Semantic-based adaptation of services composition

Tan Hanh, Hanh.Tan@imag.fr
Christine Collet, Collet.Christine@imag.fr
Genoveva Vargas-Solar, Genoveva.Vargas@imag.fr

Web Intelligence 19 - 04 - 2007

Services composition

The composition of services specifies an applicative logic. It consists of activities which invoke services that are provided by external providers. It has a precise coordination of services.
Need of adaptation

- System overload, fault tolerance, service unavailability, QoS.
- Heterogeneity of data models, DBMS
- Heterogeneity of networks (UMTS, WLAN, …), variability of network load and topology.

Plan

- Adaptation overview
- Proposal: SEBAS (SEmantic-Based Adaptation for Service composition)
  – Service adaptation ontology
  – Composition adaptation process
  – SEBAS architecture & substitution strategy
- Conclusion and future works
Adaptation

• Modification in structure or behavior of services composition (activities) to respond harmoniously to changes
  – Implicit adaptation: changes in the execution environment
  – Explicit adaptation: change of user needs
• With respect to personalized correctness and QoS criteria

Example: Adaptation of services composition
Support for adaptive composition of services

Composition adaptation problems

- Semantic conflict: service name, operation name, parameter name, service semantics, operation semantics
- Structure conflict:
  - Service2Service, Service2CompositeService, compositeService2compositeService
- Interface conflict:
  - Data type conflict: parameter data type, result data type, complex data type
  - Parameter number conflict
  - Data unit conflict
  - Parameter structure conflict
Plan

• Adaptation overview
• Proposal: SEBAS
  – Service adaptation ontology
  – Composition adaptation process
  – SEBAS architecture & substitution strategy
• Conclusion and future works

Service Adaptation Ontology (SAO)

Knowledge Domain
  - Domain
  - Category
  - Family
  - Service

Service & coordination
  - Service
  - hasMethod
  - hasRelationship
  - hasComposedOf
  - hasOrder
  - hasInterfaceMapping

Service coordination
  - Order
  - InterfaceMapping

Method
  - Method'
  - Functional properties
  - QoS properties
Example: a simple scenario

Dynamic adaptation

2x + 10 = 0

ax + b = 0

ax^2 + bx + c = 0

0x^2 + 2x + 10 = 0

b = b/2

x = b/a

SAO: Knowledge domain

Service taxonomy

Math

HasCategory

Algebra

HasFamily

Arithmetic

HasFamily

FirstDegreeEquation

HasService

SecondDegreeEquation

HasService

Division

HasService

Negation

HasService

Service Structure
Plan

- Adaptation overview
- Proposal: SEBAS
  - Service adaptation ontology
  - Composition adaptation process
  - SEBAS architecture & substitution strategy
- Conclusion and future works

Composition adaptation process

1. Service execution
2. Exception occurs
3. Composition adaptation
   - Equivalent Service
   - System faults: timeout, refused connection, …
   - Application faults: invalid operation, bad parameter types
4. Substitution Decision
   - No; re-execute the failed service
   - Yes
5. Service substitution process
Service substitution process

SEBAS Architecture
Service substitution strategies

Event

Event types representing services and coordination execution

Event of service execution:

\[ E_1: \text{type} = \text{Send_invokation}, \text{serviceid}: \text{string}, \text{timestamp}: \text{date} \]

\[ E_2: \text{type} = \text{Receive_results}, \text{serviceid}: \text{string}, \text{state}: \text{string}, \text{timestamp}: \text{date}, \text{state in} \{\text{succeed, fail, active}\} \]

Event of coordination execution:

\[ E_3: \text{type} = \text{StartProcess}, \text{processId}: \text{string}, \text{timestamp}: \text{date} \]

\[ E_4: \text{type} = \text{EndProcess}, \text{processId}: \text{string}, \text{state}: \text{string}, \text{timestamp}: \text{date}, \text{state in} \{\text{succeed, fail, active}\} \]

EventLog: the set of event instances of any of the previous types produced within the execution of a coordination
QoS metrics

**Service execution time:**  
\[ ej.timestamp - ei.timestamp \]
where  
\( ei.type = \text{Send\_invokation} \) and  
\( ej.type = \text{Receive\_results} \) and  
\( ei.timestamp \leq ej.timestamp \) and  
\( ei.serviceId = ej.serviceId \) and  
\( ej.state = \text{“succeed”} \)

**Service reliability:**
\[ \left\{ e \in \text{EventLog} \land e.type=Send\_invokation \land e.state=\text{succeed} \right\} \]
\[ \left\{ e \in \text{EventLog} \land e.type=Send\_invokation \land e.state = \text{active} \right\} \]

---

**Implementation**

- **Platform:** Web Services
- **Model:** OWL, RDF
- **Language:** Java
- **Tools:** Apache Axis, Racer/Jena2, Log4J, JDOM/MySQL
- **Function:**

<table>
<thead>
<tr>
<th>Service Semantic Model</th>
<th>Event Model</th>
<th>Service</th>
</tr>
</thead>
<tbody>
<tr>
<td>Semantics</td>
<td>Domain</td>
<td>Service</td>
</tr>
<tr>
<td>OWL</td>
<td>DB/XML</td>
<td>Web services</td>
</tr>
<tr>
<td>Racer/Jena2</td>
<td>Log4J</td>
<td>Apache Axis - Tomcat</td>
</tr>
</tbody>
</table>
Summary

- SEMantic-Based Adaptation for Services: The SEBAS approach
  - Service adaptation model:
    - Knowledge domain: Domain, category, family, service and method
    - Semantic relationships: Mapping service2service, service2compositeservice
    - Interface mappings
  - SEBAS adaptation process: Service adaptation process, substitution strategies, SEBAS architecture

Conclusion and future works

- Composition adaptation process without re-booting the execution of the application.
- Complex mappings between services and between their interfaces
- Dynamic expandability
- Choice of service based on semantics and QoS criteria

- Ontology and mappings are manually built
- Centralized adaptation process

- Adaptation triggering problem
- Pattern storage for « automatic » service mapping
- Plugging into SEBAS non-functional properties such as transaction, security and mobility
References

- DAML Services Coalition, DAML-S: Semantic Markup for Web Services
- Grigoris Antoniou, Frank van Harmelen, A Semantic Web Primer, The MIT Press, 2004
- Martin B., Gerti K., Gerhard K., Composite Events for XML, ACM, 2004
- Fabio Casati, Ski Ilnicks, Li Jie Jin, Adaptive and Dynamic Service Composition in eFlow, HP Company, 2000
- Sirish Chandrasekaran, Samuel Madden, Mihut Ionescu, Ninja Paths: An architecture for composing services over wide area networks, CS262 class project
- Boualem Benatallah, Marlon Dumas, Zakaria Maamar, Definition and execution of composite web services: The SELF-SELRV project, IEEE, 2002