Adding a metamodel

A metamodel is supported in MDWorkbench using EMF (http://www.eclipse.org/emf). A metamodel may be imported from various sources:

- EMF Ecore files (`*.ecore`): the core EMF framework includes a MOF-like meta metamodel (Ecore) for describing metamodels.
- KM3 files (`*.km3`): KM3 (Kernel Meta Meta Model), defined at INRIA, is a neutral language to write metamodels. The reference manual may be found at: http://www.eclipse.org/m2m/atl/
- UML 1.3 XMI files (`*.xmi`): using the class diagram, it is possible to define metamodels with UML.
- XML Schema files (`*.xsd` or `*.wsdl`): XML Schema Infoset Model (XSD) is a library that provides an API for manipulating the components of an XML Schema as described by the W3C XML Schema specifications. It provides tools (Generating an EMF Model using XML Schema) , used by MDWorkbench to import metamodels from XML schemas.

Importing a metamodel results in the creation of a new Eclipse plugin project, which contains Java classes that define the contents of the metamodel.

You may then add profile capability to the imported metamodel, if it makes sense.

The metamodel can then be tested and deployed.

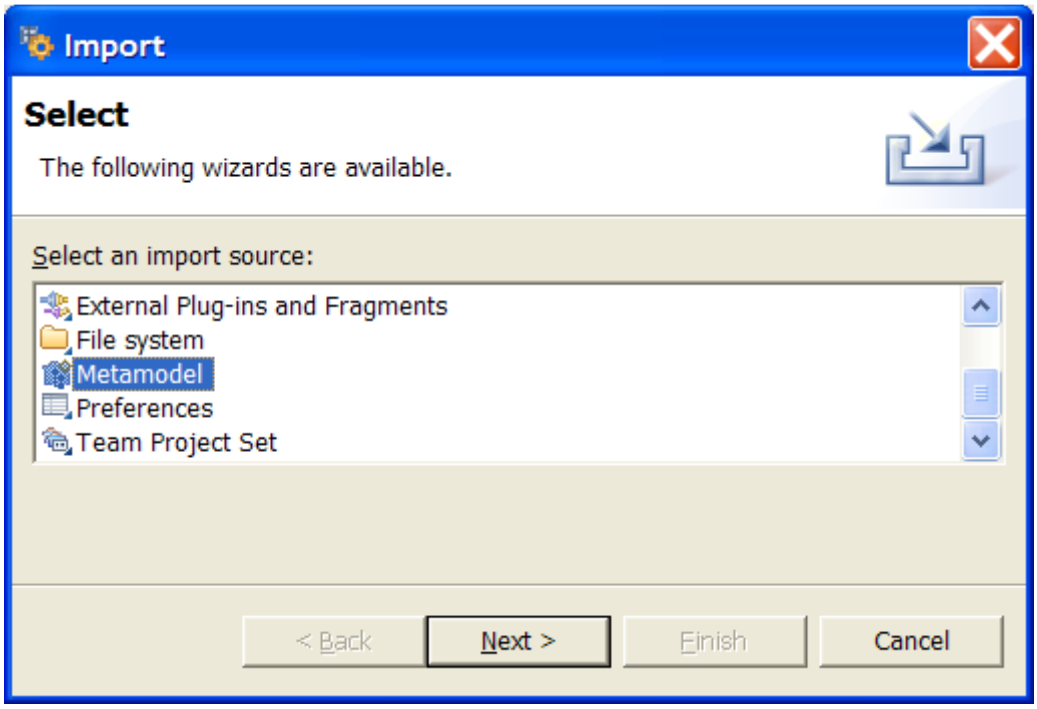
Compatibility note

Metamodels are EMF-based (http://www.eclipse.org/emf). Please note that EMF is ascending compatible only: metamodels created and compiled with a certain version of Eclipse are not compatible with lower versions.

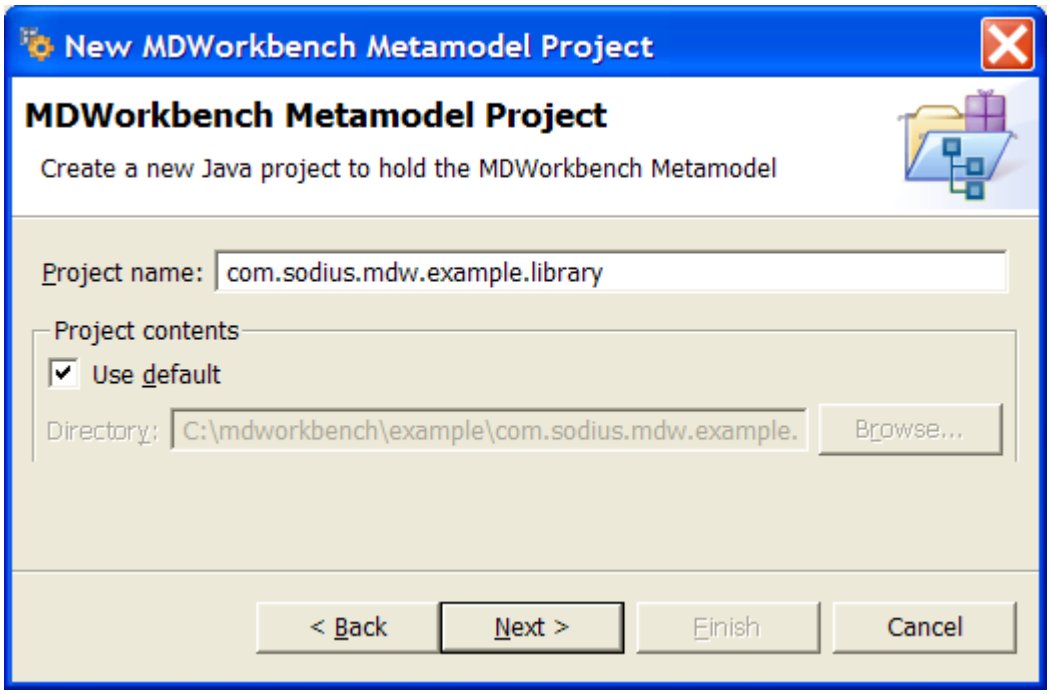
Importing a metamodel

To import a metamodel:

1. Click **File > Import**.
2. Select the **Metamodel** item and click **Next**.



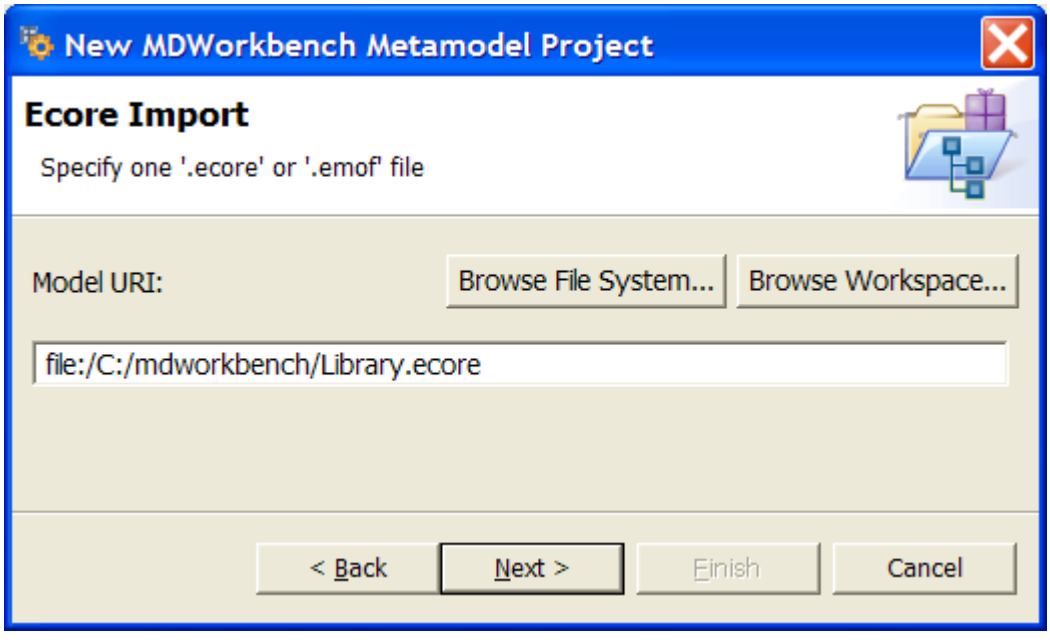
3. Type the name of the plugin project to create (e.g. **`com.sodius.mdw.example.library`**) in the **Project name** field, and click **Next**



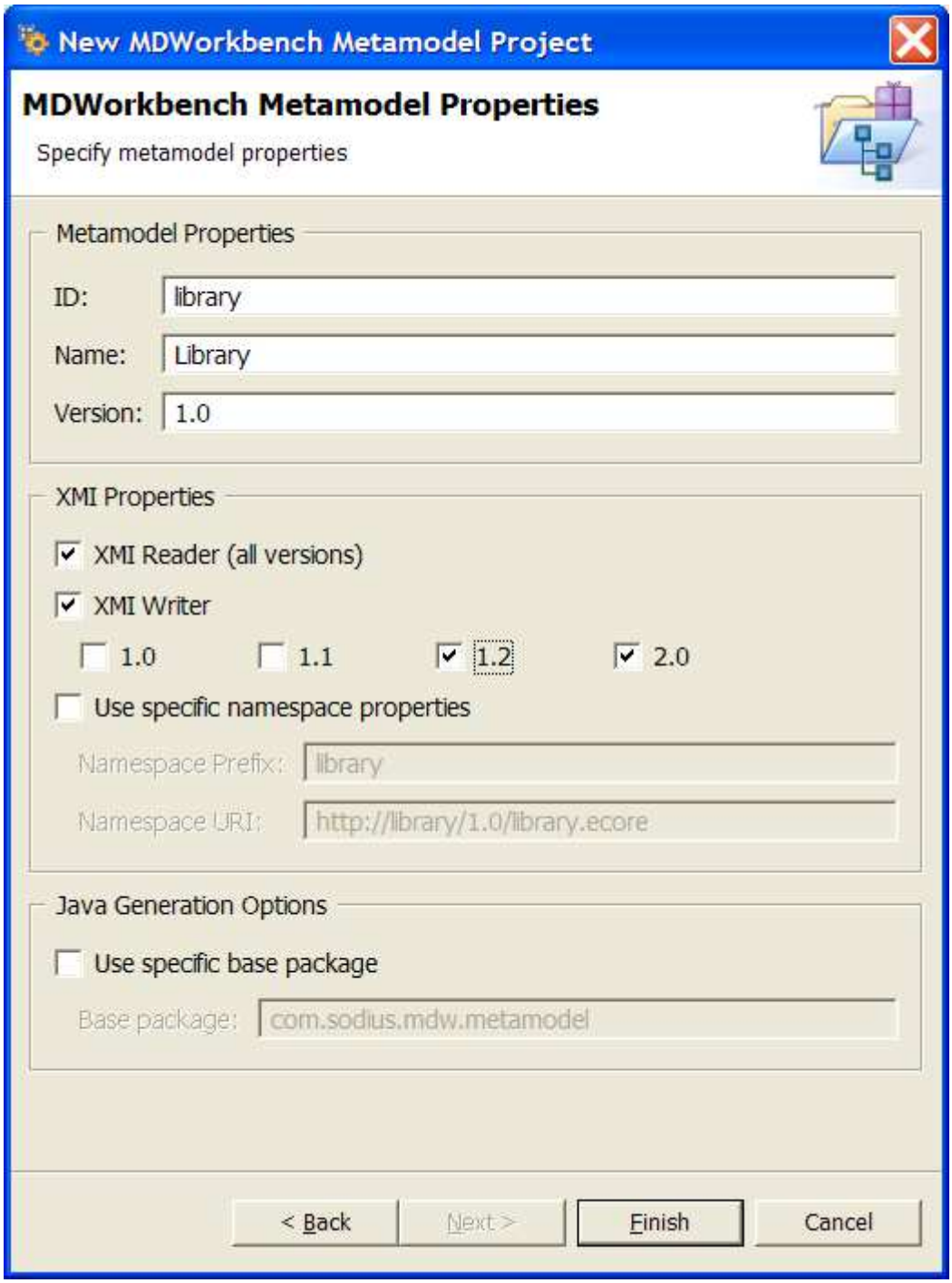
4. Choose a type of metamodel importer (e.g. **Ecore Importer)**:
   - Ecore: handles EMF metamodels (`*.ecore`).
   - KM3: handles KM3 metamodels (`*.km3`).
   - UML 1.3: handles metamodels defined in a UML 1.3 XMI file (`*.xmi`).
   - UML 2.1: handles metamodels defined in a UML 2.1 XMI file (`*.xmi` and `*.uml`).
   - XML Schema: converts an XML schema to a metamodel (`*.xsd` or `*.wsdl`).

   Click **Next**.

5. Type the file which contains the metamodel data to read and click **Next**.



6. Type the required metamodel information and click **Finish**.

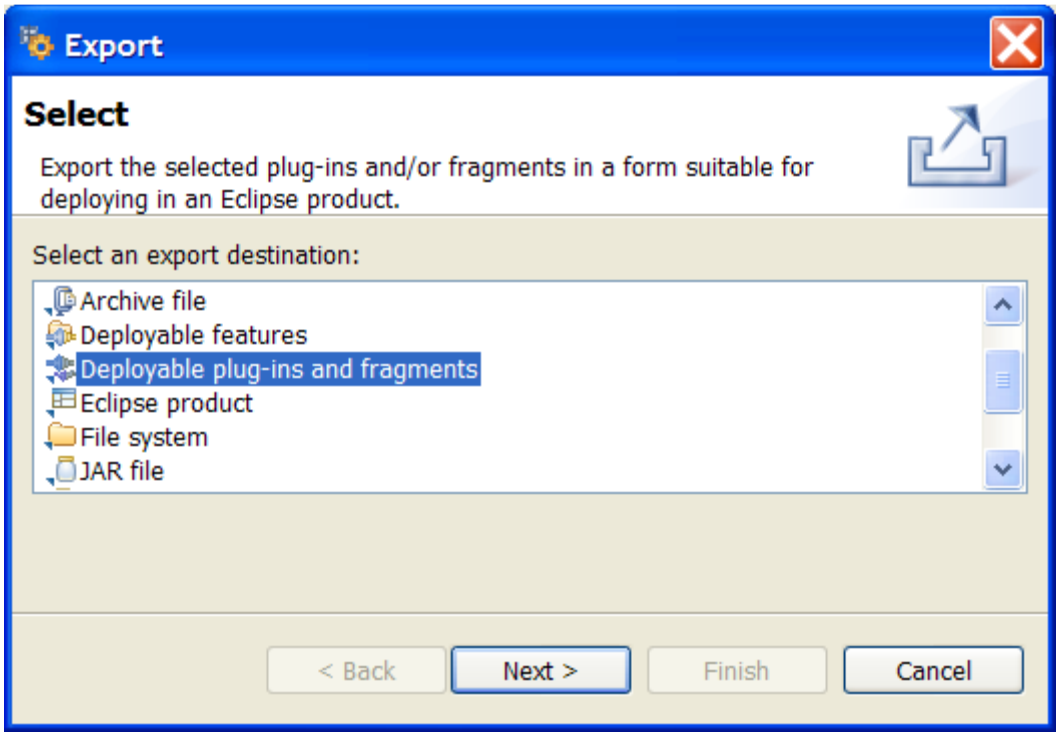


MDWorkbench analysizes the metamodel and generates a set of Java classes that defines the contents of the metamodel.

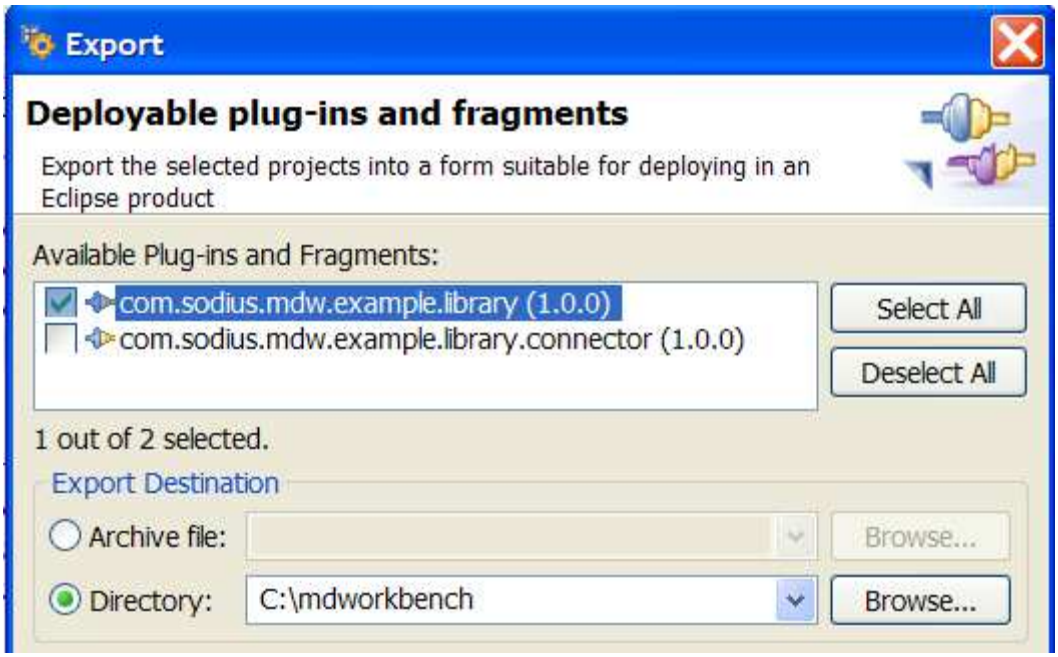
Deploying a metamodel

To deploy a metamodel:

1. Click **File > Export...**.
2. Select **Deployable Plug-ins and fragments** and click **Next**.



3. Select your plugin (e.g. **com.sodius.mdw.example.library**) in the plugin list.
4. Select **Directory**, in the **Export destination** group.
5. Type the path of an MDWorkbench installation base directory and click **Finish**.
6. Deployed procedure in Ensta Bretagne is changed :
    1. The directory to export the metamodel is c:\Temp\eclipse\plugins
    2. Id the directory eclipse\plugins doesn’t exist, please create it.



7. Restart MDWorkbench, the metamodel is now available in the environment.

Compatibility note

Metamodels are EMF-based (http://www.eclipse.org/emf). Please note that EMF is ascending compatible only: metamodels created and compiled with a certain version of Eclipse are not compatible with lower versions.

Before the point 6, verify if the plugin is correctly copied in the Eclipse\plugins folder. If you have store the file of you plugin in another folder, copy the plugin file in the Eclipse\plugins.