Chapter 14: Inheritance

The objective of this chapter is to explain how inheritance [Inheritance] is used in Modelibra and Wicket outside the inheritance found in Java. In Modelibra, an XML configuration may inherit another XML configuration. In Wicket, a panel's HTML code may extend another panel's HTML code. In addition, an inheritance of models in ModelibraModeler is presented.

In this chapter a new web application will be presented. The application is called CourseQuiz. The application provides formative quizzes so that a user may test her knowledge of a certain subject matter. There are no grades and the answers are not recorded.

Domain Models

The Course domain has two models: Reference and Quiz. The Reference model has three entry concepts: SecurityRole, CountryLanguage and QuestionType. The SecurityRole concept provides valid security roles for the Member concept in the Quiz model. The CountryLanguage concept has natural languages offered to users as a choice of an international version of the application. The QuestionType concept is used to validate types of questions for the Question concept in the Quiz model.

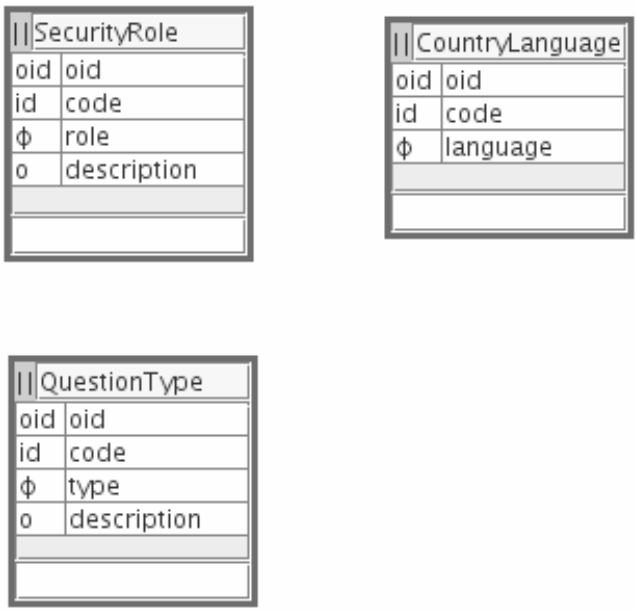

Figure 14.1. Reference Model

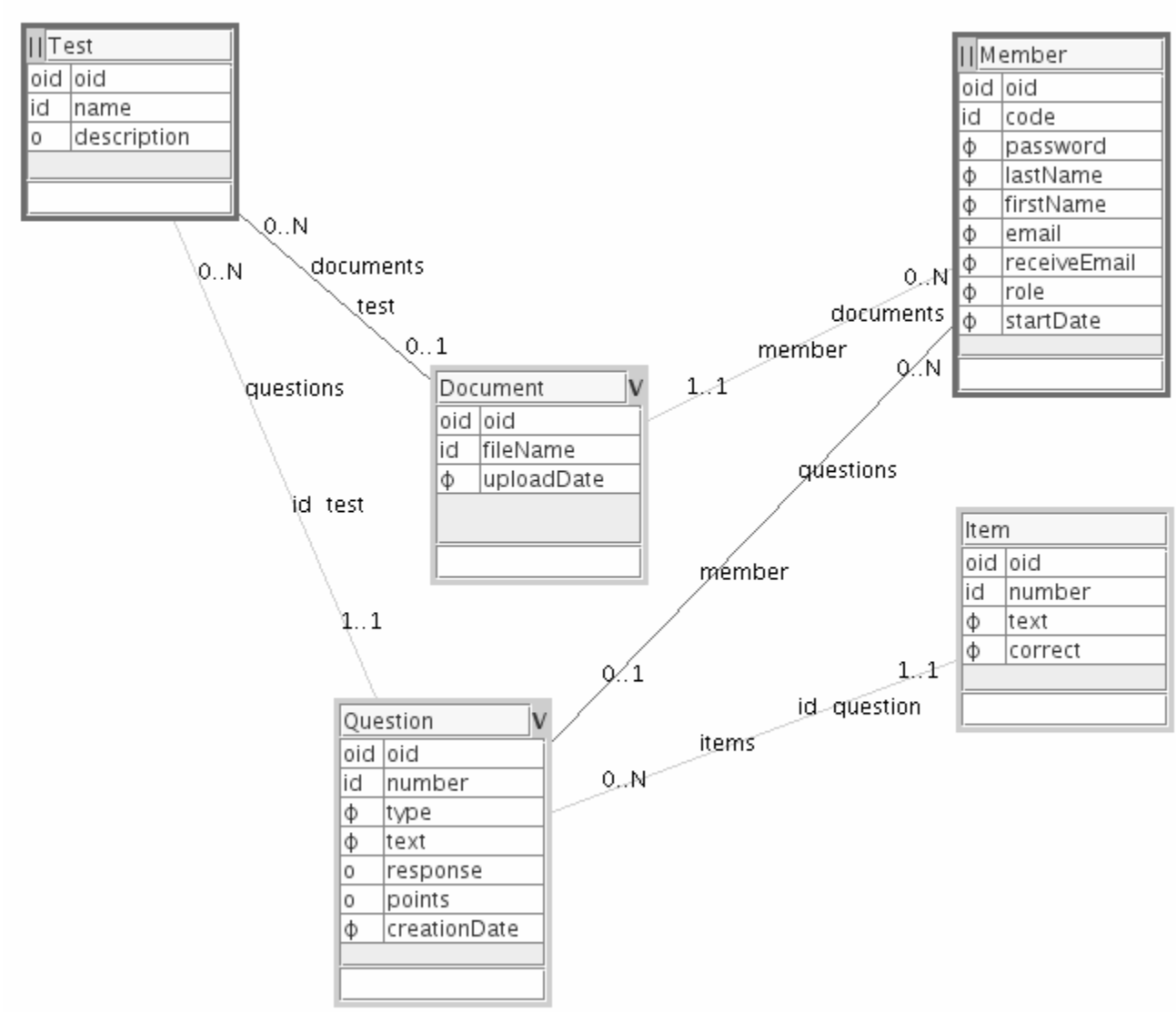

Figure 14.2. Quiz Model

The two entry points into the Quiz model are the Test and Member concepts. A test has many questions that in turn have several items. A test may have documents that explain the subject matter covered by the test. Documents belong to a member that uploaded them to the application. A document may exist without being related to a test. A question that is linked to a member indicates that the member entered the question.

The application allows its members (e.g., students) to enter questions and upload documents to help an administrator (e.g., a professor) to provide rich formative quizzes. A formative quiz is not graded. Its purpose is to help a student to test her new knowledge.

The initial version of this application was generated by ModelibraWicketSkeleton. Since the principal model has the Test concept, the model is not called Test but Quiz. In this way, any confusion between the model and the concept is avoided. It is recommended in Modelibra not to name a domain or a model in the same way as one of the model's concepts.

Domain Configuration

The Course domain and the Quiz and Reference models are initially configured in the reusable-domain-config.xml file. Actually, this configuration file was generated by ModelibraModeler. The specific configuration is entered in the specific-domain-config.xml file.

There is a new configuration file called modelibra-domain-config.xml. The Modelibra configuration has the Dm domain and the Reference model (different from the Reference model in the Course domain). The Reference model in the Modelibra configuration has five reusable concepts: CountryLanguage, CountryName, Member, QuestionType and SecurityRole. The DmReference.jar file of this model can be found in the lib directory of the CourseQuiz project.

The CountryLanguage and SecurityRole concepts from the Reference model of the Course domain extend their configurations from the corresponding concepts in the Reference model of the Dm domain. The QuestionType concept does not use the inheritance. This is done to show that you have a choice of what you want to inherit and what you want to define yourself (or what is available for inheritance).

```
<domains>

      <domain oid="1196702256688">
            <code>Course</code>
            <type>Specific</type>
            <referenceModel>Reference</referenceModel>

            <i18n>false</i18n>
            <signin>true</signin>
            <signinConcept>Member</signinConcept>
            <shortTextDefaultLength>48</shortTextDefaultLength>
            <pageBlockDefaultSize>16</pageBlockDefaultSize>
            <validateForm>false</validateForm>

            <models>

                  ...

                  <model oid="1176746269771">
                        <code>Reference</code>
                        <extension>false</extension>
                        <author>Dzenan Ridjanovic</author>
                        <persistenceType>xml</persistenceType>
                        <persistenceRelativePath>
                              data/xml/course/reference
                        </persistenceRelativePath>
                        <defaultLoadSave>true</defaultLoadSave>

                        <concepts>

                              <concept oid="1171894940420">
                              <code>CountryLanguage</code>
                              <extension>true</extension>
```

```
<extensionDomain>Dm</extensionDomain>
<extensionDomainType>Modelibra</extensionDomainType>
<extensionModel>Reference</extensionModel>
<extensionConcept>CountryLanguage</extensionConcept>

</concept>

<concept oid="1171894940520">
<code>SecurityRole</code>
      <extension>true</extension>
      <extensionDomain>Dm</extensionDomain>
      <extensionDomainType>Modelibra</extensionDomainType>
<extensionModel>Reference</extensionModel>
<extensionConcept>SecurityRole</extensionConcept>
</concept>

<concept oid="1176746303134">
      <code>QuestionType</code>
      <entitiesCode>QuestionTypes</entitiesCode>
      <entry>true</entry>

      <properties>
            <property oid="1176746310605">
                  <code>code</code>
                  <propertyClass>
                        java.lang.String
                  </propertyClass>
                  <required>true</required>
                  <unique>true</unique>

                  <essential>true</essential>
            </property>
            <property oid="1176746319184">
                  <code>type</code>
                  <propertyClass>
                        java.lang.String
                  </propertyClass>
                  <maxLength>32</maxLength>
                  <required>true</required>

                  <essential>true</essential>
            </property>
            <property oid="1176746330779">
                  <code>description</code>
                  <propertyClass>
                        java.lang.String
                  </propertyClass>
                  <maxLength>510</maxLength>

                  <essential>false</essential>
            </property>
      </properties>
```

```
                        </concept>

                    </concepts>

                </model>

            </models>

        </domain>

</domains>
```

The names of concepts do not have to have the same name when extended. Also, the whole model may be extended.

In Modelibra there are three predefined configuration files. The specific-domain-config.xml is reserved for specific configurations of the application model. In general, the content of reusable-domain-config.xml may be reused in different applications. The modelibra-domain-config.xml file exists in the source code of the Modelibra project and may be reused to share concepts that appear often in many applications.

The code is generated from all configurations as if definitions were at the level of the specific-domain-config.xml file. If the code was done by hand, the inheritance could have been used also at the level of Java classes. For example, the Member concept could have extended the Member concept in the Modelibra configuration. In addition, the Member class in the Course domain could have extended the Member class in the Dm domain. In order to show how this can be done, the CountryLanguage and SecurityRole concepts inherit classes from the DmReference.jar library.

```
package course.reference.countrylanguage;

import org.modelibra.IDomainModel;

public class CountryLanguage extends dm.reference.countrylanguage.CountryLanguage {

	public CountryLanguage(IDomainModel model) {
		super(model);
	}

}
```

```
package course.reference.countrylanguage;

import org.modelibra.IDomainModel;

public class CountryLanguages extends
		dm.reference.countrylanguage.CountryLanguages {

	public CountryLanguages(IDomainModel model) {
		super(model);
	}
```

```
}

package course.reference.securityrole;

import org.modelibra.IDomainModel;

public class SecurityRole extends dm.reference.securityrole.SecurityRole {

	public SecurityRole(IDomainModel model) {
		super(model);
	}

}

package course.reference.securityrole;

import org.modelibra.IDomainModel;

public class SecurityRoles extends dm.reference.securityrole.SecurityRoles {

	public SecurityRoles(IDomainModel model) {
		super(model);
	}

}
```

In the Course domain the reference model (referenceModel XML element) is Reference from the same domain. The reference model is used for property type validations. The type property in the Question concept is validated based on values in the QuestionType entities, i.e, in QuestionTypes. The validation is done based on the code property. If there is no QuestionType concept in the principal model, the reference model of the same domain is used to find valid values.

```
<domains>

	<domain oid="1196702256688">
		<code>Course</code>
		<type>Specific</type>
		<referenceModel>Reference</referenceModel>

		<i18n>false</i18n>
		<signin>true</signin>
		<signinConcept>Member</signinConcept>
		<shortTextDefaultLength>48</shortTextDefaultLength>
		<pageBlockDefaultSize>16</pageBlockDefaultSize>
		<validateForm>false</validateForm>
...
		<models>

			<model oid="1176742903870">
				<code>Quiz</code>
```

```
                        <extension>true</extension>
                        <extensionDomain>Course</extensionDomain>
                        <extensionDomainType>Reusable</extensionDomainType>
                        <extensionModel>Quiz</extensionModel>
...
                              <concept oid="1176743111750">
                                    <code>Question</code>
                                    <extension>true</extension>
                                    <extensionConcept>Question</extensionConcept>
...
                                          <property oid="1176743128597">
                                                <code>type</code>
                                                <extension>true</extension>
                                                <extensionProperty>type</extensionProperty>

                                                <validateType>true</validateType>
                                                <validationType>
                                                      QuestionTypes
                                                </validationType>

                                                <displayLength>8</displayLength>
                                                <essential>false</essential>
                                          </property>
...
```

Home Page

The HomePage class has a few components. The header and footer components are defined for this page only, but in principle they could have been used in other pages as well.

```
package course.wicket.app.home;

import org.apache.wicket.markup.html.panel.Panel;
import org.modelibra.wicket.concept.EntityDisplayMinPanel;
import org.modelibra.wicket.container.DmPage;
import org.modelibra.wicket.view.View;
import org.modelibra.wicket.view.ViewModel;

import course.Course;
import course.quiz.Quiz;
import course.quiz.question.Question;
import course.quiz.question.Questions;
import course.quiz.test.Tests;
import course.reference.Reference;
import course.reference.countrylanguage.CountryLanguages;
import course.wicket.app.CourseApp;
import course.wicket.quiz.test.TestDisplayTablePanel;

public class HomePage extends DmPage {
```

```
	public HomePage() {
		CourseApp courseApp = (CourseApp) getApplication();
		Course course = courseApp.getCourse();
		Quiz quiz = course.getQuiz();
		Reference reference = course.getReference();

		// Header
		add(new HomePageHeaderPanel("headerSection", "Quiz"));

		// Menu
		ViewModel mainMenuViewModel = new ViewModel();
		mainMenuViewModel.setModel(reference);
		CountryLanguages languages = reference.getCountryLanguages();
		mainMenuViewModel.setEntities(languages);

		View mainMenuView = new View();
		mainMenuView.setPage(this);
		mainMenuView.setWicketId("menuSection");

		add(new HomePageMenuPanel(mainMenuViewModel, mainMenuView));

		// Quizzes
		Tests tests = quiz.getTests();
		Tests orderedQuizzes = tests.getTestsOrderedByName(true);
		add(new TestDisplayTablePanel("quizTableSection", orderedQuizzes));

		// Random question
		Panel randomQuestionSection = null;

		ViewModel questionViewModel = new ViewModel();
		questionViewModel.setModel(quiz);
		Questions questions = quiz.getQuestions();
		if (questions.isEmpty()) {
			randomQuestionSection = new Panel("randomQuestionSection");
			randomQuestionSection.setVisible(false);
		} else {
			Question question = questions.random();
			questionViewModel.setEntity(question);

			View questionView = new View();
			questionView.setWicketId("randomQuestionSection");

			randomQuestionSection = new EntityDisplayMinPanel(
					questionViewModel, questionView);
		}
		add(randomQuestionSection);

		// Footer
		add(new HomePageFooterPanel("footerSection"));
	}

}
```

The header component is defined in the HomePageHeaderPanel class.

```
package course.wicket.home;

import org.dmlite.wicket.container.HeaderPanel;

public class HomePageHeaderPanel extends HeaderPanel {

	public HomePageHeaderPanel(final String wicketId, final String title) {
		super(wicketId, title);
	}

}
```

The class extends the HeaderPanel class. In addition, the HomePageHeaderPanel.html definition extends the HTML definition of the HeaderPanel class. The <wicket:extend> element indicates the extension.

```
<?xml version="1.0" encoding="UTF-8"?>

<html xmlns:wicket>

<wicket:extend>

	<a href="http://www.typogenerator.net/">
		<img src="img/quiz.jpg" border="0" alt="Quiz" />
	</a>

</wicket:extend>

</html>
```

The HeaderPanel class comes from the org.modelibra.wicket.container package of ModelibraWicket:

```
package org.dmlite.wicket.container;

import wicket.markup.html.basic.Label;

public class HeaderPanel extends DmPanel {

	public HeaderPanel(final String wicketId, final String title) {
		super(wicketId);
		add(new Label("title", title));
	}

}
```

The following is a HTML declaration of its panel:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<html xmlns:wicket>

<wicket:panel>

<div>
      <wicket:child/>
      <span wicket:id="title" class="app-title">
            Title
      </span>
</div>

</wicket:panel>

</html>
```

The <wicket:child/> element indicates where the HTML code may be extended.

Similarly, the HomePageFooterPanel class extends FooterPanel from ModelibraWicket.

```
package course.wicket.home;

import org.modelibra.wicket.container.FooterPanel;

public class HomePageFooterPanel extends FooterPanel {

      public HomePageFooterPanel(final String wicketId) {
            super(wicketId);
      }

}
```

The HomePageFooterPanel.html file extends the HTML declaration of the FooterPanel class.

```
<?xml version="1.0" encoding="UTF-8"?>

<html xmlns:wicket>

<wicket:extend>

    <a href="http://www.coolquiz.com/">Cool Quiz</a> |

</wicket:extend>

</html>
```

The following are the FooterPanel class and its HTML presentation.

```
package org.modelibra.wicket.container;

public class FooterPanel extends DmPanel {
```

```
	public FooterPanel(final String wicketId) {
		super(wicketId);
	}

}
```

```
<?xml version="1.0" encoding="UTF-8"?>

<html xmlns:wicket>

<wicket:panel>

<div>
	| <a href="http://www.javaforge.com/">Modelibra</a> |
	<a href="http://validator.w3.org/">XHTML</a> |
	<a href="http://jigsaw.w3.org/css-validator/">CSS</a> |
	<a href="http://bobby.watchfire.com/">508</a> |
	<wicket:child/>
</div>

</wicket:panel>

</html>
```

Home Page Menu

The HomePageMenuPanel class has the *Upload* and *About* links.

```
package course.wicket.app.home;

import org.apache.wicket.markup.html.link.Link;
import org.modelibra.wicket.container.MainMenuPanel;
import org.modelibra.wicket.view.View;
import org.modelibra.wicket.view.ViewModel;

import course.wicket.app.about.AboutPage;
import course.wicket.app.upload.UploadPage;

public class HomePageMenuPanel extends MainMenuPanel {

	public HomePageMenuPanel(final ViewModel viewModel, final View view) {
		super(viewModel, view);

		add(new Link("upload") {
			public void onClick() {
				setResponsePage(UploadPage.class);
			}
		});
```

```
            add(new Link("about") {
                  public void onClick() {
                        setResponsePage(AboutPage.class);
                  }
            });
      }

}
```

The HomePageMenuPanel.html definition extends MainMenuPanel.html.

```
<?xml version="1.0" encoding="UTF-8"?>

<html xmlns:wicket>

<wicket:extend>

      <li>
            <a wicket:id="upload">
                  <wicket:message key = "menu.upload"/>
            </a>
      </li>
      <li>
            <a wicket:id="about">
                  <wicket:message key = "menu.about"/>
            </a>
      </li>

</wicket:extend>

</html>
```

The MainMenuPanel class is defined in ModelibraWicket.

```
package org.modelibra.wicket.container;

import org.apache.wicket.markup.html.link.Link;
import org.apache.wicket.markup.html.link.PageLink;
import org.apache.wicket.markup.html.panel.Panel;
import org.modelibra.Entities;
import org.modelibra.IEntity;
import org.modelibra.IDomainModel;
import org.modelibra.wicket.app.DomainApp;
import org.modelibra.wicket.concept.CountryLanguageChoicePanel;
import org.modelibra.wicket.security.AppSession;
import org.modelibra.wicket.security.SigninPage;
import org.modelibra.wicket.view.View;
import org.modelibra.wicket.view.ViewModel;

public class MainMenuPanel extends DmPanel {
```

```
public MainMenuPanel(final ViewModel viewModel, final View view) {
	super(viewModel, view);
	final DomainApp domainApp = (DomainApp) getApplication();

	Link domainPageLink = new Link("domain") {
		public void onClick() {
			setResponsePage(domainApp.getDomainPageClass());
		}
	};
	domainPageLink.setAutoEnable(true);
	add(domainPageLink);

	// Sign in & out
	final AppSession appSession = getAppSession();

	Link signinLink = new PageLink("signin", SigninPage.class);
	add(signinLink);
	Link signoutLink = new Link("signout") {
		public void onClick() {
			appSession.invalidate();
			setResponsePage(view.getPage().getClass());
		}
	};
	add(signoutLink);
	if (!domainApp.getDomain().getDomainConfig().isSignin()) {
		signinLink.setVisible(false);
		signoutLink.setVisible(false);
	} else if (appSession.isUserSignedIn()) {
		signinLink.setVisible(false);
	} else {
		signoutLink.setVisible(false);
	}

	// I18n
	ViewModel languageViewModel = new ViewModel();
	IDomainModel reference = viewModel.getModel();
	languageViewModel.setModel(reference);
	Entities<?> languages = (Entities<?>) viewModel.getEntities();
	languageViewModel.setEntities(languages);
	String languageCode = null;
	IEntity<?> defaultLanguage = null;
	languageCode = appSession.getLocale().getLanguage();
	defaultLanguage = languages.retrieveByCode(languageCode);
	if (defaultLanguage == null) {
		defaultLanguage = languages.retrieveByCode("en");
	}
	languageViewModel.setEntity(defaultLanguage);

	View languageView = new View();
	languageView.setWicketId("languageChoiceSection");
```

```
            Panel languageChoicePanel = new CountryLanguageChoicePanel(
                        languageViewModel, languageView);
            add(languageChoicePanel);
            if (!domainApp.getDomain().getDomainConfig().isI18n()) {
                  languageChoicePanel.setVisible(false);
            }
      }

}
```

```
<?xml version="1.0" encoding="UTF-8"?>

<html xmlns:wicket>

<wicket:panel>

<div class="menu">
      <ul>
            <li>
                  <a wicket:id="signin">
                        <wicket:message key = "menu.signin"/>
                  </a>
            </li>
            <li>
                  <a wicket:id="signout">
                        <wicket:message key = "menu.signout"/>
                  </a>
            </li>
            <li>
                  <a wicket:id="domain">
                        <wicket:message key = "menu.domain"/>
                  </a>
            </li>
            <wicket:child/>
             
             
            <span wicket:id="languageChoiceSection">
                  To be replaced dynamically by the language choice.
            </span>
             
      </ul>
</div>

</wicket:panel>

</html>
```

Model Inheritance in ModelibraModeler

In ModelibraModeler a model may inherit all or some concepts from another model. In addition to

inherited concepts, new concepts, new properties, and new relationships may be added to specialize the model. Since this is a rather complex topic, the spiral approach to explaining the model inheritance in ModelibraModeler will be practiced.

In the mm directory of the CourseQuiz project there is the Modelibra.diagram file that contains two models (diagrams). The first model is the meta model of Modelibra. It represents the Modelibra configuration. The second model is the reference model that contains few concepts. In the same directory there is also the ModelibraModeler.type file that contains types used in property definitions.

A way to start a new model in ModelibraModeler is to reuse existing models by importing first types and then diagrams. This can be accomplished by the *Transfer* menu in the main window (Figure 14.3). After diagrams are imported, some of them may be removed and some of them may be declared as abstract to indicate that they will be used in defining new models by inheritance.

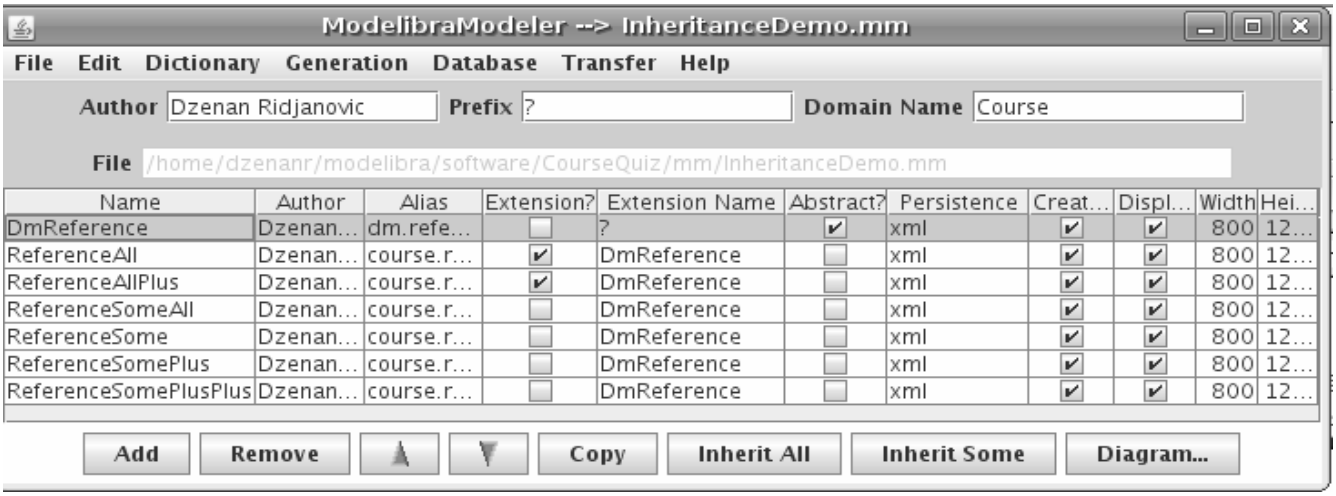

Figure 14.3. Main Window

In the mm directory of the CourseQuiz project there is the InheritanceDemo.mm file with the abstract DmReference model (Figure 14.4) obtained from the Modelibra.diagram file. The model is renamed from Reference to DmReference. Other models are defined by using the model inheritance in ModelibraModeler.

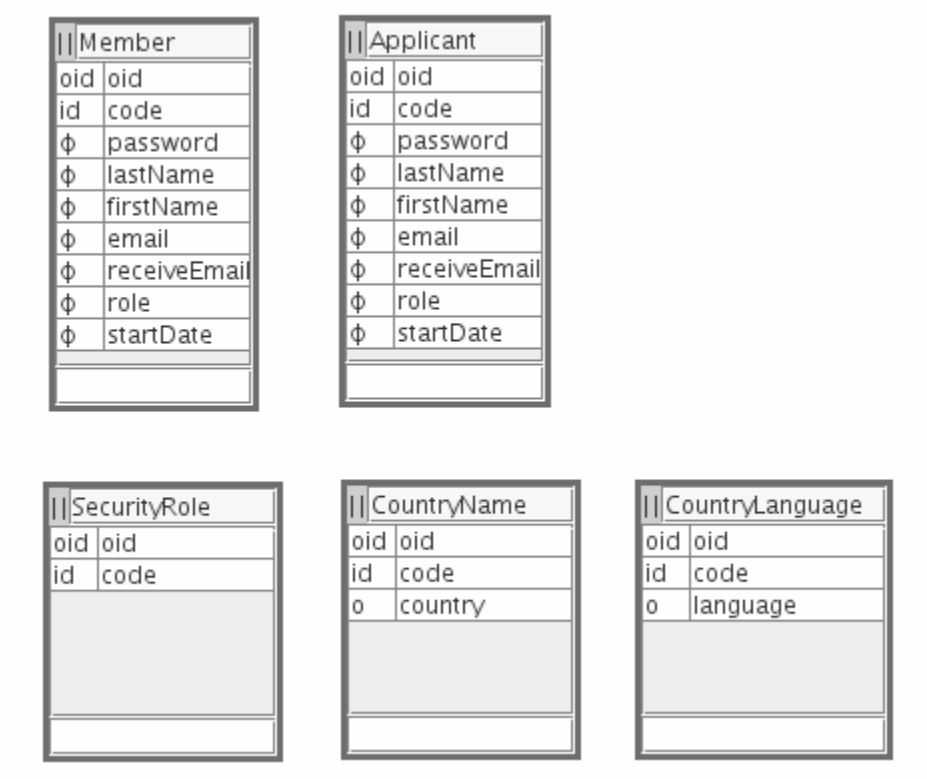

Figure 14.4. DmReference Model

The minimal Modelibra configuration is generated for all diagrams by the Generation menu in the main window. The generation output file is called InheritanceDemo.xml and is located in the mm directory of the CourseQuiz project. The XML configuration for the DmReference model indicates that the model is abstract.

```
<domain oid="1188594552974">
      <code>Course</code>
      <type>Reusable</type>

      <models>

            <model oid="1169579175415">
                  <code>DmReference</code>
                  <abstraction>true</abstraction>
                  <author>Dzenan Ridjanovic</author>
                  ...
```

The ReferenceAll model is inherited from the DmReference model. This is accomplished in ModelibraModeler first by selecting the DmReference model and second by clicking on the *Inherit All* button in the main window. The diagram of the ReferenceAll model is empty, but the diagram title shows that the model is inherited from another model: *ReferenceAll ==> DmReference*. The XML configuration shows that the model is extended.

```
            <model oid="1199036888032">
                  <code>ReferenceAll</code>
                  <extension>true</extension>
```

```
        <extensionDomain>Course</extensionDomain>
        <extensionModel>DmReference</extensionModel>
        <author>Dzenan Ridjanovic</author>
        <persistenceType>xml</persistenceType>
        <persistenceRelativePath>
              data/xml/course/referenceall</persistenceRelativePath>
        <defaultLoadSave>true</defaultLoadSave>

        <concepts>

        </concepts>

</model>
```

The ReferenceAllPlus model is inherited from the DmReference model. This is done first by selecting the DmReference model and second by clicking on the *Inherit All* button in the main window. In the diagram of the ReferenceAllPlus model the title shows that the model is inherited from another model: *ReferenceAllPlus ==> DmReference* (Figure 14.5). The new PostalCode concept is added (Figure 14.5).

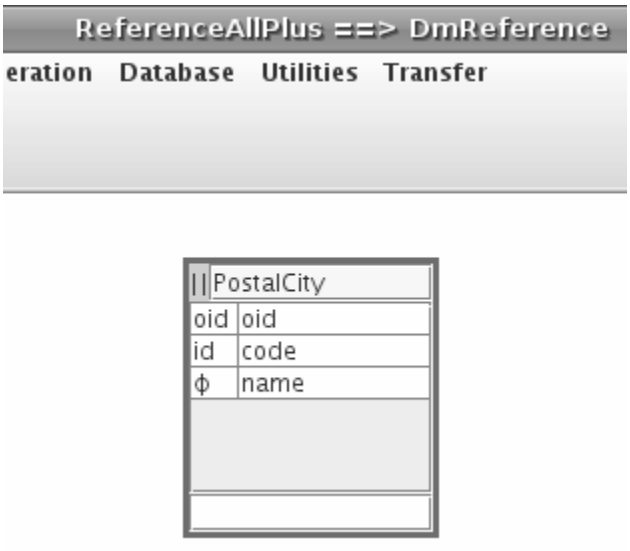

Figure 14.5. ReferenceAllPlus Model

The XML configuration shows the new concept.

```
<model oid="1199038341929">
      <code>ReferenceAllPlus</code>
      <extension>true</extension>
      <extensionDomain>Course</extensionDomain>
      <extensionModel>DmReference</extensionModel>
      <author>Dzenan Ridjanovic</author>
      <persistenceType>xml</persistenceType>
      <persistenceRelativePath>
            data/xml/course/referenceallplus
      </persistenceRelativePath>
      <defaultLoadSave>true</defaultLoadSave>
```

```
        <concepts>

            <concept oid="1199038395664">
                <code>PostalCity</code>
                <entitiesCode>PostalCities</entitiesCode>
                <entry>true</entry>

                <properties>
                    <property oid="1199038570978">
                        <code>code</code>
                        <propertyClass>
                            java.lang.String
                        </propertyClass>
                        <required>true</required>
                        <unique>true</unique>

                        <essential>true</essential>
                    </property>
                    <property oid="1199038574243">
                        <code>name</code>
                        <propertyClass>
                            java.lang.String
                        </propertyClass>
                        <required>true</required>

                        <essential>true</essential>
                    </property>
                </properties>

                <neighbors>
                </neighbors>

            </concept>

        </concepts>

    </model>
```

The ReferenceSomeAll model is inherited from the DmReference model, not by extending the model but by extending all concepts. This is done first by selecting the DmReference model and second by clicking on the *Inherit Some* button in the main window. In the diagram of the ReferenceSomeAll model the title is: *ReferenceSomeAll -- DmReference* (Figure 14.6). There are five concepts without properties, since properties are inherited from the same concepts in the DmReference model. If there had been neighbors they would have been inherited as well, but not shown as is the case with properties.

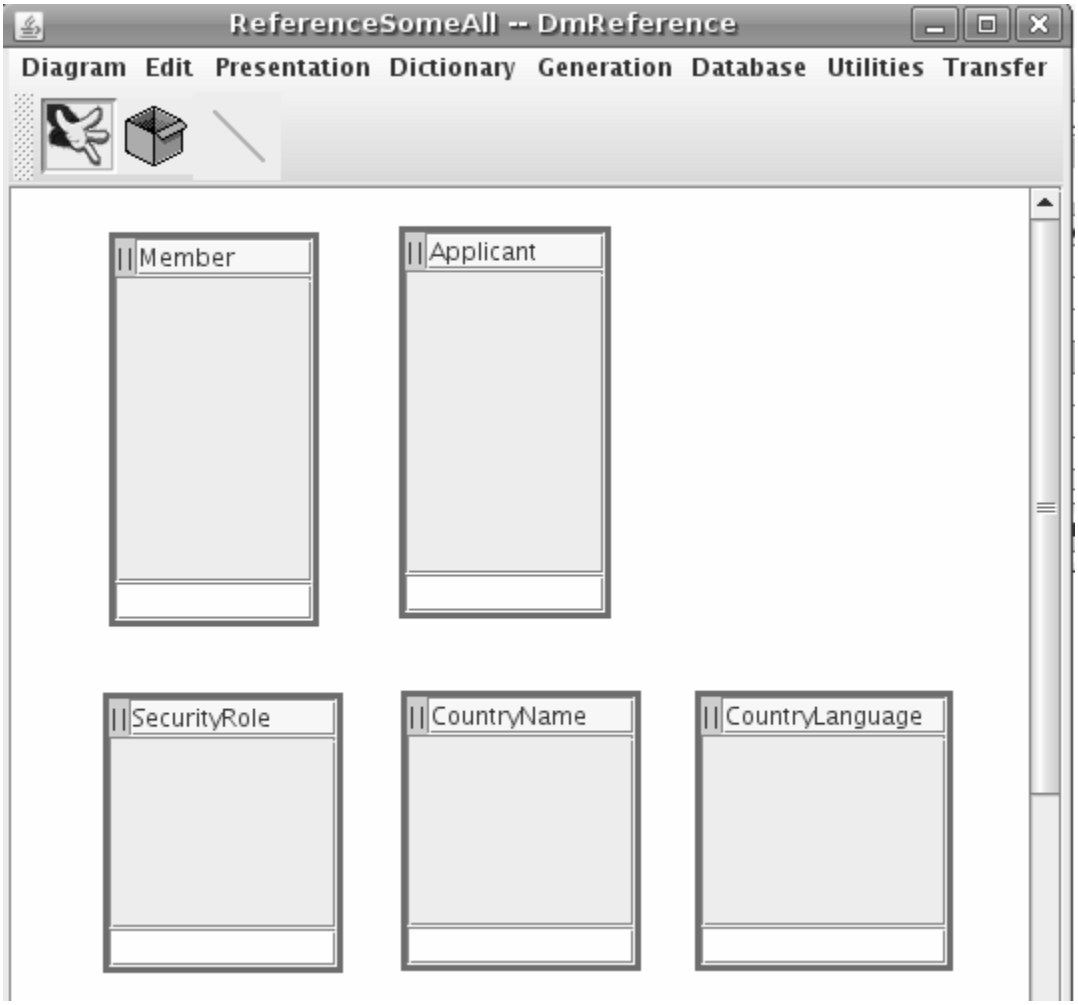

Figure 14.6. ReferenceSomeAll Model

The XML configuration shows that the model is not extended but its concepts are inherited. Semantically, there is no difference between the ReferenceAll and ReferenceSomeAll models.

```
<model oid="1199037039162">
      <code>ReferenceSomeAll</code>
      <author>Dzenan Ridjanovic</author>
      <persistenceType>xml</persistenceType>
      <persistenceRelativePath>
            data/xml/course/referencesomeall
      </persistenceRelativePath>
      <defaultLoadSave>true</defaultLoadSave>

      <concepts>

            <concept oid="1199037039164">
                  <code>Member</code>
                  <extension>true</extension>
                  <extensionDomain>Course</extensionDomain>
                  <extensionModel>DmReference</extensionModel>
                  <extensionConcept>Member</extensionConcept>
                  <entitiesCode>Members</entitiesCode>
```

```
        <entry>true</entry>

        <properties>
        </properties>

        <neighbors>
        </neighbors>

    </concept>

    <concept oid="1199037039166">
        <code>Applicant</code>
        <extension>true</extension>
        <extensionDomain>Course</extensionDomain>
        <extensionModel>DmReference</extensionModel>
        <extensionConcept>Applicant</extensionConcept>
        <entitiesCode>Applicants</entitiesCode>
        <entry>true</entry>

        <properties>
        </properties>

        <neighbors>
        </neighbors>

    </concept>

    <concept oid="1199037039167">
        <code>CountryName</code>
        <extension>true</extension>
        <extensionDomain>Course</extensionDomain>
        <extensionModel>DmReference</extensionModel>
        <extensionConcept>CountryName</extensionConcept>
        <entitiesCode>CountryNames</entitiesCode>
        <entry>true</entry>

        <properties>
        </properties>

        <neighbors>
        </neighbors>

    </concept>

    <concept oid="1199037039169">
        <code>CountryLanguage</code>
        <extension>true</extension>
        <extensionDomain>Course</extensionDomain>
        <extensionModel>DmReference</extensionModel>
        <extensionConcept>CountryLanguage</extensionConcept>
        <entitiesCode>CountryLanguages</entitiesCode>
        <entry>true</entry>
```

```
            <properties>
            </properties>

            <neighbors>
            </neighbors>

        </concept>

        <concept oid="1199037039171">
            <code>SecurityRole</code>
            <extension>true</extension>
            <extensionDomain>Course</extensionDomain>
            <extensionModel>DmReference</extensionModel>
            <extensionConcept>SecurityRole</extensionConcept>
            <entitiesCode>SecurityRoles</entitiesCode>
            <entry>true</entry>

            <properties>
            </properties>

            <neighbors>
            </neighbors>

        </concept>

    </concepts>

</model>
```

The ReferenceSome model is inherited from the DmReference model by extending some of its concepts. This is done first by selecting the DmReference model and second by clicking on the *Inherit Some* button in the main window. At the beginning there had been five concepts but two were deleted (Figure 14.7).

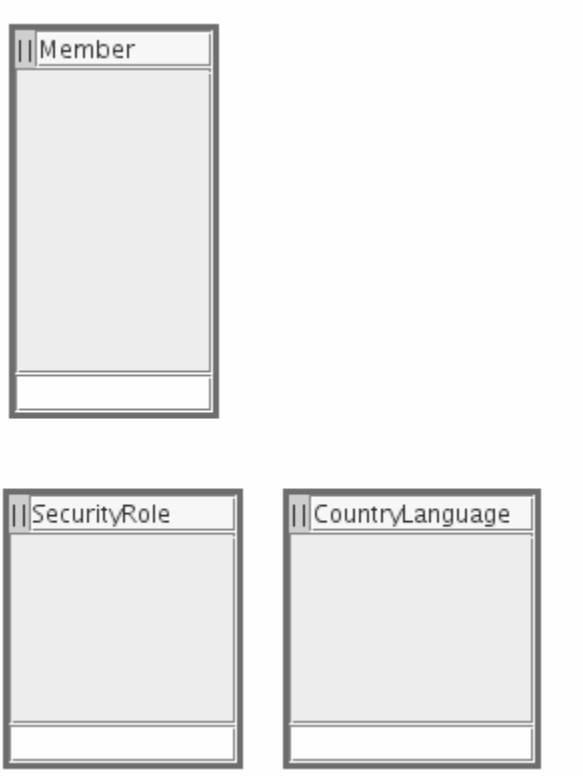

Figure 14.7. ReferenceSome Model

The XML configuration shows only the three inherited concepts.

```
<model oid="1199037275662">
      <code>ReferenceSome</code>
      <author>Dzenan Ridjanovic</author>
      <persistenceType>xml</persistenceType>
      <persistenceRelativePath>
            data/xml/course/referencesome
      </persistenceRelativePath>
      <defaultLoadSave>true</defaultLoadSave>

      <concepts>

            <concept oid="1199037275663">
                  <code>Member</code>
                  <extension>true</extension>
                  <extensionDomain>Course</extensionDomain>
                  <extensionModel>DmReference</extensionModel>
                  <extensionConcept>Member</extensionConcept>
                  <entitiesCode>Members</entitiesCode>
                  <entry>true</entry>

                  <properties>
                  </properties>

                  <neighbors>
                  </neighbors>
```

```
        </concept>

        <concept oid="1199037275668">
            <code>CountryLanguage</code>
            <extension>true</extension>
            <extensionDomain>Course</extensionDomain>
            <extensionModel>DmReference</extensionModel>
            <extensionConcept>CountryLanguage</extensionConcept>
            <entitiesCode>CountryLanguages</entitiesCode>
            <entry>true</entry>

            <properties>
            </properties>

            <neighbors>
            </neighbors>

        </concept>

        <concept oid="1199037275670">
            <code>SecurityRole</code>
            <extension>true</extension>
            <extensionDomain>Course</extensionDomain>
            <extensionModel>DmReference</extensionModel>
            <extensionConcept>SecurityRole</extensionConcept>
            <entitiesCode>SecurityRoles</entitiesCode>
            <entry>true</entry>

            <properties>
            </properties>

            <neighbors>
            </neighbors>

        </concept>

    </concepts>

</model>
```

The ReferenceSomePlus model is inherited from the DmReference model by extending some of its concepts and by specializing one of the concepts with a new property (Figure 14.8).

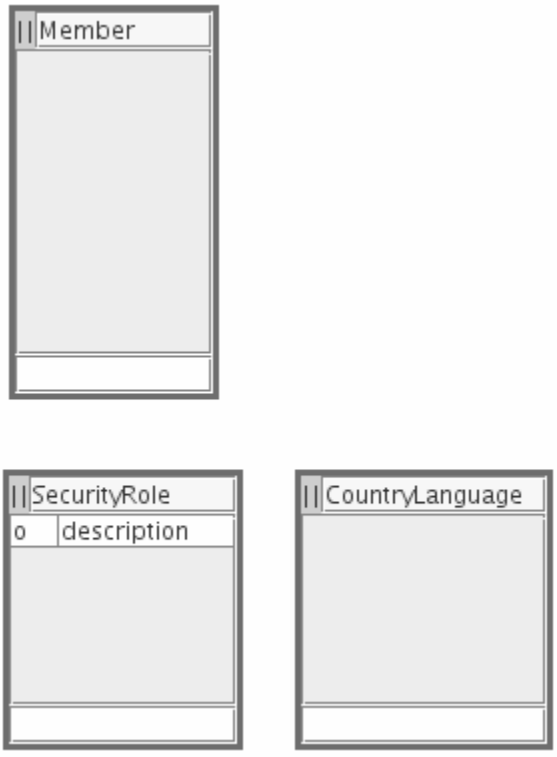

Figure 14.8. ReferenceSomePlus Model

The configuration for the SecurityRole concept includes the new description property.

```
<concept oid="1199037407444">
      <code>SecurityRole</code>
      <extension>true</extension>
      <extensionDomain>Course</extensionDomain>
      <extensionModel>DmReference</extensionModel>
      <extensionConcept>SecurityRole</extensionConcept>
      <entitiesCode>SecurityRoles</entitiesCode>
      <entry>true</entry>

      <properties>
            <property oid="1199037472710">
                  <code>description</code>
                  <propertyClass>
                        java.lang.String
                  </propertyClass>

                  <essential>false</essential>
            </property>
      </properties>

      <neighbors>
      </neighbors>

</concept>
```

The ReferenceSomePlusPlus model is specialized with a new concept and a new relationship (Figure 14.9).

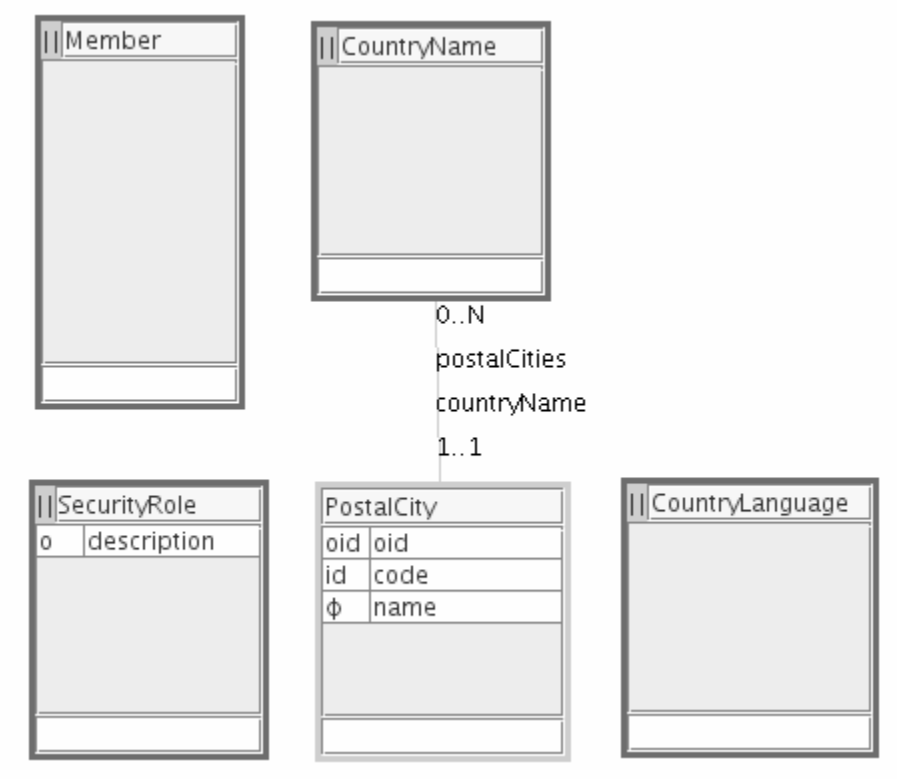

Figure 14.9. ReferenceSomePlus Model

The XML configuration presents the model with extended and specialized declarations.

```
<model oid="1199038773991">
      <code>ReferenceSomePlusPlus</code>
      <author>Dzenan Ridjanovic</author>
      <persistenceType>xml</persistenceType>
      <persistenceRelativePath>
            data/xml/course/referencesomeplusplus
      </persistenceRelativePath>
      <defaultLoadSave>true</defaultLoadSave>

      <concepts>

            <concept oid="1199038773992">
                  <code>Member</code>
                  <extension>true</extension>
                  <extensionDomain>Course</extensionDomain>
                  <extensionModel>DmReference</extensionModel>
                  <extensionConcept>Member</extensionConcept>
                  <entitiesCode>Members</entitiesCode>
                  <entry>true</entry>

                  <properties>
                  </properties>
```

```
      <neighbors>
      </neighbors>

</concept>

<concept oid="1199038773995">
      <code>CountryName</code>
      <extension>true</extension>
      <extensionDomain>Course</extensionDomain>
      <extensionModel>DmReference</extensionModel>
      <extensionConcept>CountryName</extensionConcept>
      <entitiesCode>CountryNames</entitiesCode>
      <entry>true</entry>

      <properties>
      </properties>

      <neighbors>
            <neighbor oid="1199038929599">
                  <code>postalCities</code>
                  <destinationConcept>
                        PostalCity
                  </destinationConcept>
                  <inverseNeighbor>
                        countryName
                  </inverseNeighbor>
                  <internal>true</internal>
                  <partOfManyToMany>false</partOfManyToMany>
                  <type>child</type>
                  <min>0</min>
                  <max>N</max>
            </neighbor>
      </neighbors>

</concept>

<concept oid="1199038773997">
      <code>CountryLanguage</code>
      <extension>true</extension>
      <extensionDomain>Course</extensionDomain>
      <extensionModel>DmReference</extensionModel>
      <extensionConcept>CountryLanguage</extensionConcept>
      <entitiesCode>CountryLanguages</entitiesCode>
      <entry>true</entry>

      <properties>
      </properties>

      <neighbors>
      </neighbors>
```

```
</concept>

<concept oid="1199038773999">
      <code>SecurityRole</code>
      <extension>true</extension>
      <extensionDomain>Course</extensionDomain>
      <extensionModel>DmReference</extensionModel>
      <extensionConcept>SecurityRole</extensionConcept>
      <entitiesCode>SecurityRoles</entitiesCode>
      <entry>true</entry>

      <properties>
            <property oid="1199038852482">
                  <code>description</code>
                  <propertyClass>
                        java.lang.String
                  </propertyClass>

                  <essential>false</essential>
            </property>
      </properties>

      <neighbors>
      </neighbors>

</concept>

<concept oid="1199038869526">
      <code>PostalCity</code>
      <entitiesCode>PostalCities</entitiesCode>

      <properties>
            <property oid="1199038888951">
                  <code>code</code>
                  <propertyClass>
                        java.lang.String
                  </propertyClass>
                  <required>true</required>
                  <unique>true</unique>

                  <essential>true</essential>
            </property>
            <property oid="1199038892105">
                  <code>name</code>
                  <propertyClass>
                        java.lang.String
                  </propertyClass>
                  <required>true</required>

                  <essential>true</essential>
            </property>
      </properties>
```

```
            <neighbors>
                  <neighbor oid="1199038929599">
                        <code>countryName</code>
                        <destinationConcept>
                              CountryName
                        </destinationConcept>
                        <inverseNeighbor>
                              postalCities
                        </inverseNeighbor>
                        <internal>true</internal>
                        <partOfManyToMany>false</partOfManyToMany>
                        <type>parent</type>
                        <min>1</min>
                        <max>1</max>
                  </neighbor>
            </neighbors>

      </concept>

   </concepts>

</model>
```

The abstract DmReference model appears in the same domain as other models. Thus, all generated configurations are located in the same reusable-domain-config.xml file. The specific-domain-config.xml file is left for declarations specific to the current project. Thus, the reusable configuration may indeed be reused in a different project. Some of abstract models may be placed in the modelibra-domain-config.xml file so that all users of Modelibra may benefit from them.

When designing a model, a concept may inherit from an abstract concept from the same model. For example, in the meta model of ModelibraModeler, there are two abstract concepts: Entity and Element. Inheritance relationships have the isA name. Modelibra does not support super or generic concepts that are not abstract.

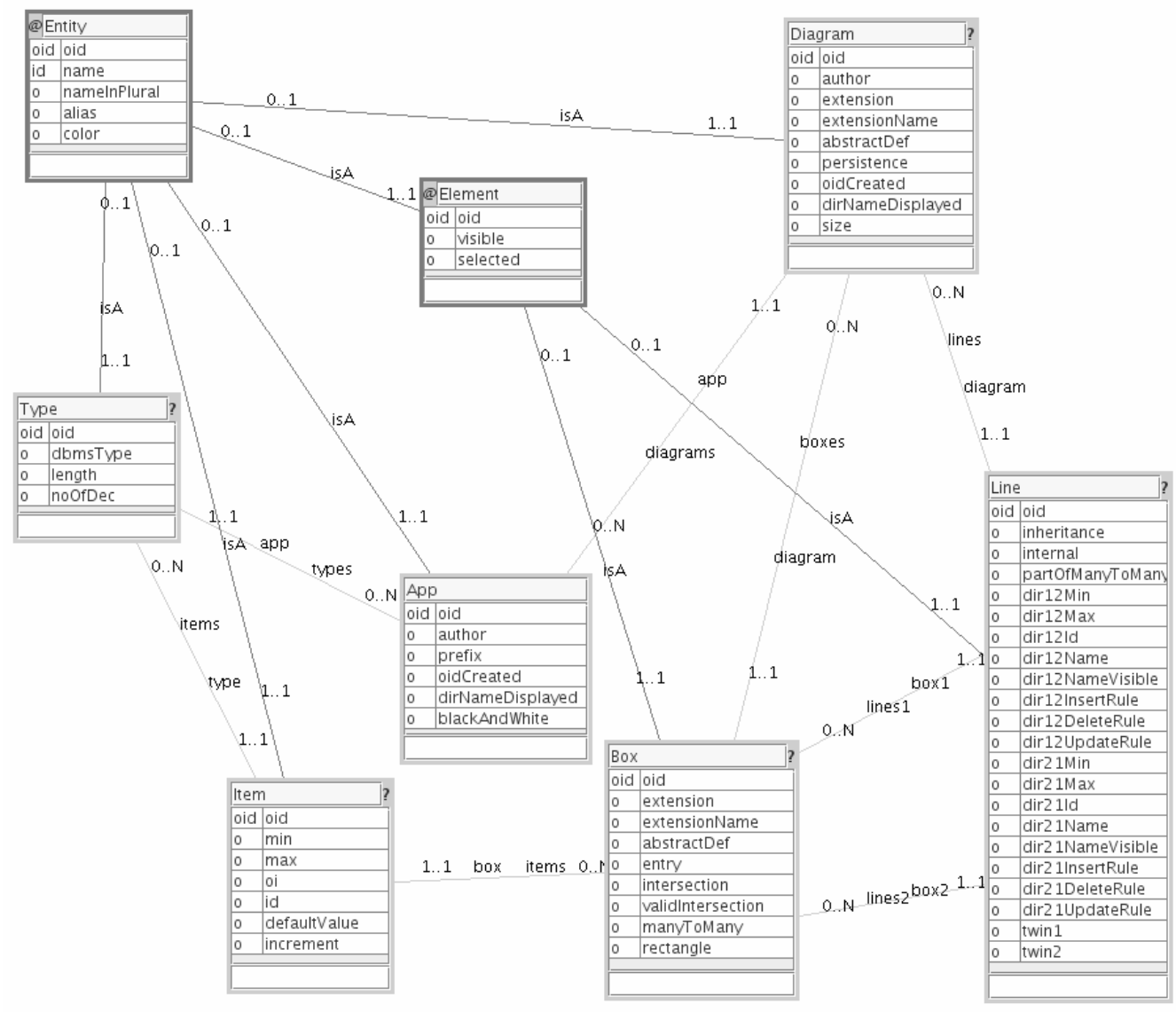

Figure 14.10. ModelibraModeler Meta Model

The *Copy* button from the main window may be used to copy a large model several times and keep only those concepts and relationships pertinent for that partition of the model. For example the meta model of Modelibra may be partitioned in two meta models, one with the Domain, Model and Concept concepts, and Domain--Model and Model--Concept relationships (Figure 14.11) and the other with the Concept, Property and Neighbor concepts, and Concept—Property and Concept—Neighbor relationships (Figure 14.12).

The same technique may be used to keep different spirals of the same model in the same .mm file.

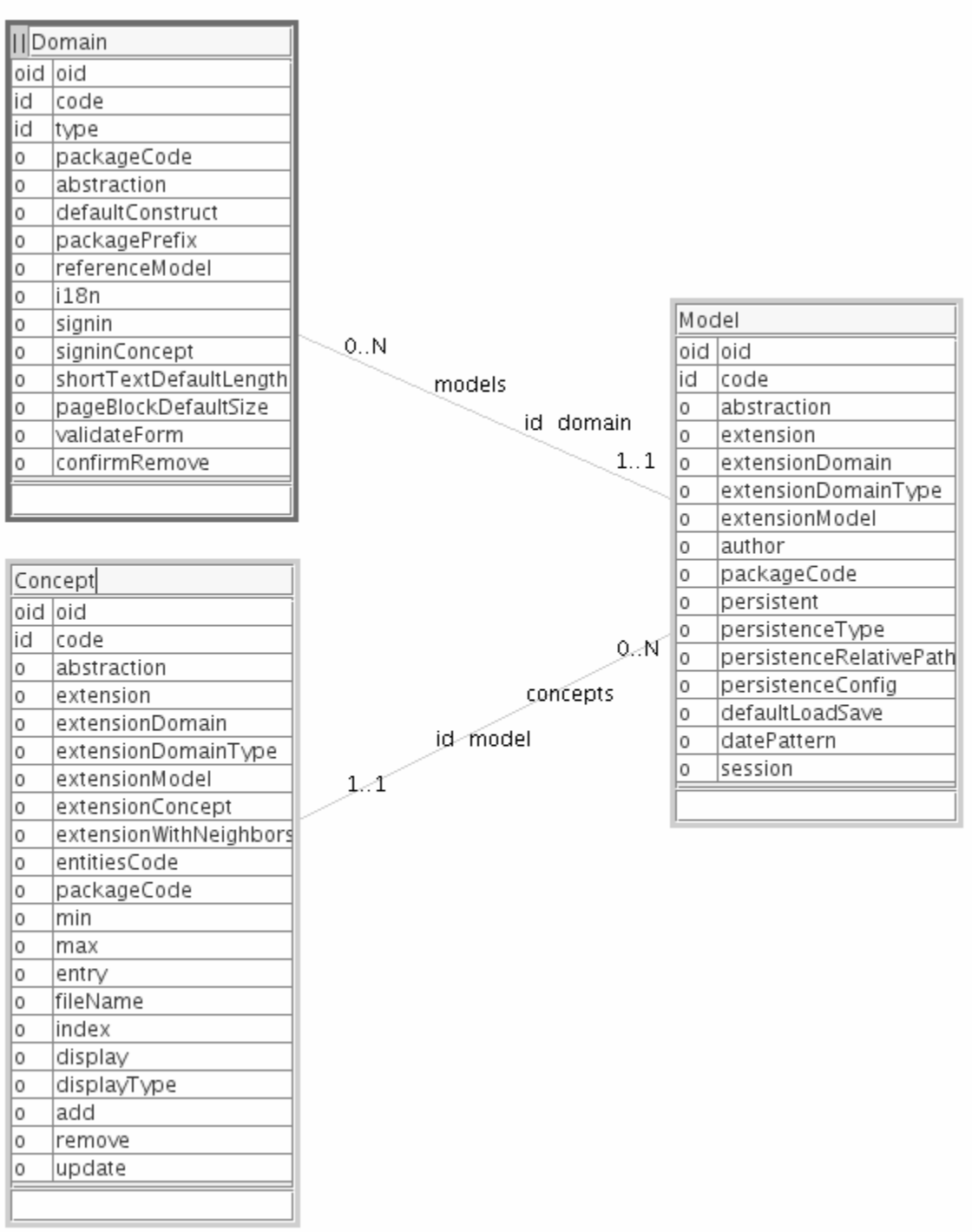

Figure 14.11. Meta Model of Modelibra with Domain, DomainModel and Concept concepts.

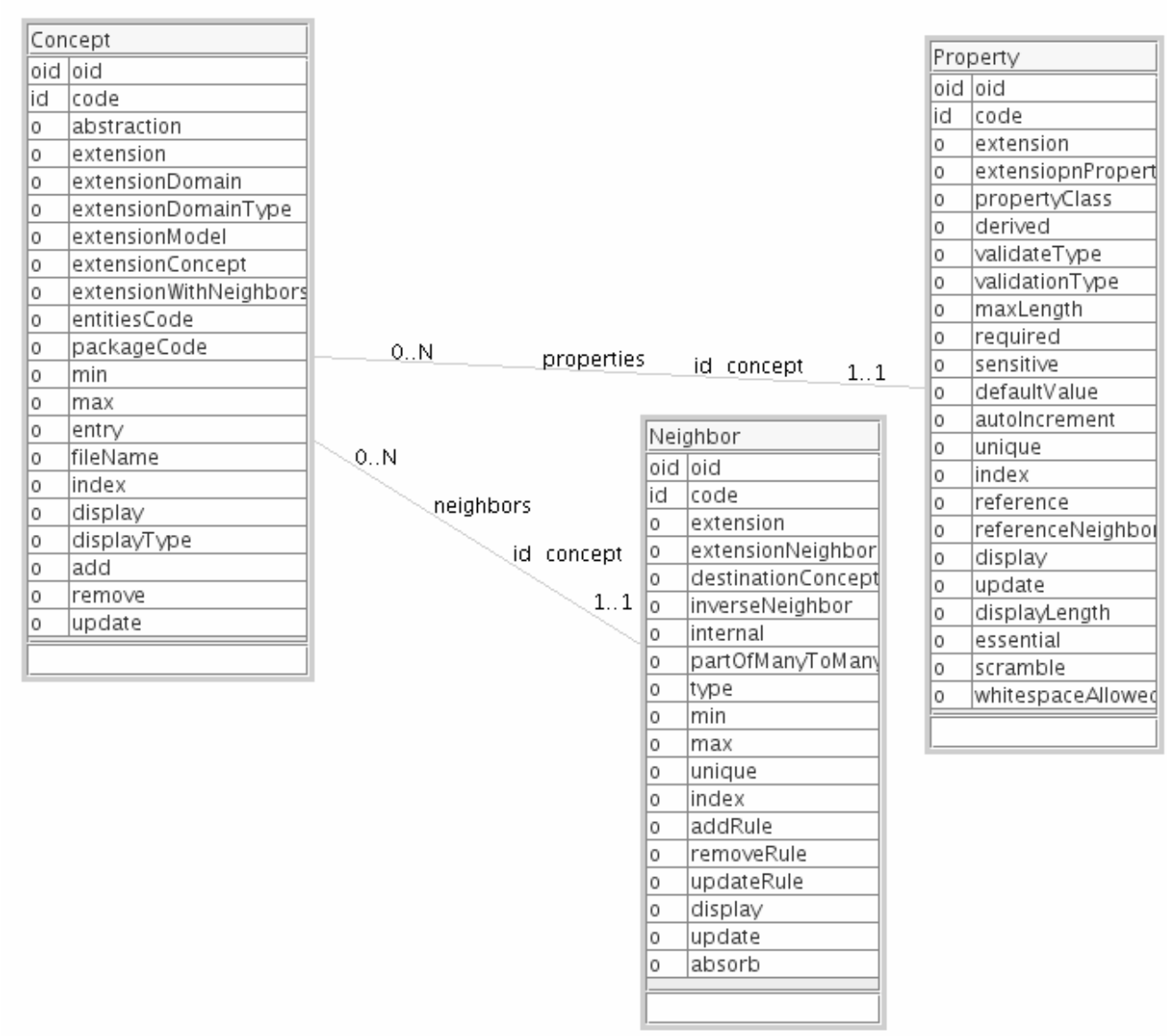

Figure 14.12. Meta Model of Modelibra with Concept , Property and Neighbor concepts.

Summary

The CourseQuiz application is introduced to show the use of inheritance at the level of models, XML configurations and HTML declarations. The web application provides formative quizzes so that a user may test her knowledge of a certain subject matter. There are no grades and the answers are not recorded.

The inheritance used in Modelibra and Wicket, outside the inheritance found in Java, is explained. In Modelibra, an XML configuration may inherit another XML configuration. In Wicket, a panel's HTML code may extend another panel's HTML code. An inheritance of models in ModelibraModeler is also presented. By using an inheritance of models, XML configurations, Java classes and XML code, in addition to component decomposition, and the code generation, the development productivity may increase dramatically.

Questions

1. What can be inherited in XML configurations?

2. Can an HTML code of a web page be extended in Wicket?

3. Compare the inheritance of Models in ModelibraModeler and the inheritance of Java classes in Java.

Exercises

Exercise 14.1.

Inherit the Member concept from the Reference model in the Dm domain. Add relationships as specific neighbors.

Exercise 14.2.

Add the CountryName concept to the Reference model. Apply inheritance in XML configurations, Java classes and HTML code. Add address properties (street, city, postal code, country) as specific definitions to the Member concept. Use country names to validate the country of a member.

Exercise 14.3.

Compare the CourseQuiz web application with one found at [ClassMarker]. Add a feature from ClassMarker that is not present in the CourseQuiz application.

Web Links

[ClassMarker] ClassMarker
http://www.classmarker.com/

[Inheritance] Inheritance
http://en.wikipedia.org/wiki/Inheritance_(computer_science)