A Context Query Language for Pervasive Computing Environments

M. Valla, C. Fra’ - Telecom Italia
R. Reichle, M. Wagner,
M. U. Khan, K. Geihs – Univ. of Kassel
N. Paspallis, G. Papadopoulos – Univ. of Cyprus

Work supported by European Union FP6, MUSIC project - contract number 35166
IEEE CoMoRea ‘08
A Context Query Language for Pervasive Computing Environments

Outline

- The framework: the MUSIC Project
- Requirements for the MUSIC CQL
- The MUSIC CQL design
- Application examples
- Evaluation
- Current and Future work
The framework: the MUSIC Project

- MUSIC project goal:
  To provide Open Source and extensible middleware & tools for easy development of context-aware & adaptive innovative mobile applications

- Application areas: mobile services for commuters & tourists (Paris metro), ad-hoc social communities, emergency handling, interaction with metro infrastructure

- Partners: device manufacturers, mobile service & middleware developers, Telco operators, final users (RATP Paris), universities (UNIK, UiO, UCY, KuL)

- Project website: http://www.ist-music.eu/

- The MUSIC middleware includes a Context Middleware to support efficient context distribution and access by applications
- Application adaptation is performed using context information
Context in MUSIC

Context Providers:
- Temperature Sensor
- GPS Sensor
- User occupation Sensor

Context Consumers:
- MUSIC Adaptation Logic
- MUSIC Application

Context space:
- Context Modeling
  - XML, key-value pairs, etc

Context Distribution:
- Interoperability, Semantics, etc

Context Query Language:

Synchronous or asynchronous communication of context information
Bidirectional synchronization of context information
Context information is encompassed in ContextElements describing a certain ContextScope (group of ContextValues) for an Entity, with associated Metadata.

- ContextValues and Metadata are structured corresponding to a representation defined in the ontology.

- **NOTE:** it’s an hybrid context model as ontology is not needed at transport layer or on devices (context is associated to scopes mapped to the ontology).
MUSIC CQL: Requirements

- Platform-independent modelling of context access
  - Multiple data representations
  - Allow tools to enable Model Driven Development of Context Reasoners, Applications

- Address the specific characteristics of context information
  - Static vs. dynamic, temporal nature, impreciseness, ambiguity, incompleteness, uncertainty, ...
  - Aggregate and reason about more elementary context elements

- Deal with the challenges of pervasive computing environments
  - Autonomy, Distribution, Mobility, Heterogeneity
Specific MUSIC CQL requirements

- Retrieve a set of context elements from different devices with a single query
  - No need to address specific context sources
- Synchronous and subscription-based context queries
- Filters and conditions on values and meta-data
- Logical operators to combine conditions and aggregation functions
- Aggregation functions (e.g. average, max, min)
- Access both current and past context information
- Complex but compact queries incorporating ontology reasoning
The MUSIC Context Query Language

- Aligned with the concepts of the MUSIC Context Model
- Simple XML-based query language
- Main tags comprise
  - Involved Entities
  - Involved context Scopes
  - Action to be performed: SELECT, (UN)SUBSCRIBE with optional aggregation function for query post-processing
  - Conditions with condition types (ONCLOCK, ONCHANGE, ONVALUE)
MUSIC Context Query Language: example

“Send me civil address of Mary each time her context shows she is in Torino”

```xml
<ctxQuery resultName="addressOfMaryInItaly">
  <entity ontConcept="prefix:music:username">user|Mary</entity>
  <scope ontConcept="prefix:music:CivilAddress"
         ontRep="prefix:music:DefaultAddressRep">civilAddress</scope>
  <action type="SUBSCRIBE"/>
  <conds>
    <cond type="ONVALUE">
      <constraint par="civilAddress.city" op="EQ" value="Torino"/>
    </cond>
  </conds>
</ctxQuery>
```
MUSIC CQL syntax features

- We can query for context information
  - corresponding to a certain concept
  - characterizing a certain entity and
  - having a certain representation

- Inter-Representation-Operations (IROs) are automatically called for automatic conversion between different representations

- Support for complex filters and conditions
  - Ordinary constraints are specified through the involved parameter, operator, value (and delta)
  - Support for the integration of (aggregation) functions

- Context information from a history (no more valid) or cached (still valid) information can be accessed → timerange tag
Reasoning constraints

Reasoning constraints incorporate relations between context entities in terms of ontology reasoning

“Find users that are currently in group with user Mary”

```xml
<ctxQuery [...]>
<entity [...]><user|*</entity>
<action type="SELECT"/>
<conds>
  <cond type="ONVALUE">
    <reasconstraint
      relation="prefix:music:isInGroupWith"
      toEntities="user|Mary"/>
  </cond>
[...]
</ctxQuery>
```
Applications of CQL

- **Mobile Advertising scenarios**
  - “Select usernames where bt.btList contains 000123456789"
  - “Select usernames in [user1...userN] where userPosition (position.lat, position.lon) < position.range AND socialStatus = with_friends"
  - “Select usernames that exit an area after having spend there >X minutes"
  - or more general:
    - “Select users that enter/exit/stay for N min. in a specific target context X [and then enter context Y]”

- **eTourism scenarios**
  - Points Of Interest (POI) group recommendations:
    - “Find all users that are currently in group with UserXYZ”

- **Ad-hoc communities**
  - “Select users on this carriage with batteryLevel > 6 AND WiFi connection AND gamer_profile”

- **Context-triggered messaging**
  - “Notify me when UserXYZ is at home AND socialStatus = with_family”

- **MUSIC TARGET**: develop context reasoners and context-aware applications by means of XML context-queries, without worrying about underlying context management infrastructure
Evaluation of related works

- **Context Modeling Language (CML)**: query through language/platform specific API; maps queries to SQL on relational db, multiple joins for reasoning; ontology support seems limited; no multiple representations; requires different context models targeted to specific application domains
- **MoGATU CQP** (Collaborative Query Processing protocol): limited filters & aggr. functions combination support; does not support heterogeneity
- **CARE middleware**: support for heterogeneous representations and easy integration of filtering functions appears not sufficiently addressed
- **SPICE UE project**: uses SPARQL; not easy for querying complex data constructs; limited support for filter and aggr. functions; queries easily become quite long and complex
- **MobiLife UE project**: uses custom XML-based CQL; lack of aggr. functions; limited semantic support. Most important: application must know beforehand the provider of context information
Design Evaluation

- MUSIC CQL achieves the following improvements:
  - definition of complex filtering/conditions mechanisms
  - elaborate aggregation and pre-processing functions
  - ontology integration
  - heterogeneous representations of context information (thanks to IRO and ontology of representations)
  - support for request/subscription to current and historical context data
  - inclusion of arbitrary meta-data is allowed, available for filtering
  - highly dynamic environments are supported: context sources may be locally deployed or remotely accessible, and context sources can appear or disappear
Current and Future work

- **Implementation and Validation**
  - Step-wise approach to CQL features implementation
    - Language structure and grammar have been already finalized
    - Language basics first implementation is already available (synch + asynch queries)
  - MUSIC context management system and CQL to be validated through trials of mobile context-aware services with real users
    - MUSIC pilots: eTourism, commuter assistance, in-station emergency scenarios
    - Telecom Italia’s trials on eTourism and Mobile Advertising applications will also use CQL

- **Evolution**
  - Integrate an ontology reasoner for reasoning constraints
    - to automatically resolve relations between entities like isInGroupWith as results from other relations: isNearBy and ...
  - Mixed-mode query processing: full-featured XML queries on context brokering servers are translated in simpler context requests to devices that cannot support CQL
Questions & Answers

...Thanks!!!