Context-aware Computing: Programming Models

Hong-Linh Truong and Schahram Dustdar

Distributed Systems Group
Vienna University of Technology
https://www.infosys.tuwien.ac.at/teaching/courses/caws/
Overview

- Programming models
- Programming models in context-aware computing
  - Direct message-based context programming model
  - Common space-based context programming model
  - Message passing context programming model
- Discussion
Programming models

- Examples of programming models
  - multi-threaded
  - message passing
  - tuple space
  - remote procedure calls
- A programming model can focus on
  - exchanging data among application processes
  - expressing algorithmic parts in applications
- Low level versus high level programming models
Programming models in context-aware computing

- Programming models in context-aware computing
  - A set of (standard) technologies to exchange context information among processes and to specify context-aware aspects in context-aware applications
- Strongly related to
  - Context representation, context storage
  - Software modeling
  - Programming languages
  - Underlying systems
Programming models in context-aware computing

- Why programming models are important?
  - Simplify the programming effort and improve interoperability
  - Deal with complex and heterogeneous systems

- So far, in context-aware computing, there is no standard/widely-accepted programming models

- Programming models/software engineering techniques are dependent on specific toolkits and middleware
Programming models in context-aware computing

- Recall our first lecture on context-aware Web services
  - Repeated activities
    - Design context models
    - Design context supporting components
    - Implement models and components
    - Develop applications using the models and components
  - Very little reuse, except some ontologies and common storages
Current software engineering method for context-aware computing

- **A typical process**

  ![Diagram showing the process of Requirement Analysis, Design, and Development with additional notes on SE in CAW, List of requirements, Context information representation, storages, exchange messages, conceptual architecture, and Specific, new, yet-another XML/OWL, middleware, etc.]

We should stop this way! Think more on programming models to support the interoperability and reusability!
Important characteristics on selecting programming models

- Centralized or distributed applications
  - E.g., context-awareness in a mobile phone application versus context-awareness in a set of distributed Web services
- Single organizational versus multiple organizational environments
  - E.g., distributed processes in a department versus distributed processes in the Internet
- Message representations
  - E.g., simple tuple versus RDF context information
Some notes

- Technologies for modeling context information
  - It is about data structure, not a programming model
  - Many language extensions allows specifying context information
    - If they are used together with high-level programming models, it could constitute a context programming model
Example of modeling context information: Context UML

- Modeling context and some context supporting components using UML

Example of modeling context information: Context UML

- In a context-aware Web service, context information are actually required parameters.

Example of language extensions: COP

- Using „layers“ to describe behaviour variables (in the form of classes and methods) which will be activated based on context

```java
class Person {
    private String name, address;
    private Employer employer;

    Person(String newName, String newAddress, Employer newEmployer) {
        this.name = newName;
        this.employer = newEmployer;
        this.address = newAddress;
    }

    String toString() { return "Name: " + name; }
    layer Address {
        String toString() { return process("Address: " + address); }
    }
    layer Employment {
        String toString() { return process("Employer: " + employer); }
    }
}

class Employer {
    private String name, address;

    Employer(String newName, String newAddress) {
        this.name = newName;
        this.employer = newEmployer;
    }

    String toString() { return "Name: " + name; }
    layer Address {
        String toString() { return process("Address: " + address); }
    }
    layer Employment {
        String toString() { return process("Employer: " + employer); }
    }
}
```

Direct message-based context programming model

- Context data is coupled via message structure
- Communication is point-to-point
Common space-based context programming model (1)

- Context data is coupled via message structure, shared objects, and APIs
- Communication is based get()/set()
Common space-based context programming model (2)

- Shared data versus shared objects
  - E.g., XML data versus Java (remote)object

- Well-known tuple space models
  - put() and get() operations
  - Different context representations and matching functions (keyword, fuzzy, etc.)
  - Distributed or centralized spaces

- Various middleware used centralized storages
  - Similar to put() and get() are store() and search()
  - With different query languages (e.g., XML, SQL, pattern matching, fuzzy logic)
Common space-based example: GAIA

- Middleware built a top CORBA
- Common space is established by the Space Repository Service and the Context File System

- But the programming model is like tuple-space and MCV (model-control-view)

---

Common space-based example: Reconfigurable Context-Sensitive Middleware

- Use the CORBA model
  - Same context sensitive objects are referred among processes
  - An ADC (Adaptive Object Container) acts as „client/server“ to manage context sensitive objects

Common space-based example: Java Context Awareness Framework

- Using RMI and Java objects
  - Context is a property of an real world entity

```java
public class Person {
    //normal properties/behaviour of a person
    //context data
    //context APIs
}
```

- An object has a context which has a set of context items
- Context objects are managed by a context service (RMI server), context monitors and context actutors (RMI client) manipulate context objects

Message passing context programming model (1)

Message passing context programming model (2)

Message passing context programming model

- **WS-Context**
  - A context provides extra information about an activity which represents a set of interrelated interactions
  - Context is assumed to associated with a message
  - Message passing style
    - Asynchronous communication
  - APIs are specified

http://docs.oasis-open.org/ws-caf/ws-context/v1.0/wsctx.html
### Context Programming Models Used in Existing Systems

<table>
<thead>
<tr>
<th>Systems</th>
<th>Types of Coupling</th>
<th>Privacy</th>
<th>Multi-organization</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>structure</td>
<td>data</td>
<td>common</td>
</tr>
<tr>
<td>MDE approach</td>
<td>+</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>inContext [41]</td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>WS-Context [12]</td>
<td></td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>USS [40]</td>
<td></td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>CATIS [37]</td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>CASD [21]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>ESCAPE [44]</td>
<td></td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>CA-SOA [20]</td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Akogrimo [8]</td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ConServ [26]</td>
<td>+</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ERMHAN [36]</td>
<td></td>
<td></td>
<td>+</td>
</tr>
<tr>
<td>Keild et al. [29]</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Discussion on Programming Models

- Strengths and weakness for context-aware Web services

<table>
<thead>
<tr>
<th>Category</th>
<th>Types of Coupling</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Structure</td>
</tr>
<tr>
<td>Development flexibility</td>
<td>high</td>
</tr>
<tr>
<td>Integration complexity</td>
<td>low</td>
</tr>
<tr>
<td>Security enforcement effort</td>
<td>low</td>
</tr>
<tr>
<td>Privacy enforcement effort</td>
<td>low</td>
</tr>
<tr>
<td>Interoperability degree</td>
<td>low</td>
</tr>
</tbody>
</table>
inContext Case Study: Requirements for Context Coupling

- Need both design-time and runtime context coupling techniques
  - Collaboration context across user boundaries
inContext Case study: Design context model

- Model individual context, team context and activity context using RDF
- Support flexible and extensive models by including domain-specific context models and reusing common RDF context models
Supporting distributed context management

Supporting the use of URI to retrieve context information
- ActivityURI and UserURI
- Embedding URIs specifying context information into SOAP message header
- No application-specific source code
- Extensible mechanism

Supporting RDF/XML context Information
- XSPARQL for querying context data and transforming RDF to XML
inContext Case Study: Runtime Context Coupling Techniques
<xml version="1.0" encoding="UTF-8"?>
<soapenv:Envelope

...  
<soapenv:Header>
  <ns1:ctxtunnelling
       soapenv:actor="http://schemas.xmlsoap.org/soap/actor/next"
       soapenv:mustUnderstand="0" xmlns:ns1="www.in-context.eu">
    <ns1:Activity>
      http://www.in-context.eu/pcsa#act1
    </ns1:Activity>
    <ns1:User>
      http://www.in-context.eu/pcsa#Rossi.E54
    </ns1:User>
  </ns1:ctxtunnelling>
</soapenv:Header>
<soapenv:Body>
  ...
</soapenv:Body>
</soapenv:Envelope>
inContext case study: Message passing and common space-based programming models

- Message passing model
  - SOAP Header extensions: carry over User/Activity ID in service calls, enables tunnelling, monitoring, mining
  - Prototypes for AXIS1, AXIS2 and .NET
  - Context aware services can exploit it, but no obligation → no specific change for services
  - Enable context ranking and constraints

- Common space-based context programming model based on the Context Storage
  - getContext(XML, XSPARQL)
  - setContext(XML, SPARQL)
Discussion on Programming Models

- Which ways are the best for implicit/explicit context exchange among multi-organizational processes in the Internet?
  - How to deal with on-the-fly service interaction?
- Is it possible to have a common/standard APIs for put(), get(), subscribe() of context information?
  - How to deal with complex representations of context information
  - Under which environments, such APIs are feasible?
- Is it possible to incorporate context model specifications into existing programming languages, e.g., using annotation or meta-data?
  - How to deal with context exchange among processes
Discussion on Programming Models

- **Is it feasible to have a global common space-based in the cloud?**
  - Would be interesting for mobile, context-aware applications!
  - Global common spaces atop cloud databases (e.g., Amazon SimpleDB [http://aws.amazon.com/simpledb](http://aws.amazon.com/simpledb))

- **Will the combination between language extensions and existing programming models provide a powerful technique for programming context-aware applications in large-scale systems?**
Thanks for your attention!

For further information, pls. check the course link: https://www.infosys.tuwien.ac.at/teaching/courses/caws/