Carnegie Mellon University

Katia Sycara

katia@cs.cmu.edu

http://www.cs.cmu.edu/~softagents
Overview

- Goals for 2004: Tools and Standards
- OWL-S Editor and Integrated Development Environment
- Enhancements to other OWL-S Tools
- Additional Research Activities
Implementation of OWL-S WS requires:

1. Implementation of WS logic
2. Generation of WSDL description
3. Generation of OWL-S description
4. Deployment of WS
5. Advertisement with registries such as UDDI
Implementation of client requires
1. Interpretation of OWL-S specification
2. Generation of messages to the WS

OWL-S IDE addresses both aspects
1. Support programmers in their generation of OWL-S starting from Java code

- Exploit Apache’s Java2WSDL and CMU’s WSDL2OWL-S to generate OWL-S descriptions

- Produces:
  - Grounding
  - Schematic Process model
  - Schematic Profile
  - WSDL description
Editing and Validation Tools

- **Form based Profile and Process Editor**
  - Guarantees Syntactic correctness
  - Validation of data flow

- **Management tools**
  - Automatic publishing OWL-S description files on Web site
  - Automatic generation of OWL-S client code

Based on popular Eclipse java IDE

- Provide a uniform environment for Java and OWL-S development
OWL-S Menu:
Java 2 OWL-S

Eclipse’s Java IDE

Java Code

Resulting OWL-S
OWL-S Actions
- Publish on Web site
- Publication to UDDI Verification

Adding/Editing processes
- Adding/Editing Inputs/outputs Preconditions/effects

Process Model tree
- Display process/subprocess relations
- Inputs/outputs Preconditions/effects

Ontology files
Completed development of

- Transformation Java->OWL-S
- Profile and Process Model editors
- Advertising with UDDI
- Generation of client code almost complete
**Editor for Grounding**
- Facilitate generation of XSLT script that maps WSDL data types into OWL classes

**Surface Syntax editor**

**Security and Policy annotations**
- Annotation of Atomic Processes and WSDL consistent with WS security

**Integration with Ontology Browsers to support selection of ontologies and concepts**

**Integration with Inference Engines to support querying and inferencing**
- Graphic display of Process model
- Verification of OWL-S code
  - Detection of workflow errors
  - Verification of XSLT scripts
- Development process driven by the suggestions of the users
- The CMU editor does not only focus on the generation of OWL-S code but on the whole WS generation environment
OWL-S Development Tools

- Updated our WSDL2DAML-S Tool to WSDL2OWL-S. Available at [http://www.daml.ri.cmu.edu/wsdll2owls](http://www.daml.ri.cmu.edu/wsdll2owls).
- Updated our DAML-S/UDDI matchmaker and DAML-S/UDDI mapping to OWL-S/UDDI matchmaker and OWL-S/UDDI mapping respectively.
- Developed and deployed our new OWL-S/UDDI matchmaker website. It provides a web interface for users to interact with our OWL-S/UDDI matchmaker. Available at [http://www.daml.ri.cmu.edu/matchmaker](http://www.daml.ri.cmu.edu/matchmaker).
- Initial implementation of process model verification tool using SPIN model checking tool.
- Complete implementation of OWL-S 1.1 API.
OWL-S Broker

- Enhancements to our OWL-S Broker that mediates between OWL-S Web services. We enhanced our current algorithm for mapping queries to requests.
Additional Research Activities

- Participated in the development of OWL-S 1.1
- Active participation in:
  - W3C Web Services Architecture WG
  - UDDI Technical committee
  - SWSI/SWSA
- Maintained in continuous operation the DAML-S/UDDI Matchmaker that was deployed in June 2002.
- Publications
  
  www.cs.cmu.edu/~softagents/publications
Plans

Place Tools on [www.semwebcentral.org](http://www.semwebcentral.org)

Timeframe: initial set of tools within next month
Verification achieves two goals when applied to OWL-S PM:

1. **Pragmatic**: SW verification provides a way to check on the validity of the OWL-S PM and Grounding.
   - Clients can verify automatically whether the Process Models and Grounding that they load are correct
   - Modelers can verify whether the OWL-S models that they construct are correct

2. **Theoretical**
   - Verification tools concentrate on control flow, but for semantic languages data flow, and logic consistency is also crucial
Correctness Claims

1. No deadlocks

2. Verification of the correctness of data flow:
   - All inputs receive a value either from outputs or from Grounding
   - Values received from the inputs are consistent with values generated by the outputs
   - Verification of correctness of XSLT transformations from XML types used in WSDL to ontological types used in OWL-S

3. No unreachable processes (every process is achievable by at least one execution trace)

4. Based on Spin verification engine

5. Current Status: complete description of the mapping from OWL-S to Spin modeling language. Ready for implementation