eXtreme Design with Content Ontology Design Patterns

Valentina Presutti and Eva Blomqvist
Lecture 5 @ Corso di Dottorato 2011
Dipartimento di Scienze dell’Informazione
Bologna, Italy
Method and tool support

- **eXtreme Design (XD)**
  - a method for developing ontologies with Content Patterns

- **XD tool**
  - a tool that supports XD method
  - released as both an Eclipse plugin and a NeOn Toolkit plugin
  - We will use it with the NeOn toolkit
  [http://neon-toolkit.org/wiki/Download/2.3.2](http://neon-toolkit.org/wiki/Download/2.3.2)
eXtreme Design

- XD is a general approach to ontology engineering
- Local Use Case (LUC)
  - represents the current modeling issue
- Generic Use Case (GUC)
  - represents a generic problem that is solved by the associated ODP
- GUC and LUC are represented in a compatible comparable way
- What ODP to reuse?
  - The one where LUC matches GUC
  - Note: GUC are often more abstract than LUC
XD with Content ODPs

• GUC and LUC are expressed in the form of Competency Questions (or sentences)
  – GUC: Who are the participant to a certain event?
  – LUC: Who are the trainees of a certain tutorial?
Matching GUC and LUC

• A LUC can be completely or partly described exactly in terms of the GUC
  – Guc: Performing in a concert
  – Luc: John Coltrane performed in a concert in Japan in 1966

• A LUC is a more specific case of the GUC
  – Guc: Participating in an event
  – Luc: Mary attended a scientific conference

• A LUC can be described in terms of part of the GUC
  – Guc: Participating in an event held in a certain place at a certain time
  – Luc: Mary attended a conference in Italy
Where does the name “XD” come form?

- Inspired by XP 😊 with focus on design
- An agile methodology for web ontology design
- It is part of the NeOn methodology
**XD principles**

- Customer involvement and feedback
- Customer stories to derive CQs and contextual statements (+ reasoning requirements)
- CP reuse and modular design (ontology networks)
- Collaboration and integration
- Task-oriented design
- Test-driven design
- Pair design
XD iteration

Tool support: matching and selection of patterns are perceived to be the most difficult tasks
XD – Method steps

• Task 1 – Familiarize with the domain and task
  – Essential to understand the context and task of the ontology
  – Customer involvement – domain experts
  – Setup the project environment (collaboration support)

• Task 2 – Collect requirements stories
  – Example scenarios (cf. the story of the exercise)
  – Should be short and “modular”

• Task 3 – Select a story (each design pair!)
  – Divide&Conquer strategy in the large – method is iterative and incremental.
  – Stories are associated with priority values
  – Based also on design pair competencies
XD – Method steps

• Task 4 – Transform the story into CQs
  – Derive requirements from the text
    • Instance free sentences...
    • ...then CQs
    • Check with customer representative!
  – Should correspond to actual queries that the user/system need to pose
  – Other requirements? Contextual statements? Reasoning requirements?

• Task 5 – Select a CQ (each pair iterates)
  – ...or coherent set of CQs
  – Together with associated contextual statements & reasoning requirements
  – Divide&Conquer strategy in the small – pair iterates over CQs and creates module(s).
XD – Method steps

- Task 6 – Match CQs to GUCs of Content ODPs
  - How? Either only intellectually or with some tool support e.g. XD Selector
  - Can I describe my local problem in terms of the general problem of the ODP?
  - Does the ODP solve the same “design issue”?  
  - Partial match – Is it worth the overhead?
  - Several ODPs needed – Composition of ODPs
  - In case there is no matching ODP – consider to create one!
XD – Method steps

• Task 7 – Select the Content ODPs to reuse
  – May exist several options
  – “Rule of thumb” – use the most (domain) specific one applicable

• Task 8 – Reuse and integrate selected Content ODPs
  – Specialize
  – Import
  – Extend
  – Integrate (compose)
Task 9. Test and Fix – Unit tests

1. SPARQL queries

   Assume the following CQ:
   What role did a certain person play in the production of a certain play during a certain time period?

   ```sparql
   WHERE {
     ?rolePlaying a :PlayingSituation .
     ?production :productionOfPlay ?play
     ?timeInterval :hasStartDate ?startTime .
     ?timeInterval :hasEndDate ?endTime .
   }
   ```

2. Producing inferences

3. “Stress testing”
XD – Method steps

• Task 10 – Release module
  – Make sure the module is commented and ready
  – Post the module so that it is accessible by the other pairs
  – Post any new reusable Content ODP developed
  – Taken over by integration pair?

• Task 11 – Integrate, test and fix (by integration pair – or by development pair)
  – Integrate with overall ontology so far
    • Alignment may be needed
    • Refactoring may be needed
  – Run all unit tests based on all included requirements

• Task 12 – Release new version of the ontology (by integration pair – or by development pair)
  – Distribute
  – Generate documentation
  – Check customer satisfaction
Summary

• XD is an agile method – start building small modules that solve a few requirements, then add more

• Testing is essential

• Collaboration is essential

• Problems are resolved in the integration phase – alignments or refactoring?

• You are about to experience the method!!