OWL-S Tools and Applications

The OWL-S Coalition

presented by Massimo Paolucci
Organization

- **OWL-S Authoring Tools**
  - KSL OWL-S Editor
  - CMU WSDL2OWL-S
  - Mind-Swap Ontolink

- **Web Service Discovery**
  - CMU OWL-S/UDDI Matchmaker
  - KSL Semantic Discovery Service
  - CMU OWL-S Broker
  - CMU OWL-S for P2P

- **Automatic WS Invocation**
  - CMU OWL-S Virtual Machine

- **Web Service Composition**
  - Mind-Swap Composer
  - KSL Composition Tool
  - CMU Computer Buyer

- **Applications**
  - Fujitsu Task Computing
  - CMU DAMLzon: OWL-S for Amazon
**Goal**: Editor tailored to the markup of Web Services in OWL-S (not just an ontology editor -- focus on end user needs and intuitions)

**Input**: graphical and form entry  
**Output**: OWL-S & Ontolingua

**Anticipated Users**:  
- Web service providers/developers  
- OWL community

**Approach**:  
- Graphical  
- Ontology editors (OILed Protégé) and reasoner behind the scenes

**Value added by reasoning**:  
- Verification of properties of services  
- Simulation of services  
- Diagnostics
KSL OWL-S Editor

Draw the control structure for composite services
Finally, generate the OWL-S for the services
WSDL2OWL - OMI/OntoLink

DEMO
upon request

www.daml.ri.cmu.edu/wsdldamls
Mapping WSDL to OWL-S

- Exploits relation between WSDL and OWL-S to generate (partial) OWL-S specification
  - Automatic generation of Grounding
  - Partial generation of Process Model and Profile
  - Up to 80% of work required to generate a OWL-S description is done automatically
  - Allows programmers to concentrate on the information that is really different between the two Web services descriptions

- Combined with Java2WSDL to provide Java2OWL-S
Contribution

- Tool to facilitate generation of OWL-S
- Methodology to generate OWL-S
- In constant use by the community since Spring 2003
Moving On…

Tools for Web Service Discovery

- **OWL-S Authoring Tools**
  - KSL OWL-S Editor
  - CMU WSDL2OWL-S
  - Mind-Swap Ontolink

- **Web Service Discovery**
  - CMU OWL-S/UDDI Matchmaker
  - KSL Semantic Discovery Service
  - CMU OWL-S Broker
  - CMU OWL-S for P2P

- **Automatic WS Invocation**
  - CMU OWL-S Virtual Machine

- **Web Service Composition**
  - Mind-Swap Composer
  - KSL Composition Tool
  - CMU Computer Buyer

- **Applications**
  - Fujitsu Task Computing
  - CMU DAMLzon: OWL-S for Amazon
Three Models of Discovery

- Matchmaking
- Broker
- P2P
Matching Engine

• Core of all discovery mechanisms is a Matching Engine that
  – takes Requester description of *ideal* Web Service to interact with
  – advertisements of providers
  – Matching Engine finds Web Service(s) that more closely fit the description
  – Result is a flexible matching which shows the relation between advertisement and request

• Extensions: matching on additional properties:
  – **Security:**
    • Match security requirements of Requester and Provider
**OWL-S 4 UDDI**

- **UDDI** is the de-facto standard registry for Web services
  - UDDI Provides keyword search of Web Services
  - **NO CAPABILITY SEARCH**
    - It is impossible to find a WS that does ...
- **OWL-S 4 UDDI** integrates OWL-S Matching engine within UDDI
  - **PROVIDES CAPABILITY SEARCH**
  - Leverages on OWL-S semantic representation
Contribution

- OWL-S encoded in UDDI allows an expansions of the registry functionalities adding capability-based discovery

- OWL-S4UDDI provides *clear evidence of the contribution of Semantic web to Web services technology*

- In collaboration with Toshiba (Japan), the DAML-S matchmaker is currently available on the NTT UDDI registry (Main UDDI provider in Japan).
Semantic Discovery Service

• We argue that:
  – Web Services must embrace representation and reasoning ideas from Semantic Web community
  – Must also recognize evolutionary influence of industry standards and machinery on Semantic Web services

• From this viewpoint, we build on BPEL4WS, a leading choreography framework

• Integrate Semantic Web technology to enable automated service discovery, customization, and semantic translation

• Our efforts take the form of a *Semantic Discovery Service* (SDS)
Contribution

- By integrating the SDS with BPWS4J, the industrial system gained the following abilities:
  - Automatic, runtime binding of service partners
  - Selection between multiple service partners based on user-defined constraints
  - Integration of service partners with syntactically distinct but semantically translatable service descriptions

- Does not automate composition of Web services, which requires:
  - Well-defined operational semantics describing functional behavior of service partners
  - Automated reasoning machinery to manipulate them
• Broker performs both discovery and mediation for a client

• Challenging OWL-S:
  – OWL-S Process Model describes an interaction between 2 parties: a provider and a requester
  – Broker introduces third parties
Extension to OWL-S

- Dynamic loading of the Process Model
  1. The Broker publishes an initial Process Model
  2. During the interaction the Broker communicates a new Process Model
  3. The Broker and the Requester adopt to the new Process Model for the rest of their interaction

- First step toward multiparty interactions
  - Same mechanism can be used for
    - Modeling Auctions
    - Modeling transactions requiring third parties

- Supports automatic composition of Web Services
P2P Discovery

- No Centralized registry
  - NO UDDI/Matchmaker
  - NO BROKER
- Discovery based on message passing between peers
- Useful for ad-hoc networks and ubiquitous computing
- Support switch from file-sharing to service-sharing
Basic idea

- **Advertisement**
  - Web services advertise using P2P network
  - Requesters may store advertisements

- **Request**
  - Requesters broadcast requests for services using P2P network
  - Providers match their capabilities with the request and respond when the match is positive

- **Transport**
  - Based on Gnutella network
Contribution

- Show how OWL-S can be used in P2P networks
  - We describe a Web services discovery protocol that makes use of Gnutella for connectivity and OWL-S for capability descriptions
- Supports use of OWL-S in ad-hoc network and ubiquitous computing
- Support switch from file-sharing to data/service sharing
Moving On…
Automatic WS Interaction

- **OWL-S Authoring Tools**
  - KSL OWL-S Editor
  - CMU WSDL2OWL-S
  - Mind-Swap Ontolink

- **Web Service Discovery**
  - CMU OWL-S/UDDI Matchmaker
  - KSL Semantic Discovery Service
  - CMU OWL-S Broker
  - CMU OWL-S for P2P

- **Automatic WS Invocation**
  - CMU OWL-S Virtual Machine

- **Web Service Composition**
  - Mind-Swap Composer
  - KSL Composition Tool
  - CMU Computer Buyer

- **Applications**
  - Fujitsu Task Computing
  - CMU DAMLzor: OWL-S for Amazon
• OWL-S VM a generic processor for the OWL-S Process Model
  – It can interact with any OWL-S Web service
  – Based on the Process Model formal semantics (Ankolekar et al 2002)
  – Exploits Web services technology such as Axis and WSIF
Contribution

• **OWL-S VM** can be used to automatically invoke **OWL-S** Web services
  – It conforms with the **OWL-S** semantics
  – It is based on **OWL** inference engine

• The use of **OWL-S** does not result in a performance penalty
  – In interactions with Amazon.com only 8% of time was devoted to the **OWL-S VM**
Moving On…
Web Service Composition

- **OWL-S Authoring Tools**
  - KSL OWL-S Editor
  - CMU WSDL2OWL-S
  - Mind-Swap Ontolink

- **Web Service Discovery**
  - CMU OWL-S/UDDI Matchmaker
  - KSL Semantic Discovery Service
  - CMU OWL-S Broker
  - CMU OWL-S for P2P

- **Automatic WS Invocation**
  - CMU OWL-S Virtual Machine

- **Web Service Composition**
  - Mind-Swap Composer
  - KSL Composition Tool
  - CMU Computer Buyer

- **Applications**
  - Fujitsu Task Computing
  - CMU DAMLzon: OWL-S for Amazon
MindSwap’s Web Service Composer

- Demonstrates how tasks can be composed from different OWL-S services.
- Leads the user (via a web interface) through a top-down dynamic view of the composition.
- Generates a composition that is directly executable through WSDL groundings.

http://www.mindswap.org/~evren/composer/
Contribution

• WS composition environment
  – Uses SHOP2, a well established planner
  – Contains an OWL-S execution environment

• Used for many applications of WS composition ranging from
  – Information gathering
  – Language translation
  – etc...
Problem: Automated Web Service Composition
E.g., Make my travel arrangements for the DAML PI Meeting

Approach:
I. Plan a sequences of services that realize user’s objective.
   (NP complete or worse)

II. Customize reusable generic procedures
   - Define and archive reusable **generic procedures**
   - Customize with **user’s constraints**.
   (NP complete or worse in a reduced search space)

Advantages: efficiency, ease of use, customization
Status & Challenges

Implementation:

✓ DAML+OIL/DAML-S FOL -> Ontolingua, Golog & sit’n calculus in Prolog

✓ Java, Prolog, Ontolingua-DAML+OIL translator, OKBC, DAML-S to PDDL translator, bubble gum, scotch tape

Challenges:

• Conversion to OWL-S and JTP
• OWL-S-ize our work; Reduce number of repn’s required.
• Technical challenges:
  • Execution Monitoring & Recovery, Info vs. world-chging services
  • Automate Service Selection
  • Low-level synchronization, message passing and grounding issues
CMU Composition Architecture

- Exploits Retsina Architecture for WS composition
  - OWL-S/UDDI Matchmaker for discovery
  - Retsina planner to control the agent
    - Use interleaving of planning and execution to allow communication while planning
  - OWL Reasoner
  - OWL-S Virtual Machine to communicate with other Web Services
Contribution

- Used in a number of applications: travel domain, supply chain management
- Supports composition and execution of Web Services
- Connection with autonomous agent technology

in collaboration with TOSHIBA
Moving On…

Applications

- **OWL-S Authoring Tools**
  - KSL OWL-S Editor
  - CMU WSDL2OWL-S
  - Mind-Swap Ontolink

- **Web Service Discovery**
  - CMU OWL-S/UDDI Matchmaker
  - KSL Semantic Discovery Service
  - CMU OWL-S Broker
  - CMU OWL-S for P2P

- **Automatic WS Invocation**
  - CMU OWL-S Virtual Machine

- **Web Service Composition**
  - Mind-Swap Composer
  - KSL Composition Tool
  - CMU Computer Buyer

- **Applications**
  - Fujitso Task Computing
  - CMU DAMLzon: OWL-S for Amazon
Task Computing

- User wants to do “Tasks” while on the run
  - email – printing – sharing documents – complex tasks

Gap: the user should use configurations and “leg work” to use the tools

- Services offered in the environment

UPnP: Universal Plug and Play
“Task Computing” is computation to fill the gap between a user’s tasks and the available means.

Help users with access Services (Web based and not) and
- Discovery using UPnP
- Composition
- Manipulated at execution time, not at the design time

Use:
- Semantics (OWL-S)
Task Computing

- Task Computing Environment
  - Implemented using RDF, OWL, DAML-S and pervasive computing discovery (UPnP).
  - Supports composition of services within a pervasive environment
  - Generates DAML-S process models which are then enacted

DAMLzon: DAML-S for Amazon.com

- Amazon provides a publicly available WS
- OWL-S was derived automatically using WSDL2DAML-S
- OWL-S VM used to interact with Amazon Web Service
Performance

- DAML-S VM client on browsing+reserving task
- Analyzed data by computing:
  - Time required by DAML-S VM to execute Process Model
  - Time required for data transformation to fit Amazon requirements
  - Time required to invoke an operation on Amazon
- 98 runs total over 4 days in varying load conditions
- Results in milliseconds

<table>
<thead>
<tr>
<th></th>
<th>VM</th>
<th>Data Trsfm</th>
<th>Invocation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average percentage</td>
<td>83</td>
<td>156</td>
<td>2797</td>
</tr>
<tr>
<td></td>
<td>3%</td>
<td>5%</td>
<td>92%</td>
</tr>
<tr>
<td>Strd dev</td>
<td>107</td>
<td>146</td>
<td>1314</td>
</tr>
</tbody>
</table>
Conclusion

- Good number of tools available or under construction
- Transition to OWL-S already undergoing
- A number of early adopters both from DAML program and outside