or client and available deployed web services
- Not a SWS framework on its own but uses WSMO as its ontology and follows the WSMO design principles
- IRS Architecture:
Web Service Semantics - WSDL-S

- A mechanism to augment WSDL descriptions with semantics
  - a set of annotations can be created to semantically describe the inputs, outputs and operations of a Web service.
  - keeps the semantic model outside WSDL, making the approach agnostic to any ontology representation language

- WSDL-S doesn’t provide a conceptual model and language for SWS
  - a bottom up approach to SWS (annotating existing standards with metadata)
- Could be used as a grounding mechanism for SWS
Outline

• Semantic Web
• Web Services
• Semantic Web Services (SWS)
  – Tasks to be Automated
• Existing Approaches to SWS
  – OWL-S, SWSF, IRS-III, WSDL-S
  – The WSMO Approach: WSMO, WSML, WSMX
• Conclusions
• Proposed Challenge for Measuring Success of SWS
The WSMO Approach

WSMO WG

A Conceptual Model for SWS

European Semantic Systems Initiative

WSML WG

A Formal Language for WSMO

WSMX WG

An Execution Environment for WSMO
The Web Service Modeling Ontology (WSMO)

Design Principles

- Web Compliance
- Ontology-Based
- Strict Decoupling
- Ontological Role Separation
- Centrality of Mediation
- Execution Semantics
- Description versus Implementation
Top-level elements defined by WSMO

Objectives that a client may have when consulting a Web Service

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
WSMO – the Ontology element

• Ontology elements:
  – **Concepts** - set of concepts that belong to the ontology
  – **Attributes** - set of attributes that belong to a concept
  – **Relations** - define interrelations between several concepts
  – **Instances** - set of instances that belong to the represented ontology
  – **Axioms** - axiomatic expressions in ontology (logical statements)
  – **Non-functional properties**
  – **Imported ontologies** - importing existing ontologies where no heterogeneities arise
  – **Used mediators** - ontology import with terminology mismatch handling

• Ontologies - used as the ‘data model’ throughout WSMO
  – all WSMO element descriptions rely on ontologies
  – all data interchanged in Web Service usage are ontologies
  – Semantic information processing & ontology reasoning
WSMO – the Web service element
The big challenges of defining a WSMO service

- **Capabilities**
  - What is a service able to do?
  - What are the requirements on the input and output?
  - **Preconditions, Assumptions, Postconditions and Effects need to be defined.**

- **Interfaces**
  - How can a service be accessed?
  - How does a service solve its task?
  - **Choreography and Orchestration of services need to be defined.**
WSMO – Goals, Mediators

• **Goals**
  – Defined in a similar way as WSMO Web services

• **Mediation**
  – Data Level - mediate heterogeneous Data Sources
  – Protocol Level - mediate heterogeneous Communication Patterns
  – Process Level - mediate heterogeneous Business Processes

• **WSMO Mediators:**
  – *OO Mediators* - terminology import with data level mediation
  – *WW Mediators* - enable interoperability of heterogeneous Web Services
  – *WG Mediators* - link a Web Service to a Goal and resolve occurring mismatches
  – *GG Mediators* – Support specs of goals by reusing exiting goals
WSMO and the other SWS approaches

<table>
<thead>
<tr>
<th>Scope</th>
<th>OWL-S</th>
<th>WSMO</th>
<th>SWSF</th>
<th>WSDL-S</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Top Level Elements</strong></td>
<td>Service Profile, Process Model, Grounding</td>
<td>Ontologies, Goals, Web Services, Mediators</td>
<td>Processes</td>
<td>Operations / WSDL descriptions</td>
</tr>
<tr>
<td><strong>Service Level Description</strong></td>
<td>non-functional aspects IOPE for service-level functional description</td>
<td>capability (PAPE) for provided and requested functionality</td>
<td>not in the scope</td>
<td>keyword classification (ontology-based)</td>
</tr>
<tr>
<td><strong>Operation Level Description</strong></td>
<td>IOPE for processes</td>
<td>interfaces for consumption (choreography) and interaction (orchestration)</td>
<td>internal behavior (atomic and composite processes)</td>
<td>preconditions &amp; effects for WSDL operations</td>
</tr>
<tr>
<td><strong>Language (static)</strong></td>
<td>OWL</td>
<td>WSML</td>
<td>SWSLFOL &amp; SWSLRules</td>
<td>not specified</td>
</tr>
<tr>
<td><strong>Language (dynamic)</strong></td>
<td>Process Model and OWL</td>
<td>Abstract State Machines</td>
<td>FLOWS</td>
<td>not specified</td>
</tr>
</tbody>
</table>
The Web Service Modelling Language (WSML)

- Ontology / Rule Languages
  - 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)
  - for data (data mediation)
WSML – relation to SW standards
WSMO/WSML – Some Modelling Examples

• Concept example
  concept phoneNumber
  nonFunctionalProperties
    dc#description hasValue "concept of a phone number"
  endNonFunctionalProperties
  countryCode ofType _string
  areaCode ofType _string
  number ofType _string

• Sub-concept example
  concept mobilePhoneNumber subConceptOf phoneNumber
  nonFunctionalProperties
    dc#description hasValue "concept of a mobile phone number"
  endNonFunctionalProperties
  mobileProvider ofType Provider

• Relation example
  relation hasRoute(ofType routeDescription, ofType route)
  nonFunctionalProperties
    dc#description hasValue "Relation that holds between a route description and a route"
  endNonFunctionalProperties

• Instance example
  instance myPhoneNumber memberOf phoneNumber
  countryCode hasValue "43"
  areaCode hasValue "664"
  number hasValue "49322607"

• Axiom example
  axiom ValidInformationQuality
  definedBy
    forall {?x} (
      ?x memberOf informationQualityType
      implies
      ?x[value hasValue "low"] or
      ?x[value hasValue "high"]).
webService _"https://asg-platform.org/AttractionBooking/MobtelPhoneLocationService"

nfp
dc#title hasValue "MobtelPhoneLocationService"
dc#publisher hasValue "Mobtel"
dO#informQualityType hasValue "high"
endnfp

importsOntology _"https://asg-platform.org/AttractionBooking/domainOntology.wsml"
capability MobtelPhoneLocationServiceCapability
sharedVariables {?P}

precondition
   definedBy
   ?P memberOf dO#phoneNumber.

postcondition
   definedBy
   ?L memberOf dO#location
   and
   dO#hasLocation(?P,?L).

interface MobtelPhoneLocationServiceInterface
choresography MobtelPhoneLocationServiceChoreography
stateSignature
   in
   dO#phoneNumber withGrounding
   ssWSDL#wsdl.interfaceMessageReference(MobtelPhoneLocationServicePortType/doIt/In)
   out
   dO#location withGrounding
   ssWSDL#wsdl.interfaceMessageReference(MobtelPhoneLocationServicePortType/doIt/Out)
transitionRules
   forAll {?P} with (?P memberOf dO#phoneNumber) do
      add (?L memberOf dO#location and dO#hasLocation(?P,?L))
endForall
The Web Service Execution Environment (WSMX)

- A software framework for runtime binding of service requesters and service providers
- WSMX interprets service requester’s goal to
  - discover matching services
  - select (if desired) the service that fits best
  - provide mediation (if required)
  - make the service invocation
- Is based on the conceptual model provided by WSMO
- Has a formal execution semantics
- SO and event-based architecture based on microkernel design using technologies as J2EE, Hibernate, JMX, etc.
WSMX Motivation

- Provide middleware ‘glue’ for Semantic Web Services
  - Allow service providers focus on their business
- Provide a reference implementation for WSMO
- Provide an environment for goal based service discovery and invocation
  - Run-time binding of service requester and provider
- 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
- Define formal execution semantics
  - Unambiguous model of system behaviour
WSMX Usage Scenario

Three Tier Architecture

- User (Service Requester)
  - Client / Browser
  - WSMX Client

- Retailer
  - Web Server / Application Server
  - WSMX Server

Manufacturer

- WSMX
- Back-End Application
- Adapter / WSDL Interface
- Back-End Application

Two Tier Architecture

B2B

B2C

Making Semantic Web real.
WSMX Components

Service Requester

Service Provider

WSML

WSML

WSML

Internet

Management and Monitoring

WSMO4J Editor

WSMX Core (Space-based messaging)

WSMO4J Parser

Discovery

Orchestration

Process Mediation

Communication

Data Mediation

Choreography

New Component

Reasoner

WSMO4J Repository

Making Semantic Web real.
WSMX Discovery Component

- **Functionality**
  - Identify possible web services W which are able to provide the requested service S for its clients

- **An important issue ...**
  - “being able to provide a service“ has to be determined based on given descriptions only (WS, Goal, Ontos)
  - Discovery can *only be as good as* these descriptions
    - *Very detailed WS descriptions*: are precise, enable highly accurate results, are more difficult to provide; in general, requires interaction with the provider (outside the pure logics framework)
    - *Less detailed WS descriptions*: are easy to provide for humans, but usually less precise and provide less accurate results
Support a wide-variety of applications wrt. needed accuracy

Basic possibilities for the description of web services:

- **Syntactic approaches**
  - Keyword-based search, natural language processing techniques, Controlled vocabularies

- **Lightweight semantic approaches**
  - Ontologies, What does W provide (not how)?, Action-Object-Modelling, Coarse-grained semantic description of a service

- **Heavyweight semantic approaches**
  - Describes the service capability in detail, Pre/Post-Cond, takes „in-out“ relationship into account, Fine-grained web service description
WSMO Discovery - Basic idea for Matching on the single levels

Common keywords

Set-theoretic relationship

Adequate (common) execution/state-transition

{Keyword}  
W1 ... WL  K1 ... Kn

WS  
X

Level of Abstraction

Syntactic

Semantic ("Light")

Semantic ("Heavy")
WSMO Discovery Process

Goal-Repos.

Predefined formal Goal

Requester Desire

Goal Discovery

Selected predefined Goal

Requester Goal

Goal refinement

Available WS

Abstract Capability

Web Service Discovery

Concrete Capability (possibly dynamic)

Still relevant WS

Web Service (Service Discovery)

Service to be returned

Ease of description

Efficient Filtering

Accuracy
WSMX Reasoner Component – an Overview

- **WSMO4J**
  - validation, serialization and parsing
- **WSML2Reasoner**
  - Reasoning API
    - mapping from WSML to a vendor-neutral rule representation
  - Contains:
    - Common API for WSML Reasoners
    - Transformations of WSML to tool-specific input data (query answering or instance retrieval)
- **WSML-DL-Reasoner features**:
  - T-Box reasoning (provided by FaCT++)
  - Querying for all concepts
  - Querying for the equivalents, for the children, for the descendants, for the parents and for all ancestors of a given concept
  - Testing the satisfiability of a given concept with respect to the knowledge base
  - Subsumption test of two concepts with respect to the knowledge base
  - Wrapper of WSML-DL to the XML syntax of DL used in the DIG interface

**Mins**
- Datalog + Negation + Function Symbols Reasoner Engine
- Features
  - Built-in predicates
  - Function symbols
  - Stratified negation

**Other Reasoner Engines**, e.g. Kaon2, DLV and Flora

**WSML2 Reasoner**

**WSML-DL-Reasoner**

**FaCT++**

**Other Reasoner Engines**
Outline

• Semantic Web
• Web Services
• Semantic Web Services (SWS)
  – Tasks to be Automated
• Existing Approaches to SWS
  – OWL-S, SWSF, IRS-III, WSDL-S
  – The WSMO Approach: WSMO, WSML, WSMX
• Conclusions
• Proposed Challenge for Measuring Success of SWS
But first what about the predicted impact of semantic technologies? (Gartner, Oct. 2005)

Strategic Planning Assumption: Enterprise clusters that adopt robust semantic B2B specifications and Web services will achieve savings of 10 percent or more in trading processes by year-end 2007 (0.7 probability).

Standards Power B2B Commerce After 2008 'Sweet Spot' is Reached

The semantic Web emphasizes XML and machine-readable content

Development and testing remaining on core Web services standards stack: SOAP, WSDL, UDDI, BPEL, WS-Events, others.

2000  2008  2010

Boom in B2B Trading Efficiencies

2008 Sweet Spot

Growth of semantic B2B standards

These Standards Are All About Meaning of Terms OWL, RDF, UCC/Rosettanet, HL 7, Accord, others

Making Semantic Web real.
Semantic Web – Conclusions

- The Semantic Web is **real** – tremendous progress in the past five years
  - Growing support in industry and govt use
  - Lots of tools out there:
    - **Browsers**: mSpace, Longwell, OINK, BrownSauce, Piggy Bank, Tabulator, etc
    - **Annotators**: Annotea, Clipmarks, PhotoStuff, M-OntoMat-Annotizer, KIM, WSMT
    - **Storages**: Oracle Spatial 10g, Kowari, Jena, Yars, 3Store, AllegroGraph, Joseki, ARC RDF Store
    - **Ontology Mappers**: OntoMerge, HMARFA, CMS
    - **Reasoners**: BOR, Bossam, FaCT++, Jess, OWLJessKB, RacerPro
    - **Composite Applications/Frameworks**: Cerbera, Corse, IODT, Jena, TopBraid Composer, KAON
- New languages under way
  - GRDDL/RDFa – integration of HTML world and Semantic Web
  - RIF (Rules interchange format) – representing rules on the Web
  - And more: Multimedia annotation, Web-page Metadata annotation, Health Care and Life Science, Privacy
- Easy to get involved – many open source tools; new languages and techniques reaching critical mass
- …and research opportunities still abound: scaling, inconsistency, access and acquisition, etc.
Semantic Web Services – Conclusions

- Although lots of progress in the last couple of years, SWS are still immature technologies; not too many use cases
  - But high potential in **BPM**: B2B, EAI, eCommerce, etc.
- The WSMO Approach to SWS looks promising
  - Covers many aspects of SWS; unifying approach
  - Large-scale ongoing initiative supported by both industry and academia
- Standardization activities are emerging in this area
  - OWL-S, SWSF, WSDL-S, WSMO – submitted to W3C
  - OASIS SEE technical committee formed (based on WSMX)
  - W3C SAWSDL Working Group formed; close to recommendation
- More collaboration is needed between research community and the industrial community
- The biggest challenge for the future: the movement to service-orientation and the semantic enablement of industrial scale infrastructures and applications
Outline

• Semantic Web
• Web Services
• Semantic Web Services (SWS)
  – Tasks to be Automated
• Existing Approaches to SWS
  – OWL-S, SWSF, IRS-III, WSDL-S
  – The WSMO Approach: WSMO, WSML, WSMX
• Conclusions
  • Proposed Challenge for Measuring Success of SWS
Proposed Challenge for Measuring Success of SWS

• An industrial-scale application that has run successfully in a production environment for at least six months; the application must:
  – Support some form of collaboration substantially involving at least five internal or external, but separate, organizations
  – Consist of at least 1,000 entity types and 1,000 service types with at least 2 subscribers per service type.
  – Have an average daily service transaction rate of 10 million service executions.
  – Comply with at least three industrial standard ontologies and the majority of relevant SOA and Web service standards.
  – Support at least 20 concurrent domain-specific problem solvers.
  – Support problem solving in at least two distinct aspects (e.g., ordering and billing) of standard problem domain, e.g., manufacturing, financial services, health care, inventory, and tourism. This requires at least:
    • 10 industry-specific tools of which 5 must be standard industry practice
    • 10 industry-specific problem solving capabilities supported by the tools
    • Automatic workspace configuration: The application must automatically configure the workplace for each user with the tools and capabilities required by each user as defined in their profile which defines their roles, responsibilities, and user-specified or system-deduced configuration preferences.
Proposed Challenge for Measuring Success of SWS (cont’)

• At least 50% of service discovery, selection, negotiation, adaptation, composition, invocation, and monitoring, as well as service interaction requiring data, protocol, and process mediation -- are fully automated, with no human intervention.
  – At a minimum, this must dynamically and automatically address and resolve the conflicting non-functional business aspects that arise when a consumer discovers a service offered by a producer with whom there is no business agreement for the discovered service.

• Normal business: All normal business conventions must apply. There must be significant, e.g., legal and financial, consequences should there be a failure in any of the above automated service operations.

• All service offerings and requests are expressed in terms of service descriptions that contain:
  – a functional and behavioural specification expressed in semantic terms consistent with one or more industrial standard ontologies, and
  – a non-functional specification consisting of at least 5 non-functional terms such as price, promised service levels (SLAs), and performance characteristics.
• **STI2** - Semantic Technology Institutes International
  - the leading international think tank in this field unifying three major initiatives:

  - DERI Innsbruck – a founding member of STI2
STI2 Services

More details: [www.STI2.org](http://www.STI2.org)
Thank you!

Questions?
Acknowledgements

- This tutorial is based on the WSMO Tutorial Series, series to which several people contributed. For the list of contributors to the WSMO tutorials, and the various WSMO tutorials, please check [http://www.wsmo.org/TR/d17/](http://www.wsmo.org/TR/d17/)

- Some of the slides related to ontologies are based on the Semantic Web lecture at University of Innsbruck, slides which were initially developed by Jos de Brujin and Stijn Heymans