Template-Based Semi-Automatic Web Service Composition

Abdaladhem Albreshne
Computer Science Department,
University of Fribourg, Switzerland
abdaladhem.albreshne@unifr.ch

Supervisor: Jacques Pasquier
Computer Science Department,
University of Fribourg, Switzerland
jacques.pasquier@unifr.ch
Outline

- Problem Statement (with the help of a motivating Scenario)
- State of the Art / Open Challenges
- Proposal
- Conclusion / Research Plan
Problem Statement (1)

- Imagine a smart home with many services
- End users (inexperienced) want to create and manage sophisticated scenarios implying complex services compositions (e.g. energy saving scenario, maximizing security, ...)

Source: http://blog.ipv6.com/?paged=2

European Conference on Web Services (ECOWS'11)
Problem Statement (2)

Requirements

- Services discovery
- Process definition
- Process execution (dynamic adaptation to context)
- User’s involvement
  - 100% manual
  - 100% automatic
  - Semi-automatic (template based)
State of the Art / Open Challenges (1)

User Involvement
- Semantic Description
- GUI

Composition / Execution
- BPEL
- OWL-S

Discovery (Fundability)
- WS Discovery Engine
- Semantic Discovery Engine

Semantic Description
- BPEL4 SWS
- OWL-S

Services Providers
- WS*
- Devices
- Software

European Conference on Web Services (ECOWS'11)
## State of the Art / Open Challenges (2)

<table>
<thead>
<tr>
<th>Requirements</th>
<th>BPEL</th>
<th>OWLS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semantic Service Description</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Dynamic invocation</td>
<td>Yes (services must use the same WSDL)</td>
<td>No</td>
</tr>
<tr>
<td>User Intervention</td>
<td>external partner</td>
<td>external partner</td>
</tr>
<tr>
<td>Events</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Control structures</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Service Discovery</td>
<td>Syntax-based</td>
<td>Semantic-based</td>
</tr>
<tr>
<td>synchronous invocation</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>asynchronous invocation</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>High level of abstraction / generic</td>
<td>No</td>
<td>No</td>
</tr>
</tbody>
</table>
State of the Art / Open Challenges (3)

User Involvement
Automatic generation of user interfaces including presentation and interaction which target unsophisticated end users

Composition
Define a generic process template language with a suitable execution engine to run it dynamically

Discovery (Fundability)
Use semantic features related to each domain to improve service discovery

Semantic Description
Add semantic features related to each domain to give the necessary information about involved services

Services Providers
- WS*
- ? Restful
- OWL-S
- Devices
- Software

European Conference on Web Services (ECOWS'11)
Proposal (1) Generic Software Architecture

European Conference on Web Services (ECOWS'11)
Proposal (2) software architecture components

- Domain Ontology
  - OWL
- Process Template Tool
  - Generic Process Language
  - OWL Library
  - BPEL Extension
- Process Builder
- Client User Interface
  - JSP+HTML
- Process Executor
  - Process Template Repository
  - Process Generator
    - Java
  - Process Execution Engine
    - BPEL or OWL-S
- WS Matching Engine
  - Service Discovery
  - OWL Library

Library (changeable)
Program
Script Process Language
Engine
Tool
Web Services (changeable)
Hardware Equipments (changeable)

European Conference on Web Services (ECOWS'11)
Conclusion / Research Plan (1)

We have explored emerging concepts in order to help end-users define and apply complex scenarios in domains having many services and a rich context:

- BPEL
- OWL-S
- Semantic & Service discovery
- Abstract & Generic process
- User involvement
Conclusion / Research Plan (2)

Several challenges need to be addressed in order to build a flexible and generic web services composition framework:

- definition of a simple, yet flexible, language to describe generic process templates
- The creation of a process generator engine to transform a generic scenario into an executable one
- Enabling the end-user intervention
- Using a powerful semantic discovery mechanism
- the selection (or creation) of an execution engine, which will be able to react to a changing context, represents another non trivial task.