Multi Back-Ends for a Model Library Abstraction Layer

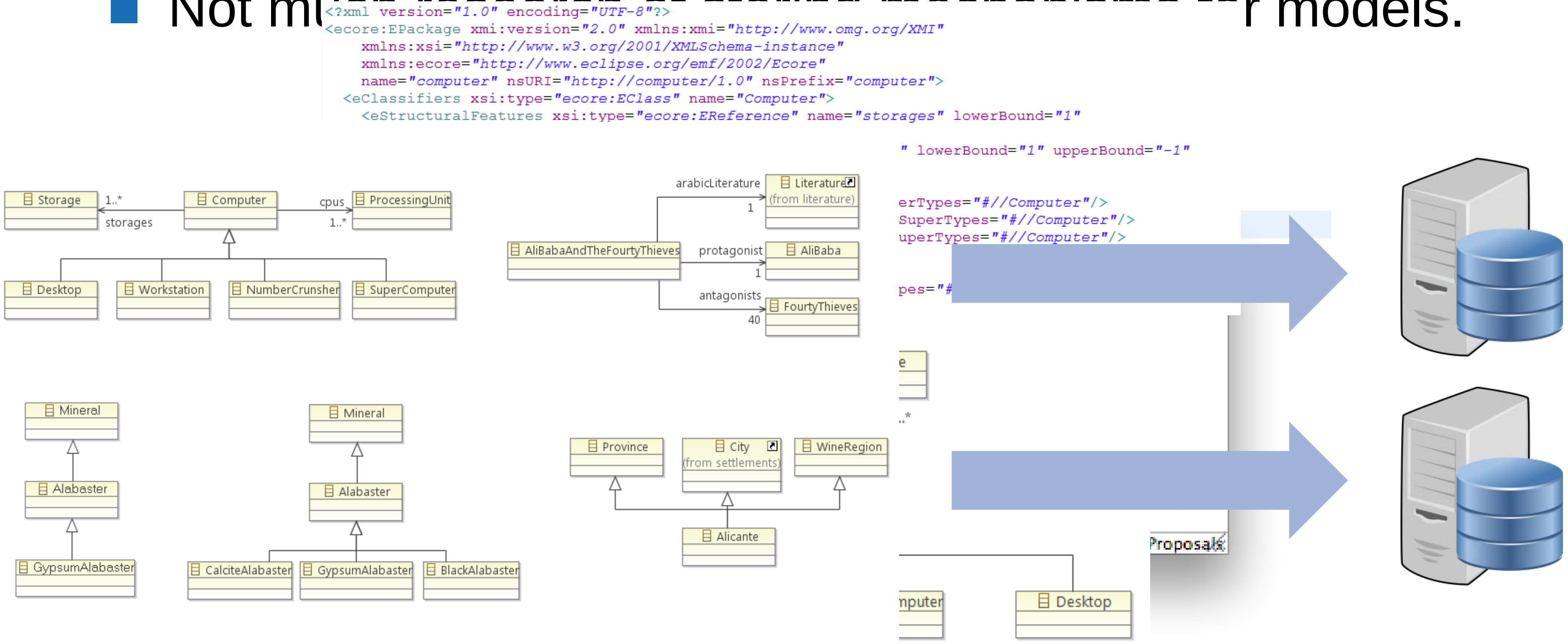
Tran Ngoc Viet, Andreas Ganser, and Horst Lichter
Software Construction, RWTH Aachen University, Aachen, Germany
viet.tran.ngoc@rwth-aachen.de, {ganser, lichter}@cs.rwth-aachen.de
Home page: http://www.swc.rwth-aachen.de

ICCSA Hochiminh, June 2013

- ➢Motivation
- ➢Recommender System
- ➢Concept
- ➢Realization
- ➢Result and Future Work

Model Libraries

- Reuse is not a new term.
- Model reuse Challenges.
- Model reuse systems have model repositories.
- Not much research of storing mechanisms for models.



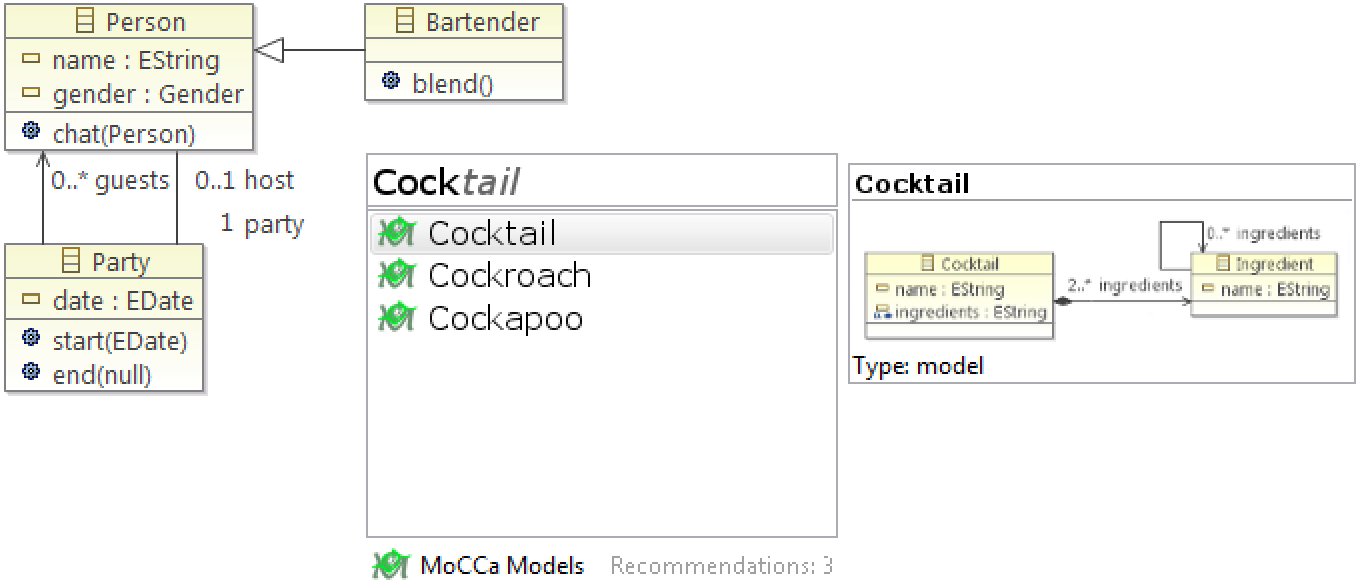
```
<?xml version="1.0" encoding="UTF-8"?>
<ecore:EPackage xmi:version="2.0" xmlns:xmi="http://www.omg.org/XMI"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:ecore="http://www.eclipse.org/emf/2002/Ecore"
    name="computer" nsURI="http://computer/1.0" nsPrefix="computer">
  <eClassifiers xsi:type="ecore:EClass" name="Computer">
    <eStructuralFeatures xsi:type="ecore:EReference" name="storages" lowerBound="1"
```

```
" lowerBound="1" upperBound="-1"
erTypes="#//Computer"/>
SuperTypes="#//Computer"/>
uperTypes="#//Computer"/>
pes="#
```

- ➢Motivation
- ➢Recommender System
- ➢Concept
- ➢Realization
- ➢Result and Future Work

Recommender System

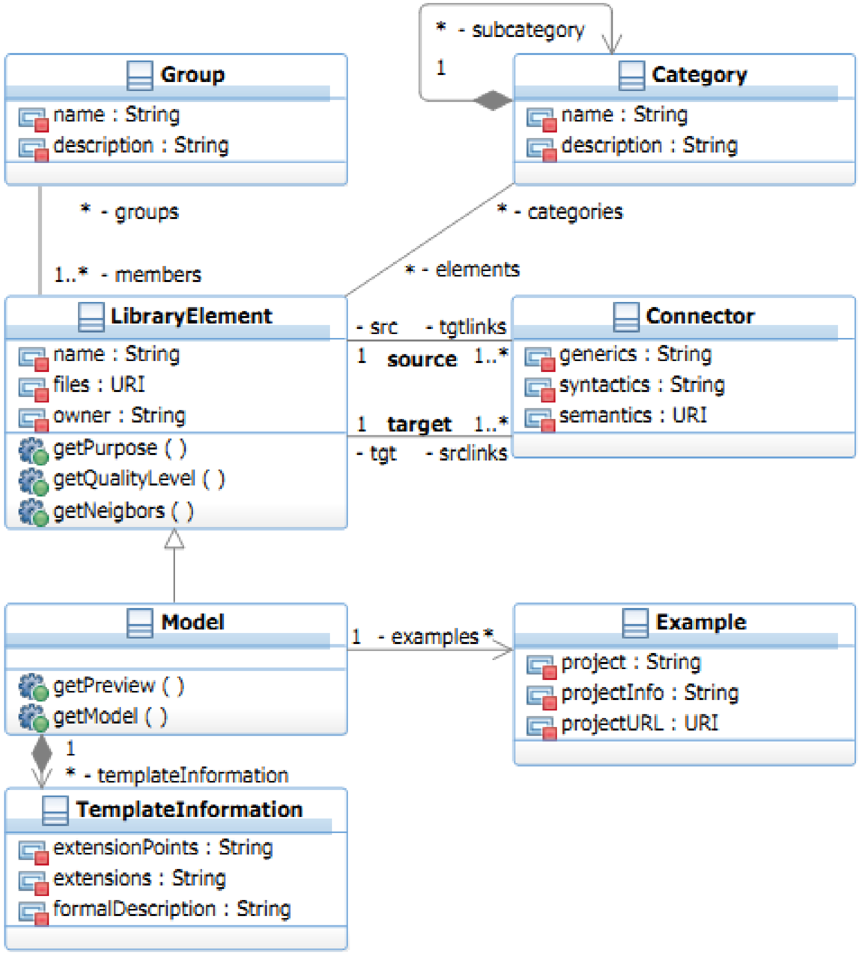
- Model for Model Libraries supporting for multi-backends with different databases.
- Close the gap between servers (databases holding the models) and the local application.



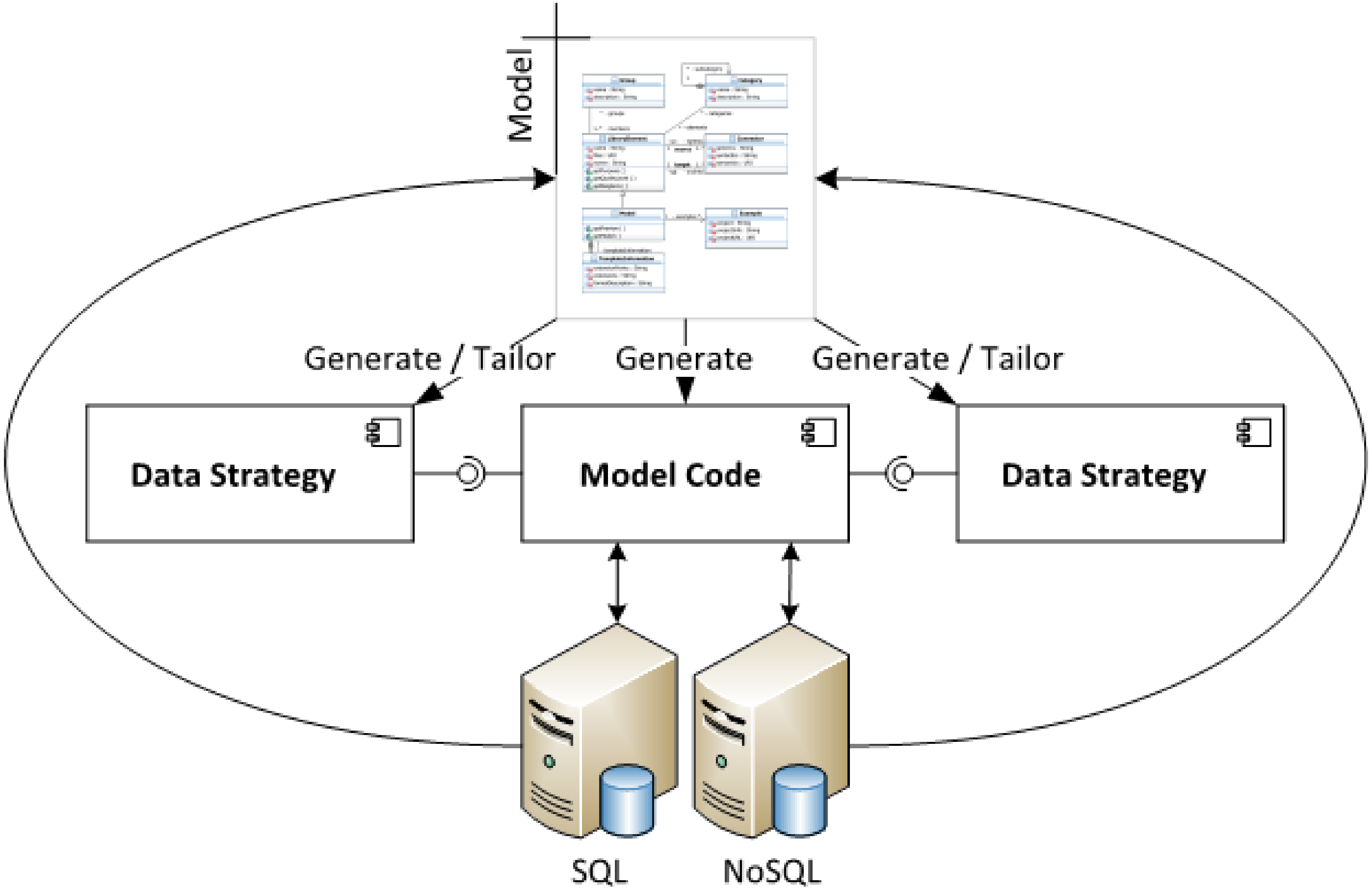
- Motivation
- Recommender System
- Concept
- Realization
- Result and Future Work

Model domain model

- LibraryElement has 3 attributes name, files and owner.
- LibraryElements could be grouped or categorized.
- Connectors present relationships of LibraryElements.
- Different mapping approaches for Connector.

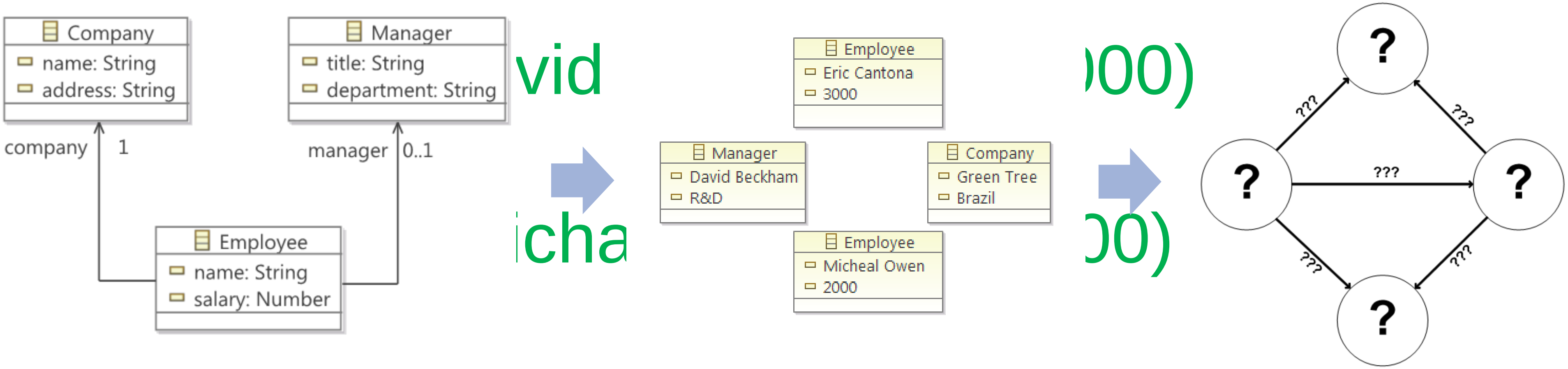


Model Library Abstraction



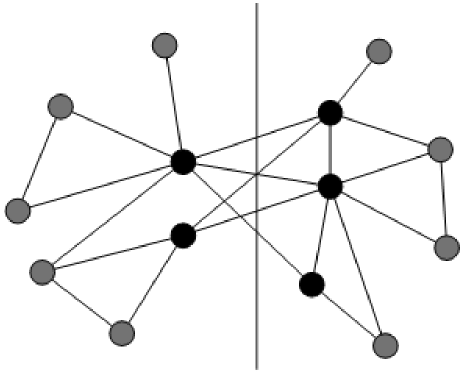
Mapped Library Entities

- Objects and Elements (Vertices and Edges).
- Attributes and Properties.
- References and Edges.
- Connector objects are mapped with edges while others are mapped with vertices.

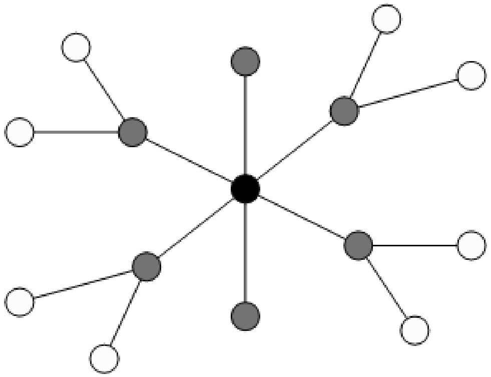


Loading and Saving Mechanisms

- Divide the graph into subgraphs:
  - How do we keep data of two subsets after cutting?
  - How do we decide on a suitable cut-set?
  - How do we keep the information of the cut-set?



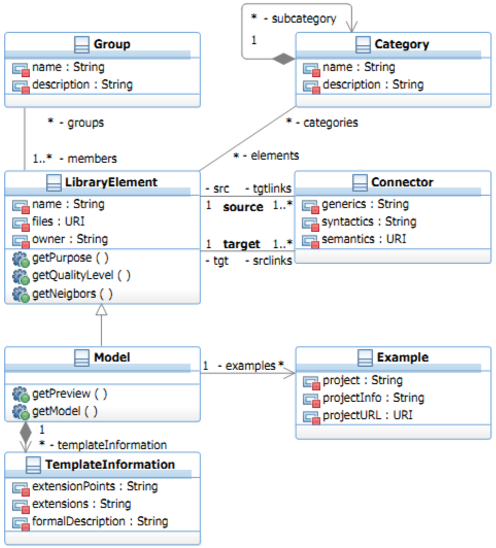
- Load data in cut-set balls
  - Return users what they exaclty need.
  - Set the level of loading.
  - Getting neighbors of a LibraryElement

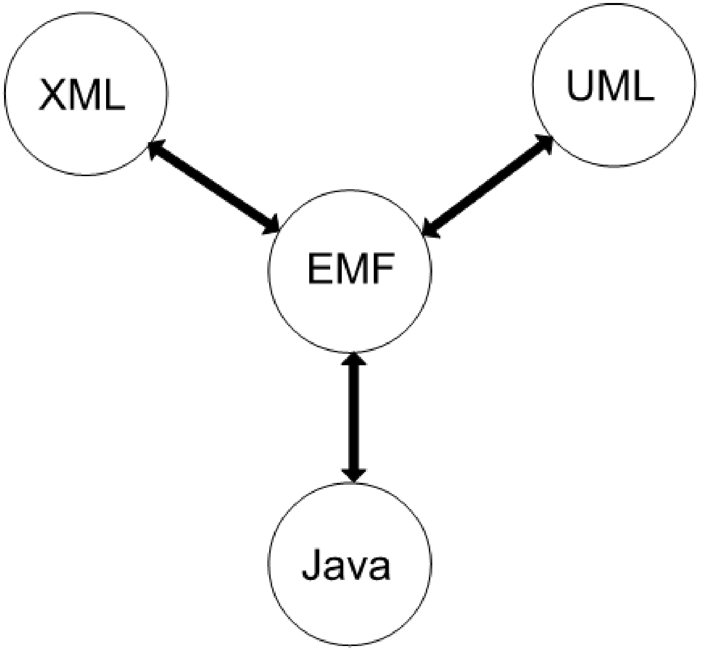


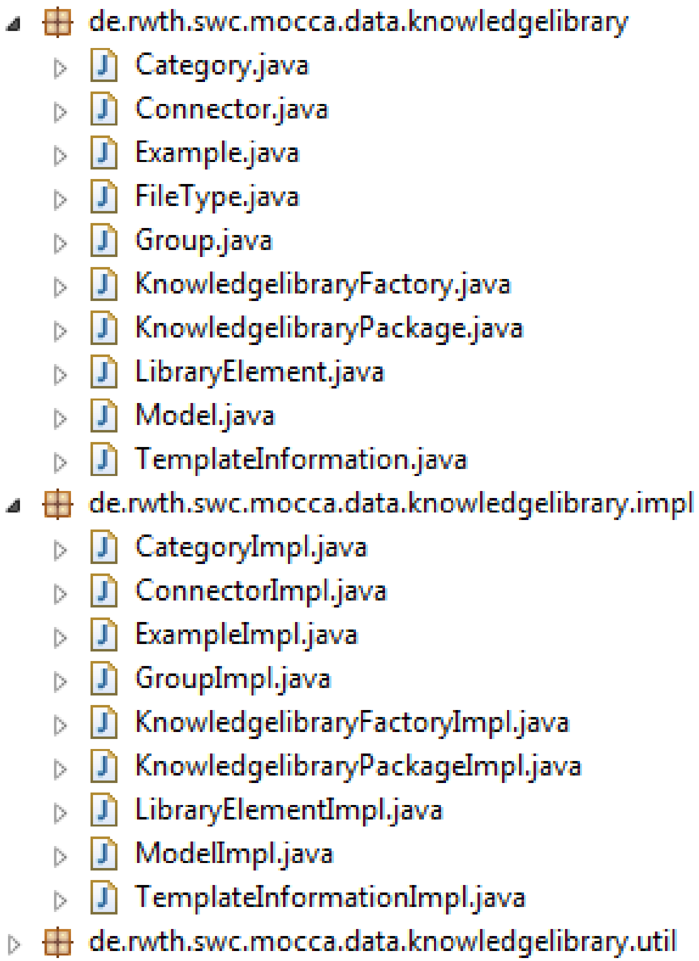
- Save data which are changed.
  - New objects or deleted objects.
  - Track the changes on objects.

- ➢Motivation
- ➢Recommender System
- ➢Concept
- ➢Realization
- ➢Result and Future Work

EMF Code Generation

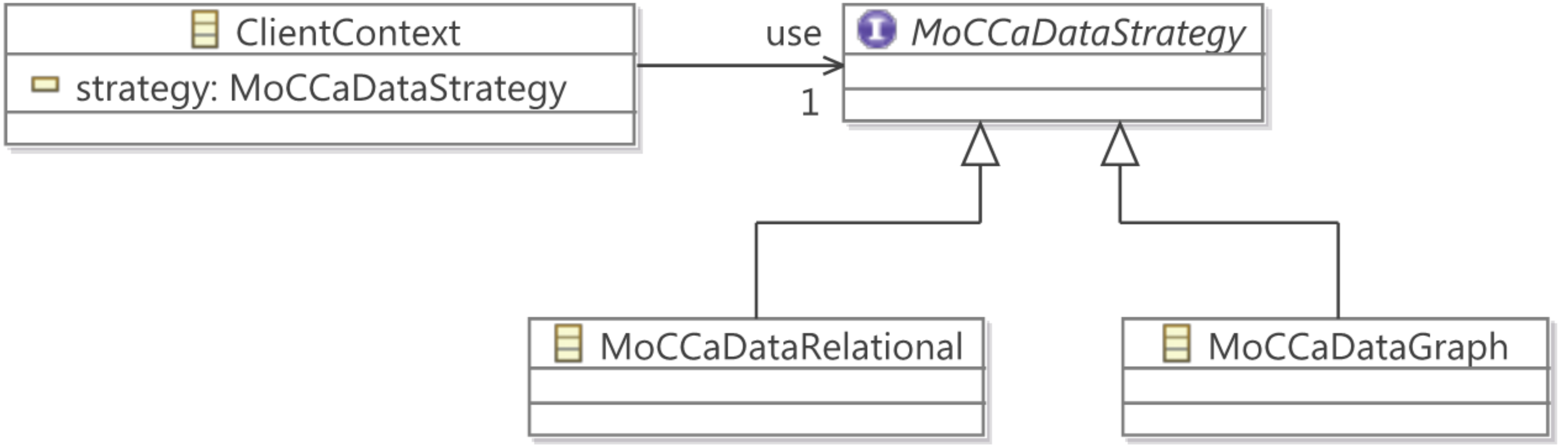






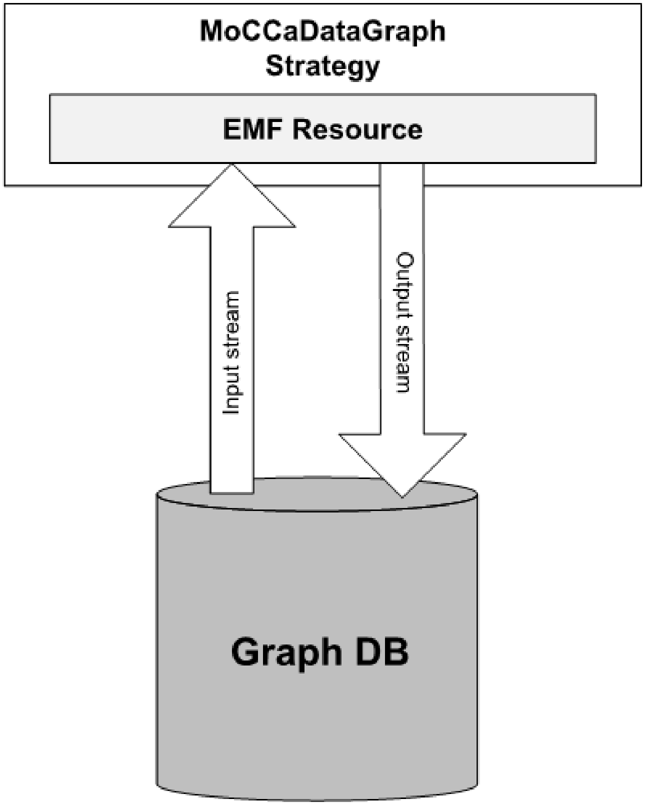
- de.rwth.swc.mocca.data.knowledgelibrary
  - Category.java
  - Connector.java
  - Example.java
  - FileType.java
  - Group.java
  - KnowledgelibraryFactory.java
  - KnowledgelibraryPackage.java
  - LibraryElement.java
  - Model.java
  - TemplateInformation.java
- de.rwth.swc.mocca.data.knowledgelibrary.impl
  - CategoryImpl.java
  - ConnectorImpl.java
  - ExampleImpl.java
  - GroupImpl.java
  - KnowledgelibraryFactoryImpl.java
  - KnowledgelibraryPackageImpl.java
  - LibraryElementImpl.java
  - ModelImpl.java
  - TemplateInformationImpl.java
- de.rwth.swc.mocca.data.knowledgelibrary.util

Data Strategy



Graph Streams

- Graph Input Stream
  - Load graph elements into EMF Resources.
  - Resources have URIs with parameters.
  - No duplication.
- Graph Output Stream
  - Update graphs from EMF Resources.
  - Resources just hold which objects that the users want to update.

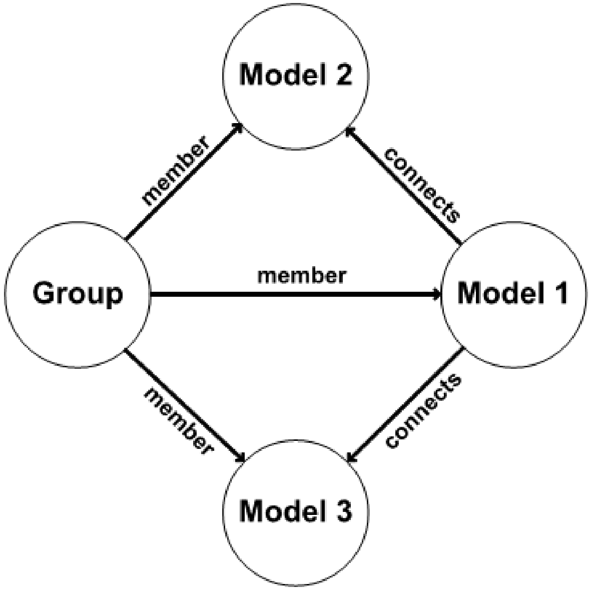


Object Graph Mapping

- Two directions:
  - Programming objects to graph elements
  - Graph elements to programming objects

- Avoid recursion in mapping
  - Circles of references or connections
  - Keep the mapping of object hashcodes

and element ID



Loading and Saving Mechanisms

- The loading mechanism
  - Find/Get graph elements based on property values and names.
  - Create eObject from vertex.
  - Put on a resource content list after mapping hashcode and id.
  - Return to users what they need.

- The saving mechanism
  - Put new or loaded but changed eObjects into content list.
  - Create or find mapped vertices or edges, then fill vertices and edges.
  - Map eObjects and vertices/edges.

Relational Strategy

- Configuration
  - Hibernate: database and user parameters
  - Teneo: mapping (inheritance, entities), names
- Initialize DataStore:
  - Register Epackage (KnowledgeLibraryPackage)
- The saving process
  - Support of Hibernate Session.

- Motivation
- Recommender System
- Concept
- Realization
- Result and Future Work

Result and Future Work

- Result:
  - Modeled the model libraries.
  - Researched the Abstraction layer for MoCCa.
  - Object Graph Mapping
  - Interchangeable and extendable databases
  - Reliable Loading and Saving Mechanisms for Graph Streams
  - Efficient saving process with data limition in graph data streams.
- Future Work:
  - Improve the loading and saving mechanisms.
  - Simulate the graph traversal.
  - Implement and test for other databases.
  - Out of boundary of EMF.

Thank you!

Time for Questions