COSMOS: COntext entitieS coMpositioN and Sharing

D. Conan and S. Leriche

Kick-off BROCCOLI
February 2008
Motivations and objectives

- Ubiquitous computing \(\implies\) High number of heterogeneous devices, huge amount of context data

- Context management [Coutaz et al., 2005] to identify/detect the situations of adaptations

- Process context data in a usable, scalable, and efficient manner
  - Usable: Compose, deploy, configure, and reconfigure (without programming)
  - Scalable:
    - No performance degradation when multiple clients’ observations
    - Separation of context collections according to context sources
  - Efficient: Control resources consumption of context management tasks (memory and activities)
Outline

1. Functionalities of a context manager .................................................. 4
2. COSMOS concepts: Context node, context report, and context policy .......... 5
3. Towards a case study: Mobile commerce ............................................. 9
4. Status of the COSMOS framework ..................................................... 11
5. Ongoing and future work specific to COSMSOS ................................... 12
6. Tentative agenda for the forthcoming months specific to COSMOS ............ 16
1 Functionalities of a context manager

- Separation of concerns
  - **Collection** = different context sources
  - **Interpretation** = different inference engines
  - **Adaptation** = several “client” applications with different situation identifications

- Compose context frameworks in a component-oriented architecture

<table>
<thead>
<tr>
<th>Context adaptation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Context processing</td>
</tr>
<tr>
<td>Context collection</td>
</tr>
<tr>
<td>System resources</td>
</tr>
<tr>
<td>User profiles</td>
</tr>
<tr>
<td>Sensors</td>
</tr>
<tr>
<td>Remote data</td>
</tr>
</tbody>
</table>

- Situations identification, Data interpretation, Data collection
2 COSMOS concepts: Context node, context report, and context policy

- **Context policy** = Abstract context information provided to the user/application
  - A hierarchy of context nodes
  - With sharing of context nodes between context policies

- **Context node** = Context information modelled by a component
  - Basic structuring “component” of COSMOS

- **Context report** = Extensible message structure
  - \([O..n]\) chunks: Identifier + values
  - \([O..m]\) sub-messages: Encapsulation
2.1 Software architecture approach: Context policies

- Apply architecture-based principles to design context policies
  - Software architecture for system instrumentation, deployment, configuration
    - “A software architecture of a program or computing system is the structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships among them.” [Bass et al., 1998]

- Use an architecture description language [Medvidovic and Taylor, 2000] to describe the context policy
  - Compose rather than program...
    - Reify a context policy as a tree of components with sharing
    - Architectural patterns for context node composition and sharing
  - ...during design, implementation, and execution

- Use a component-based message-oriented middleware [Leclercq et al., 2005]
  - Fine-grained management of context activities and context reports
2.2 Software component approach: Context node

- Apply component-based principles to design context nodes
  - Units for system modularity, reconfiguration, fault isolation
    - “A component is a unit of composition with contractually specified interfaces and context dependencies only. A software component can be deployed independently and is subject to composition by third parties.” [Szyperski, 2002]

- Compose rather than program...
  - When programming, apply attribute/annotation-oriented programming
  - ...during design, implementation, and execution
2.3 Context node parametrisation

■ Properties of a context node

♦ Controls propagation of information
  ► Can observe (down to the leaves) and/or notify (up to the root)
    ★ Attributes *Observe* and *Notify*
  ► May block the context flow (down or up) or not
    ★ Attributes ObserveThrough and NotifyThrough
      + attributes ObserveOnlyOnce and NotifyOnlyOnce

♦ Controls the propagation mode
  ► Is passive or active
    ★ Attributes isActiveObserver and isActiveNotifier
      + attributes period*

♦ Has a name to be registered into a registry and searched for configuration
  ► Attribute nodeName
3 Towards a case study: Mobile commerce

- **Family shopping** in a mall with all the members of the family equipped with a mobile device
  - Share information
  - Consult product prices
  - Download discount tickets
  - Be notified of advertisements
  - Access additional information and comments about a product
  - Find the location of a product or a shop in the mall
3.1 Context management with COSMOS

- Forest of context policies

- RFID tag observation enabled
  - Bluetooth notification enabled
    - Bluetooth detectable
      - Bluetooth availability
        - User preference’s change detector
          - Battery lifespan
            - Connectivity detector
              - Average link quality
                - WiFi link quality
                  - WiFi link variable
                    - Is bit rate variable?
                      - WiFi manager
                        - WiFi download enabled
                          - Decision stabilisation
                            - WiFi adjusted bit rate
                              - Group Membership Service (with location)
                                - Disconnection detector
                                  - Failure detector (with location)
                                    - Heartbeat counters
                                      - Location
                                        - Periodic Heartbeat Sender/Receiver
                                          - Network call

- RFID tag readable
  - Bluetooth detectable
    - Battery
      - User preference’s charge state
        - Battery time left
          - WiFi link quality
            - Bluetooth link quality
              - User preference’s charge state
                - Battery charge state
                  - Battery manager
                    - RFID tag manager
                      - Tag call
                        - Registry call
                          - System call
                            - System call
                              - Network call

D. Conan and S. Leriche — February 2008 — Kick-off BROCCOLI
3.1 Context management with COSMOS

- Configuring context nodes: non-/blocking, active/passive
3.1 Context management with COSMOS

- Mapping context node activities to threads
3.1 Context management with COSMOS

Mapping context nodes to message managers
4 Status of the COSMOS framework

- **Publications**: [Conan et al., 2007, Conan et al., 2008]
- **Web site**: http://picoforge.int-evry.fr/projects/cosmos
- **Forge**: https://picoforge.int-evry.fr, guest/guest
  
  ♦ Project currently unstable, under a refactoring and mavenisation process

  ▶ From Ant to Maven
  ★ Decomposition into cosmoscore, cosmoslib and cosmossaje
  ★ No deployment web site for the moment

  ▶ From Fraulet-Xdoclet to Fraclet-Java and Dream-Annotation
  ★ Dream-Annotation depends on Fraclet-annotation: No @Legacy
  ★ Perhaps conflicts between Fraclet-Java and Dream-Annotation

  ▶ Unitary tests to replace cosmossaje tests
  ★ Especially, Dream activity management

  ▶ Design pattern “Singleton” using dynamic sharing of components
  ★ Two many layers of composition $\implies$ up to now, using “Singleton” objects
  ★ See email on the Fractal mailing list of Romain Rouvoy dated ...
5 Ongoing and future work specific to COSMSOS

5.1 Domain specific language for context composition..............................13
5.2 Generic context operators.....................................................14
5.3 Deployment and distribution of context information............................15
5.1 Domain specific language for context composition

■ DSL [Mernik et al., 2005] for writing context policies
  ♦ Generate context policies written in Fractal ADL
    ▶ First step towards analysis, verification, optimisation, transformation, etc.
      ★ E.g., merge context policies, deadlock prevention

# Functional part: Compose context nodes

sensor RfidTagMgr = RFIDTagRM[BO,A0,AN];
sensor PrefMgr = UserPreferenceRM;
processor AdvertisementChange = ChangeDetectorCO[BO,A0,AN]
  (PrefMgr.extract("advertisement-preference-chunk"));
processor TagReadable = TagReadableCO[BO,A0,AN]
  (RfidTagMgr.extract("tag-presence-chunk"),AdvertisementChange);
processor TagObservationEnabled = IsEnabledCO(TagReadable);

# Extra-functional part: Threads and memory consumption

task RFIDTask = AdvertisementChange,TagReadable,RfidTagMgr;
thread RFIDThread = RFIDTasks[5000];
reporting UserPrefReport = AdvertisementChange/descendant-or-self::*;
reporting RFIDReport = TagObservationEnabled,TagReadable,RfidTagMgr;
5.2 Generic context operators

- Using Fractal-Generics

- First ideas
  - processor Foo1 = \texttt{add(BarInt1, BarInt2, 1)}
  - processor Foo2 = \texttt{add(BarInt1)}
  - processor Foo3 = \texttt{and(BarBool)}
  - processor Foo4 = \texttt{myOperator(Bar)}

- Using a generic context operator
  - Argument = Method of the operator (e.g., \texttt{cosmos.op.add} or \texttt{myapp.myOperator})
  - Undefined number of child context nodes

- Chunk types automatically deduced
  - Either a Java primitive type (e.g., \texttt{cosmos.NumberChunk} containing a \texttt{j.l.Number})
  - Or \texttt{dream.msg.AbstractChunk} returned by application-specific operators (e.g., \texttt{myapp.BarChunk} containing a \texttt{myapp.Bar})
5.3 Deployment and distribution of context information

- Deployment of COSMOS with FDF [Flissi et al., 2008]
  - Description of Dream software
  - Description of COSMOS software

- Distribution of context information with Dream [Leclercq et al., 2005]
  - Study of the Dream communication components library

- COSMOS as a network-accessible service
  - Dynamic instanciation/removal of new context policies
  - Dynamic merging of context policies
  - COSMOS = a distributed service
6 Tentative agenda for the forthcoming months specific to COSMOS

- End of March: Stabilisation of cosmoscore
- End of April: Generic context operators in cosmoslib
- End of May: First proposition of the DSL for context composition
References


