Stratified Composition of Web Services

Riina Maigre

Institute of Cybernetics at TUT

Theory Days at Kääriku 2009

---

Outline

Motivation
Large information systems with service oriented architecture

Knowledge architecture of the composition tool
User knowledge level
Logical level
Service implementation level
Motivation

- Goal of this work is to overcome the complexity of service composition on very large sets of atomic services.
- To achieve this goal we propose an automatic service composition methodology with three distinguished knowledge levels:
  - user knowledge level
  - logical level
  - implementation level
Example application domain

- Estonian e-government information system has a service-oriented architecture.
  - Services accessible through this information system are described in WSDL.
  - More than thousands of services are provided by information systems of different institutions.
- Secure access is provided (over X-Road) to nearly all public (i.e. governmental) but also some private databases.
- All Estonian residents having the national ID card can access these services through X-Road within their limits of authority.
- In this work we refer to the whole system as X-Road.
X-Road
Example model

- We have performed experiments on a part of X-road information system.
- Service model containing about 300 atomic services and about 600 references to semantic resources has been created.
- Totally more then thousand of atomic services are available.
Model of the application domain
Querying X-Road

- Only predefined queries can be done through X-Road portals.
- Combined queries between information systems of different institutions are not possible.
- Possible semantic relationships between queries are ignored.
Example: querying person’s contact addresses

(Without complex queries:)

From the 1th DB: NationalIdCode → Address
From the 2nd DB: NationalIdCode → Address
...
From the n-th DB: NationalIdCode → Address
Example complex query

NationalIdCode →
AddressString, EstonianAddressString,
OwnerAddressString, ResidencyAddress,
ResponsibleUserAddress
Service composition tool

- We have created a tool for service composition that supports the stratified composition methodology and hides complexity of composition from the end user.
- Tool is created in the software development environment CoCoViLa that supports automatic synthesis of programs and generates Java code from visual and textual specifications.
Knowledge architecture of the composition tool

Knowledge system – module of knowledge architecture.
- knowledge language
- knowledge handling mechanism
- method for associating meanings to knowledge objects

\[
\begin{array}{c|c}
S & M \\
\hline
\end{array}
\]

S – knowledge objects (notations)
M – set of meanings (denotations)
Connections of knowledge systems

Knowledge systems can be composed into larger knowledge architecture by using:

- Hierarchical connections (a)
- Semantic connections (b)
- Operational connections
Knowledge levels

1. User knowledge level
2. Logical level
3. Service implementation level
Service composition tool

visual KS
- schemas
- unfolded text
- logical formulas
- realizations of formulas

textual KS
- textual specs

user knowledge level

logical level
(intuitionistic logic)

service algorithm \cup grounding

Java program

Java program = service generator

service description

service implementation level
1. User knowledge level

- visual KS
  - schemas
- textual KS
  - textual specs
- unfolded text
1. **User knowledge level**

Visual knowledge representation.
1. User knowledge level

Specifying a complex service on visual representation.
1. User knowledge level
Visual and textual representation of knowledge.
2. Logical level

Formal representation of knowledge and automatic composition of new services.

- Knowledge about atomic services, goal => algorithm of expected service
- Structural synthesis of programs is used to synthesize a structure of a new complex service
3. Service implementation level

service algorithm \cup \text{grounding}

Java program

Java program = service generator

service description
3. Service implementation level

Synthesizing a structure of the complex service.
3. Service implementation level

Grounding of services - calling services and performing actual computations.
3. Service implementation level

Generating BPEL from Java code.
Conclusion

- We have described the architecture of the complex knowledge-based tool with stratified knowledge levels.
- The main features of the presented architecture are
  - user friendly upper level
  - precisely defined mappings between levels
- We hope to make the tool usable first of all to the developers
Thank you for listening!

Questions?

Supporters:
Estonian Science Foundation

Tiger University