A Topic Map Templates based Prototype for Software Development Support

M. Ueberall, O. Drobnik

Telematics Group, Institute of Computer Science
J. W. Goethe-University, Frankfurt/Main, Germany

2007/10/11
Motivation and Objective

Problems in Software Development:

- Participants (stakeholders, architects/designers, programmers) have different objectives/points of views as well as different domain knowledge → consistent conceptualisations
- Any loss of information is to be considered fatal → traceability
- semantics-preserving phase transitions

Our approach is based on

- use of light-weight representations (Topic Maps)
- development process support by means of templates
User Support (GUI)

- **Roles:**
  - mainly used for access control (e.g., certain domain-/phase-specific concepts can only be annotated)

- **Views:**
  - based on roles; used to filter information and present results according to user’s domain knowledge

- **Queries:**
  - filter/constraint mechanism; both pre-defined (for views) and user-adjustable
Prototype

Who is it supposed to look like?

- **Eclipse Plugin**

- **Based on the Eclipse GMF (Graphical Modeling Framework) Mindmap Example**

  - Essentially, we’re interested in “a kind of formalised mindmap using domain-specific visualisations”
Hierarchical Structuring

- A *faceted classification system* allows the assignment of multiple classifications to an object.
- This enables the classifications to be ordered in multiple ways, rather than in a single, pre-determined, taxonomic order.
- The most prominent use of faceted classification is in faceted navigation systems that enable a user to navigate information hierarchically, going from a category to its sub-categories, but choosing the order in which the categories are presented.
Usage Restrictions for Templates
Constraints Example

(a) cardinality constraint for “Age”
(b) context-sensitive labeling
(c) occurrence-type cannot be used as template attribute
Versioning

Metadata for Units of Information, LTM notation

// <occurrence | association> ~object-ref

{object-ref, dc:creation-date, [[2007-06-04T2359:59+01:00]]}
{object-ref, dc:version, [[version-id]]}
{object-ref, dc:author, [[user-id]]}

{object-ref, skos:changeNote, [[description]]}
...

is-replaced-by(object-ref\(_1\) :old-obj, object-ref\(_2\) :new-obj)
is-deprecated(object-ref\(_3\) :obj)
is-deleted(object-ref\(_4\) :obj)
### Versioning
**Granularity of Objects**

<table>
<thead>
<tr>
<th>Name</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>UseCase</td>
<td>FlightBooking</td>
</tr>
</tbody>
</table>
| Description | Statement A  
Statement B  
Statement C |
| Precondition | ... |
| Action | ... |
| Postcondition | ... |

<table>
<thead>
<tr>
<th>Name</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>UseCase</td>
<td>FlightBooking</td>
</tr>
</tbody>
</table>
| Description | Statement A  
Description1 | Statement A  
Description2 | Statement B  
Description3 | Statement C |
| Precondition | ... |
| Action | ... |
| Postcondition | ... |

\[
is\text{-}\textit{replaced-by}(\textit{desc-ref}_1 : \text{old-obj}, \textit{desc-ref}_2 : \text{new-obj}) \\
is\text{-}\textit{part-of}(\textit{desc-ref}_2 : \text{whole}, \textit{description}_1 : \text{new-obj})\]
Versioning

Importance of the Directed Acyclic Graph structure

(Underlying) Application Logics

- **Dynamically loadable Java Classes ("modules"):**
  - constraints/templates can be added as topic map objects; however, executable (application/scenario-dependent) logics must be "hard-coded"; this mechanism allows at least to change resulting java classes on-the-fly

- **Versioning:**
  - pre-defined "module" – any topic map object will be annotated with metadata concerning, e.g., author, date, ancestor information

- **Meta Process Model:**
  - another pre-defined "module" which forces participants to re-concile definitions whenever they commit or checkout changes from/into their workspace
Existing Tools/Libraries

- **TMAPI (Common Topic Map Application Programming Interface, www.tmapi.org):**
  - well-known programming interface for accessing and manipulating data held in a topic map

- **H2 Database Engine (http://www.h2database.com):**
  - very fast, free SQL database with a small memory footprint written in Java featuring a JDBC and (partial) ODBC API; in-memory mode also makes it an alternative to TinyTIM (http://tinytim.sourceforge.net)

- **Jakarta Commons Id Component (http://commons.apache.org/sandbox/id/):**
  - component used to generate identifiers which offers several different algorithms that are also suitable for distributed collaboration/development scenarios

- **ANTLR 3.x (http://www.antlr.org):**
  - parser generator that comes with ANTLRWorks, a very useful graphical tool to test your grammars
Prototype: Ongoing Work

- modularised querying/filtering support for preliminary prototype
  - evaluation with regard to more complex development scenarios
  - combination with other Eclipse plugins, e.g., pattern scanners

- medium-term objective: support for coping with ontology changes and/or mergers within meta process model subphases

- long-term objective: support of operations as needed for decentralised version management (n-way merge)
Thank you!

e-mail to:
ueberall@tm.informatik.uni-frankfurt.de