Scarab - Developer's Guide

Scarab for Scarab developers

The Scarab Development Team

Scarab - Developer's Guide: Scarab for Scarab developers

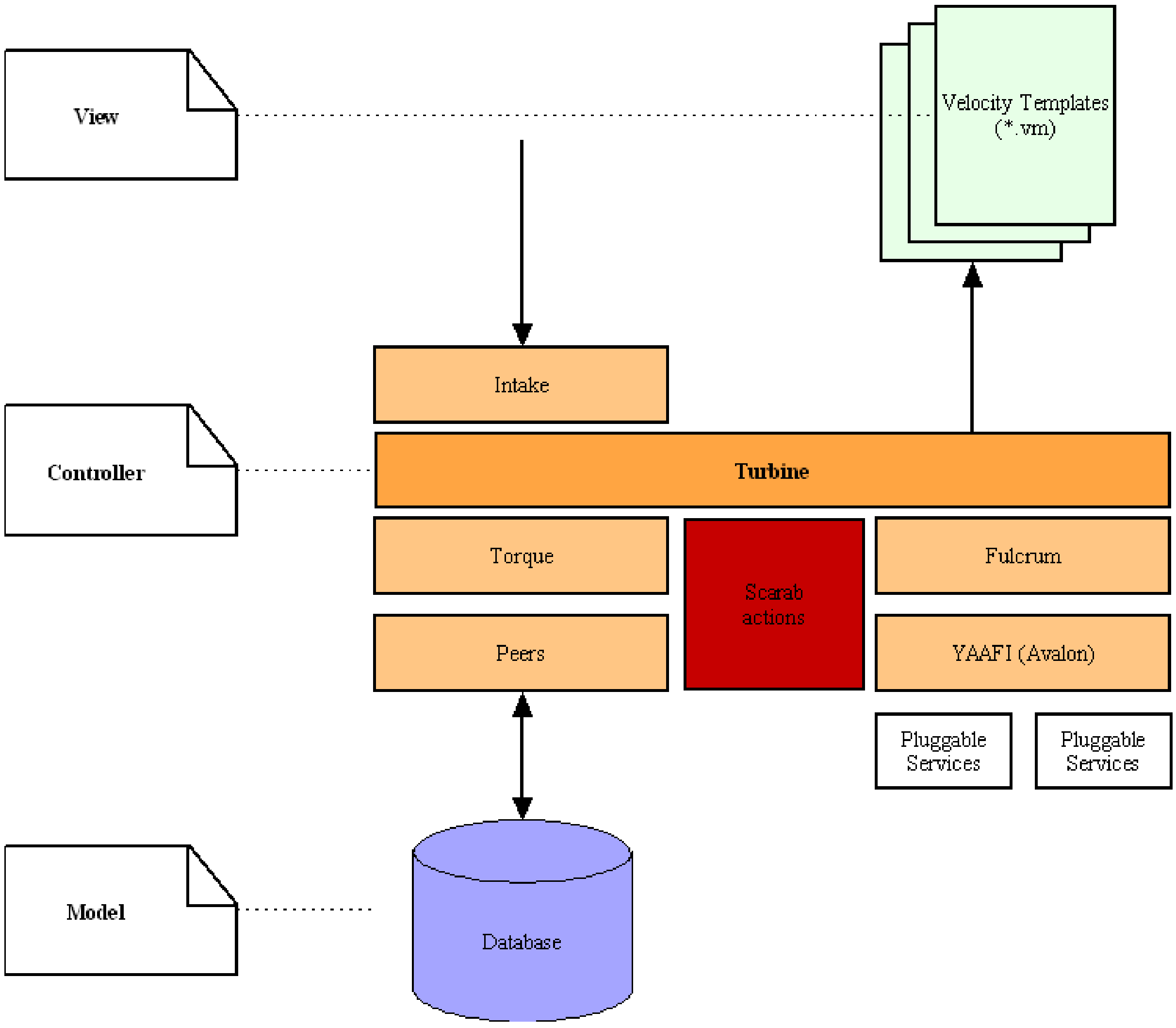
The Scarab Development Team

Table of Contents

List of Tables

Chapter 1. Scarab architecture

Scarab: a bird's eye view



An MVC web application based on the Turbine framework

A web application

Scarab is a web application that uses the Java servlet technology..

MVC Architecture

Regarding its global architecture, Scarab uses the MVC (Model-View-Controller) design pattern. The part that interacts with the user (View) is decoupled from the data defining the application state (Model) by a Controller that managing the navigation in Scarab.

The Turbine framework

A framework is a way of building applications, plus tools to help building them. The Turbine framework (http://jakarta.apache.org/trubine [http://jakarta.apache.org/trubine/]) is a doctrine to program web applications according to the MVC design pattern. Turbine also brings quite a set of tools (components, projects) to help code your application. Some of these components are detailed below, as they are used by Scarab.

The main Scarab components

Intake

The form data sent by the user may be submitted to a validation process (required fields, well-formed answers,...). This preliminary check is performed by a Turbine component called Intake.

Intake is now a part of Fulcrum (see below).

Note

Intake plays in Turbine the same role as the Struts Validator in Struts, if you're familiar with this well-known framework.

In the Scarab distribution

Intake validation rules are expressed in XML in the *`src/conf/intake.xml`* file.

On the Internet

The Intake documentation is available at this URL: http://jakarta.apache.org/turbine/fulcrum/fulcrum-intake [http://jakarta.apache.org/turbine/fulcrum/fulcrum-intake/].

Turbine (in the strict sense of the word)

It's the heart of the framework. As in others MVC2 frameworks, there is only one entry point in the application, the *`scarab`* servlet, that is an instance of the *`org.apache.turbine.Turbine`* class.

The behaviour of this servlet is ruled by five properties files, declared in the `TurbineConfiguration.xml` deployment descriptor :

1. `custom.properties`
2. `defaultCustom.properties`
3. `PermissionMapping.properties`
4. `TurbineResources.properties`
5. `Torque.properties`

Note

Which version of Turbine ?

If you know Turbine or if you have read the documentation on the Jakarta web site, you have probably realized that the architecture of the different versions of Turbine (2.3 and 2.4 are currently available) is quite different.

In fact, Scarab has been developed initially by members of the Turbine project and it used a pre-version of Turbine 3 -- that will probably never be released. So the current development team spends some energy trying to bring Scarab back in the current Turbine development flow by "downgrading" the version of some of the components and integrating some of the Turbine 3 evolutions back in Turbine 2.4.

In the Scarab distribution

Configuration files mentioned above are in the *`src/conf/conf/`* directory of the Scarab distribution, and under *`WEB-INF/conf`* in the web application.

Turbine itself is a jar in `www/repository/turbine/jars` (and the corresponding source files in `www/repository/turbine/src`). As explained above, it is an intermediary release to which you probably cannot sustitute another tagged release.

On the Internet

The Turbine project documentation is available at this URL: http://jakarta.apache.org/turbine [http://jakarta.apache.org/turbine/].

Fulcrum: the services

The Turbine architecture in 2.4= versions is based on the concept of services and uses a microcontainer of the Avalon family. Some of these services (among which Intake, mentioned earlier) are part of Fulcrum and used by Scarab.

In the Scarab distribution

The Fulcrum archives (a dozen of jars) are in `www/repository/fulcrum/jars`

On the Internet

The Fulcrum project documentation is available at this URL: http://jakarta.apache.org/turbine/fulcrum/

YAAFI (Avalon): the microcontainer for Fulcrum services

The services used by Scarab are implemented on top of a microcontainer, member of the Avalon family. (By the way, YAAFI is just an acronym for *Yet Another Avalon Framework Implementation*.)

In the Scarab distribution

YAAFI is in `www/repository/fulcrum/jars/fulcrum-yaafi-1.0.3.jar`

On the Internet

The YAAFI project documentation and its interface with Fulcrum are available at this URL: http://jakarta.apache.org/turbine/fulcrum/fulcrum-yaafi [http://jakarta.apache.org/turbine/fulcrum/fulcrum-yaafi/].

O/R mapping: Torque and the *Peer* classes

Scarab Java objects are populated from, and persisted to, a relational database (RDBMS) using a software component that was once part of Turbine but became independent since: Torque (Torque now belongs to the Apache database project).

From an XML high-level description of the database schema, Torque can generate:

1. database schema creation scripts (SQL/DDL), for various RDBMS;
2. Java classes that map the various database tables to JavaBeans, with finders and accessor methods.

If you are interested in further details about the database schema, read chapter 3.

In the Scarab distribution

The three XML files describing the database schema are in the `src/schema` of the Scarab distribution.

On the Internet

The documentation of the Torque project is available at this URL: http://db.apache.org/torque [http://db.apache.org/torque/]

The View: Velocity

Turbine can use JSP (JavaServer Pages) but the first Scarab developers chose to use the Velocity technology for the dynamic pages sent to the user.

Velocity is a template language: at runtime, the values of JavaBeans placed in the Velocity "Context" are inserted in a predefined (static) content.

There are two differences with the JSP technology that are worth a mention here, that may well have been the motives of the initial Scarab development team:

1. first, there is a point about architecture. There are control structures in Velocity (such as conditions, loops, etc.) but it is not possible to insert Java code in a template, there is no way to insert the so-called "scriptlets". This should make the maintenance easier in the long term because the view and the controller are completely decoupled from each other.
2. in practice, the JSP can only be used to generate web pages (HTML or XML); Velocity is a more general templating language and can be used to generate mails, SQL scripts, PostScript and quite a lot of other funny things.

In the Scarab distribution

The Velocity templates for the Scarab web pages (and emails, etc.) are under `src/webapp/WEB-INF/templates`, grouped by functionality.

On the Internet

The documentation of the Velocity project is available at this URL: http://jakarta.apache.org/velocity [http://jakarta.apache.org/velocity/].

Chapter 2. Understanding the Ant build

Abstract

The Scarab application currently may be built by the end user using Ant. Ant is used to create and initialize the database.

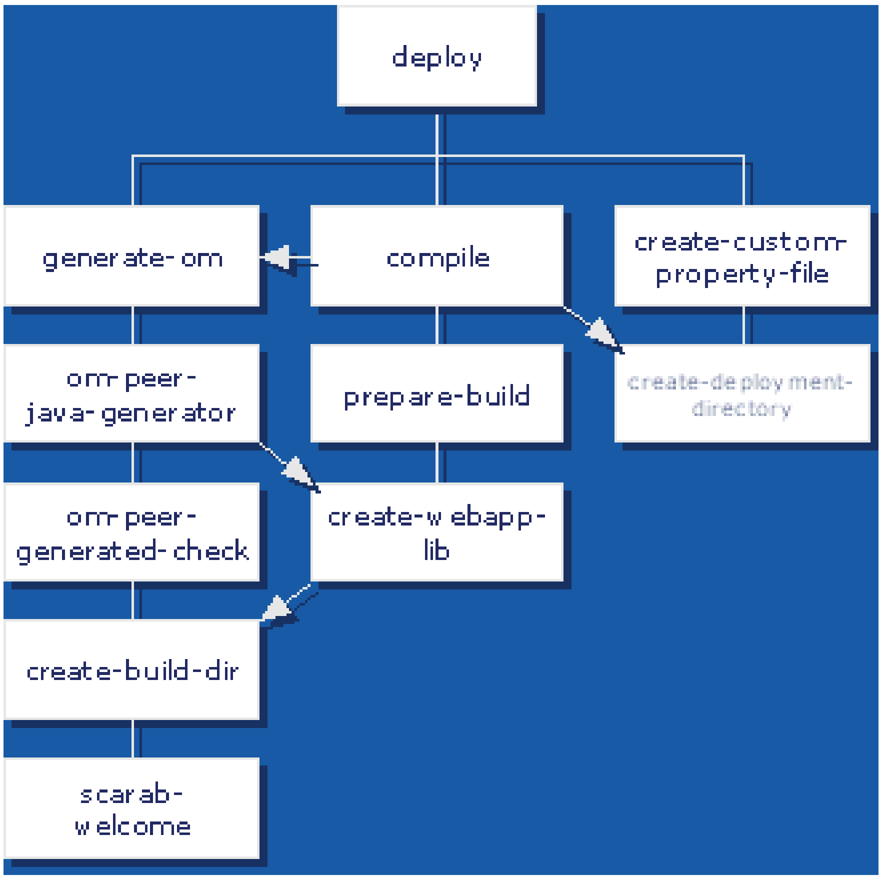
$SCARAB_ROOT/build/build.xml

This is the main build file.

The default target, deploy, is used to generate the code of the OM (Object Mapping) classes and to compile and build the Scarab application.

The second main target, create-db, is used to create and initialize the Scarab database. It does this by calling the second ant build file, described in the second section.

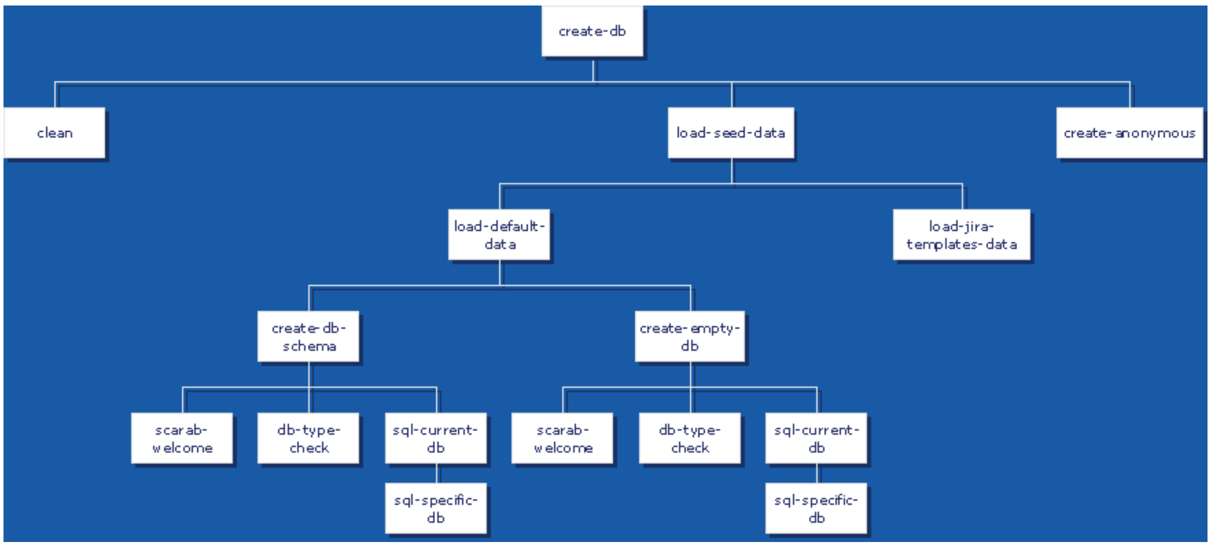
The target dependency graph below may help you visualize how targets are related to each other and the order in which they are called.



$SCARAB_ROOT/src/conf/conf/build.xml

This build file is used to create and populate the Scarab database.

The target dependency graph below may help you visualize how targets are related to each other and the order in which they are called.


create-db
clean
load-seed-data
create-anonymous
load-default-data
load-jira-templates-data
create-db-schema
create-empty-db
scarab-welcome
db-type-check
sql-current-db
scarab-welcome
db-type-check
sql-current-db
sql-specific-db
sql-specific-db

Chapter 3. Maven

Abstract

Though Scarab end-users are more likely to use Ant to build Scarab, as a developer you will need Maven to access all development functionalities, among which documentation generation.

Maven 1

Though Maven 2 is getting momentum and has officially become the mainstream version at the end of 2005, Scarab still uses Maven 1.

Maven 1 and Maven 2 are functionnally equivalent but they are not syntactically compatible and use different build files.

So you will need to install Maven 1 and refer to the Maven 1 documentation at this URL: http://maven.apache.org/maven-1.x/

Building Scarab with Maven

Normal (full) build

Go to the `$SCARAB_ROOT` directory.

Launch Maven:

maven war

Quick build

Go to the `$SCARAB_ROOT` directory.

Launch Maven:

maven war -Dmaven.test.skip

`-Dmaven.test.skip` skips the execution of unit tests. Scarab unit tests have been designed to be executed against a database; if yours is not (yet) configured, they will most certainly fail.

Running the tests

Just launch:

```
maven test
```

The results of the tests will be in `$(SCARAB_HOME)/target/test-results/`

If you'd like to make your tests with another db environment, follow these steps:

1. `maven clean` (you better start from scratch if you gonna test)
2. Configure your `build.properties` to contain at least the correct `scarab.database.type` value.
3. maven war
4. maven scarab:create-db (make sure you choose the same database than before!!)

5. maven test

Chapter 4. Scarab development with Eclipse

Getting the Scarab sources

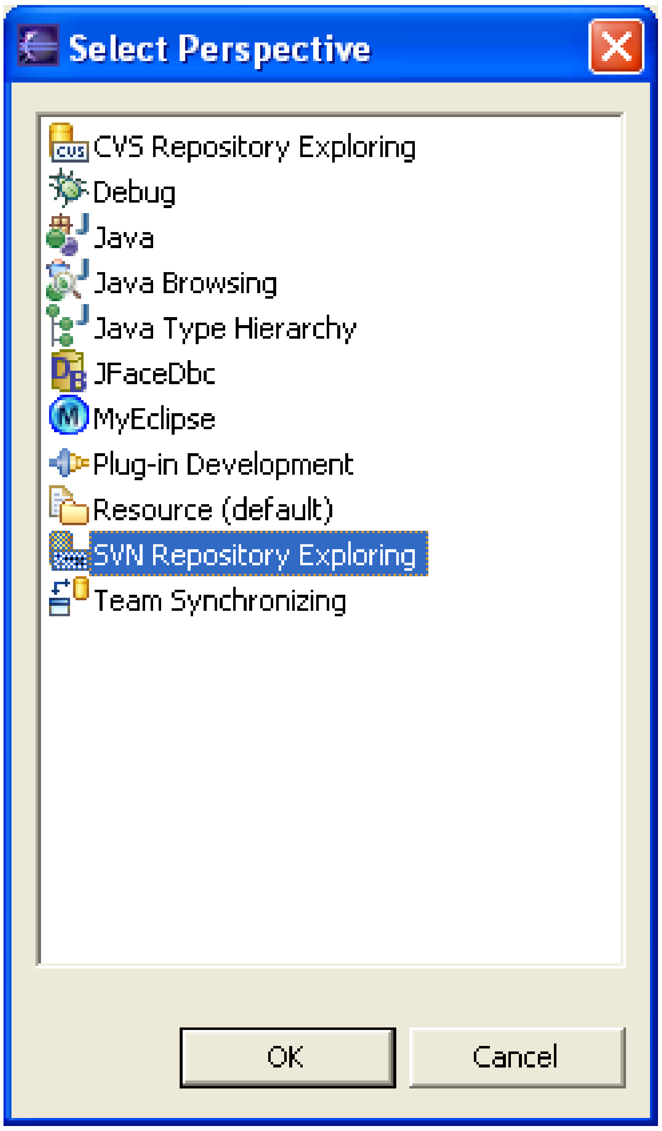
Getting the subclipse plugin

You will need the Subclipse plugin to connect to the Subversion repository that hosts and manages the Scarab source code base.

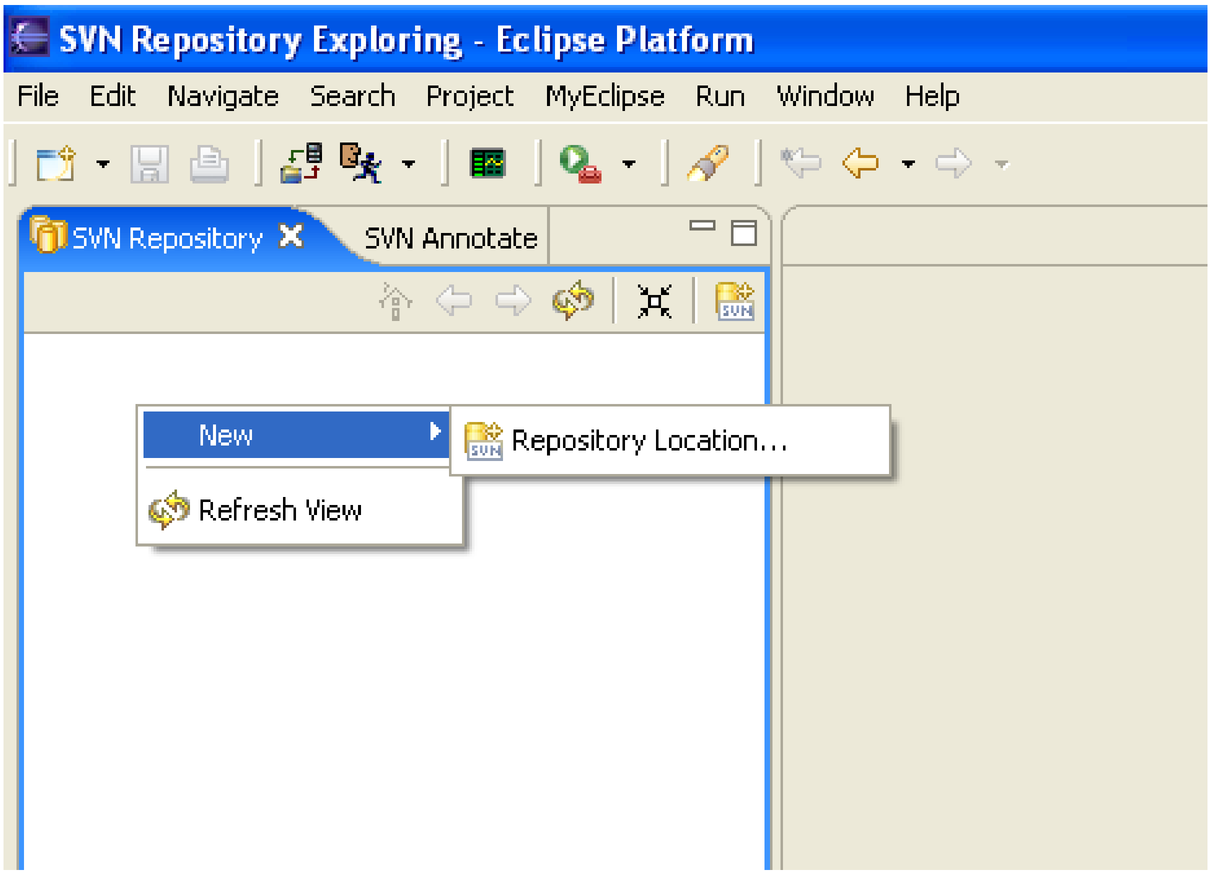
If you do not have this plugin already installed, you will find detailed installation instructions at this URL : http://subclipse.tigris.org/install.html

Connecting to the Scarab Subversion repository

Select the 'SVN Repository Exploring' perspective :



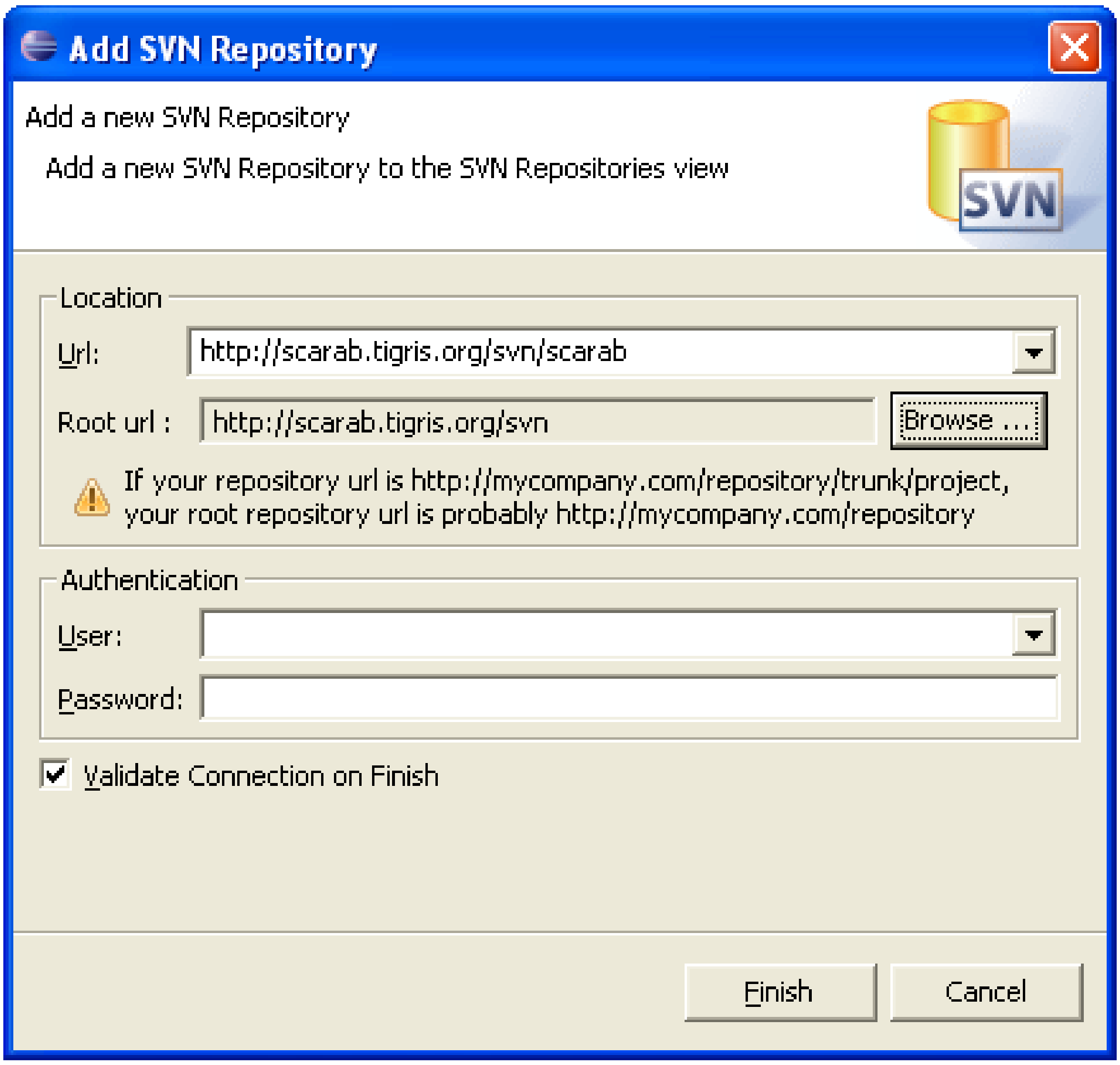
Right-click in the SVN Repository view to connect to a new Subversion repository :



The connection URL is :: *http://scarab.tigris.org/svn/scarab*

and the Root URL is : *http://scarab.tigris.org/svn*

If you are a Scarab committer, use your tigris.org login and password to connect; otherwise you do not need any particular identification and you may leave the *User* and *Password* fields blank (though it seems that any *User/ Password* will work here).


Add SVN Repository

Add a new SVN Repository

Add a new SVN Repository to the SVN Repositories view

Location

Url: http://scarab.tigris.org/svn/scarab

Root url : http://scarab.tigris.org/svn

If your repository url is http://mycompany.com/repository/trunk/project, your root repository url is probably http://mycompany.com/repository

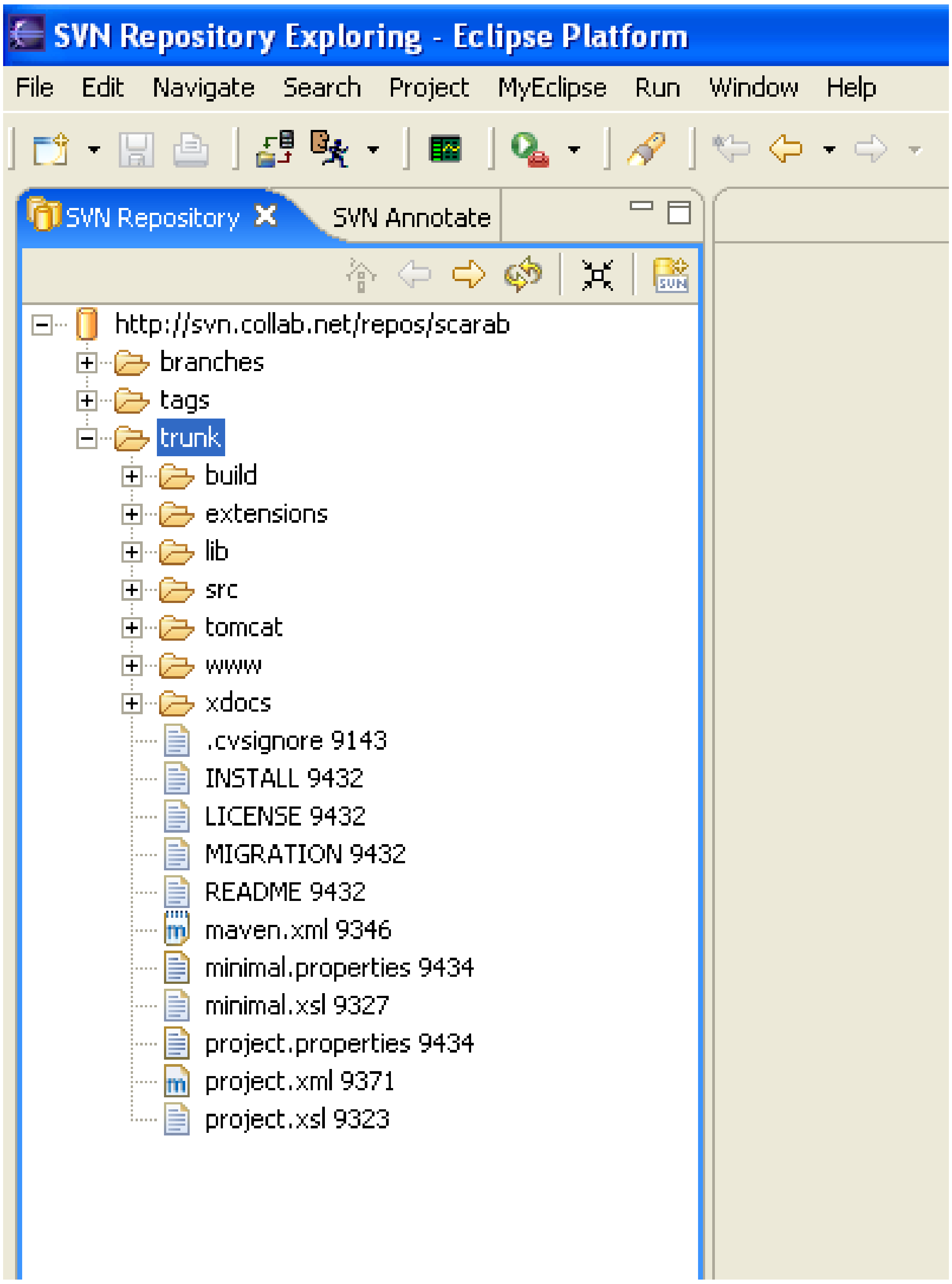
Authentication

User:

Password:

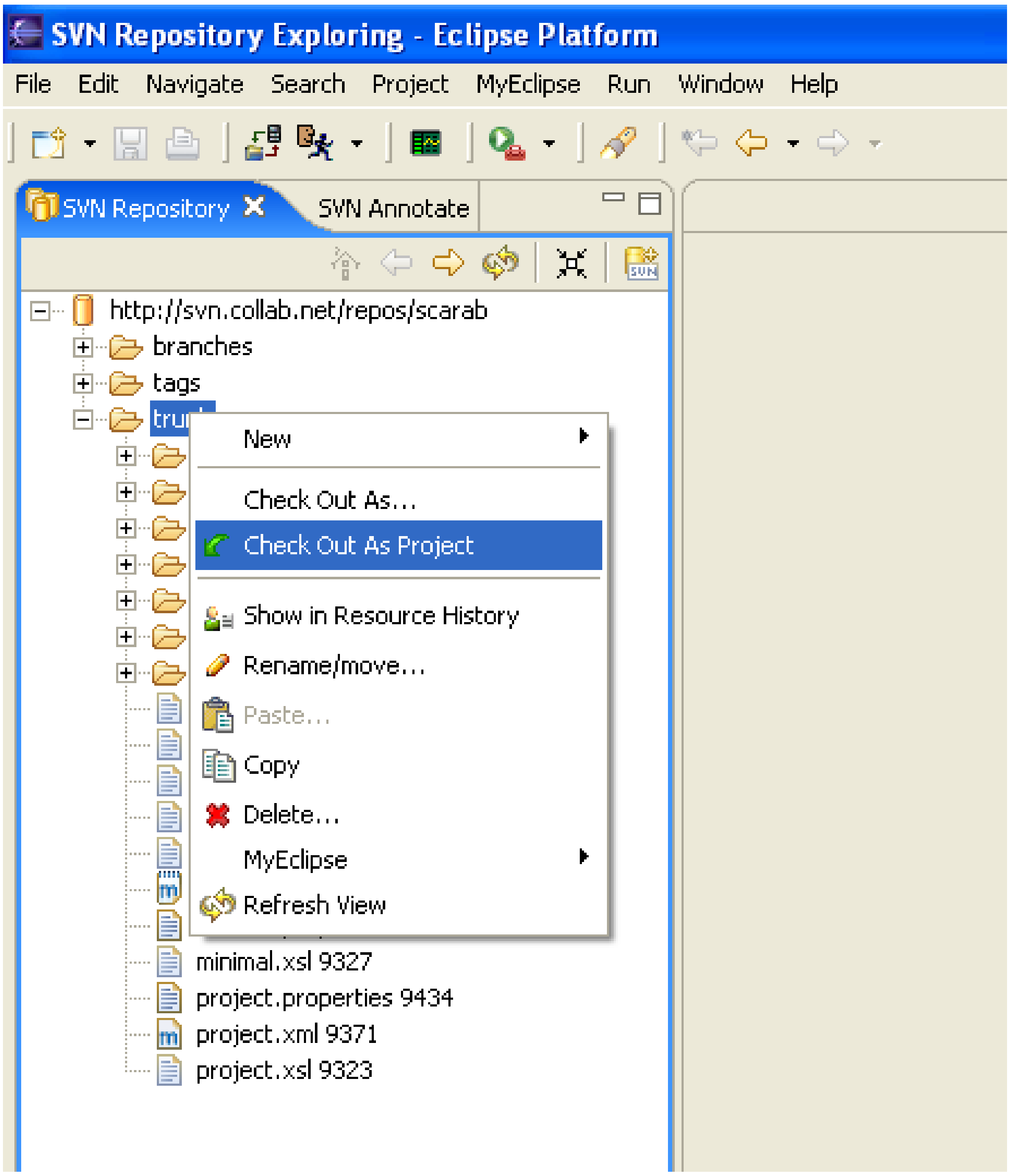
[✓] Validate Connection on Finish

To get the current version, choose the 'trunk' branch (equivalent to CVS HEAD for those who have used this version control system before).

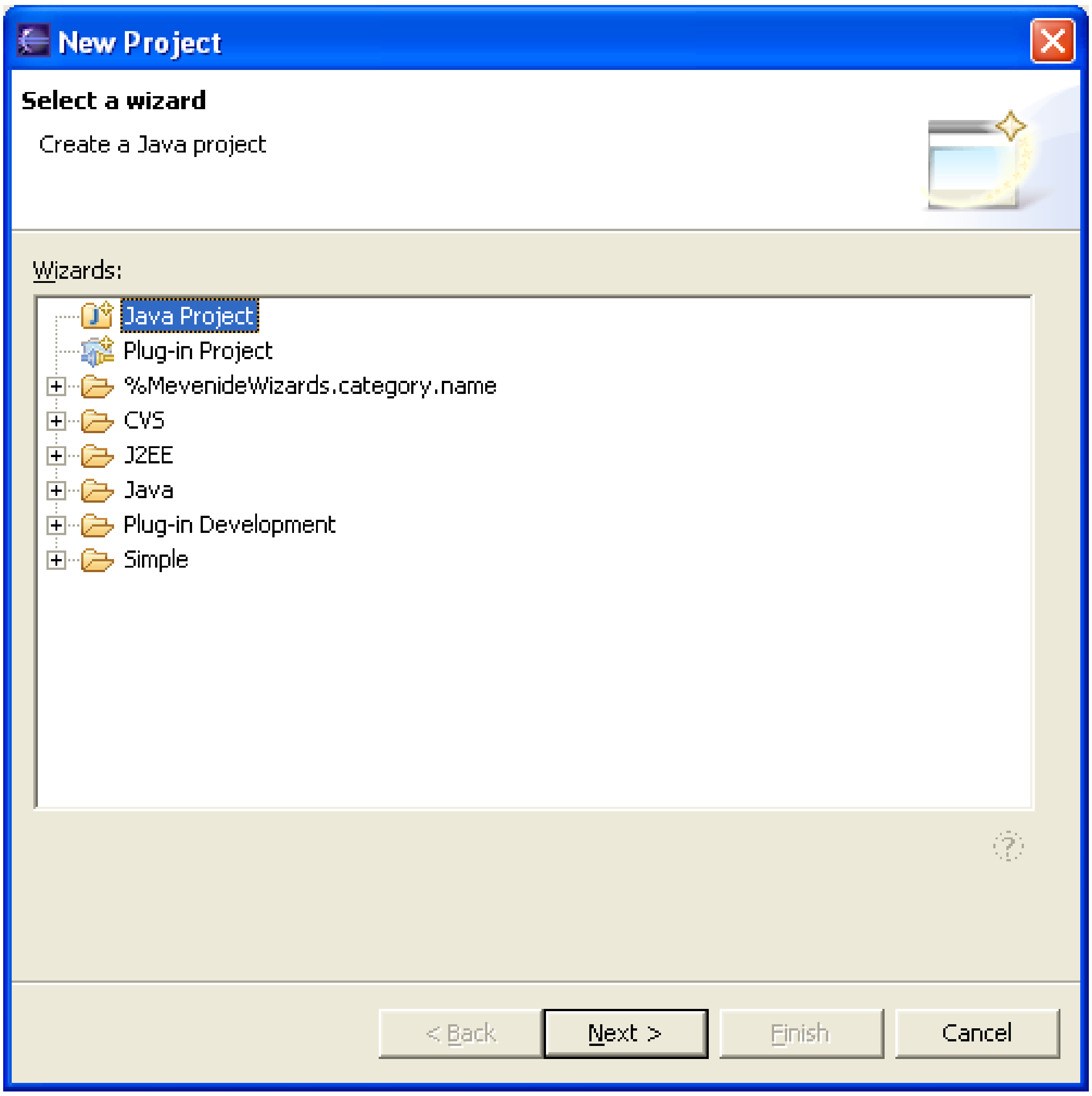

SVN Repository Exploring - Eclipse Platform
File Edit Navigate Search Project MyEclipse Run Window Help
SVN Repository
SVN Annotate

```
http://svn.collab.net/repos/scarab
  branches
  tags
  trunk
    build
    extensions
    lib
    src
    tomcat
    www
    xdocs
    .cvsignore 9143
    INSTALL 9432
    LICENSE 9432
    MIGRATION 9432
    README 9432
    maven.xml 9346
    minimal.properties 9434
    minimal.xsl 9327
    project.properties 9434
    project.xml 9371
    project.xsl 9323
```

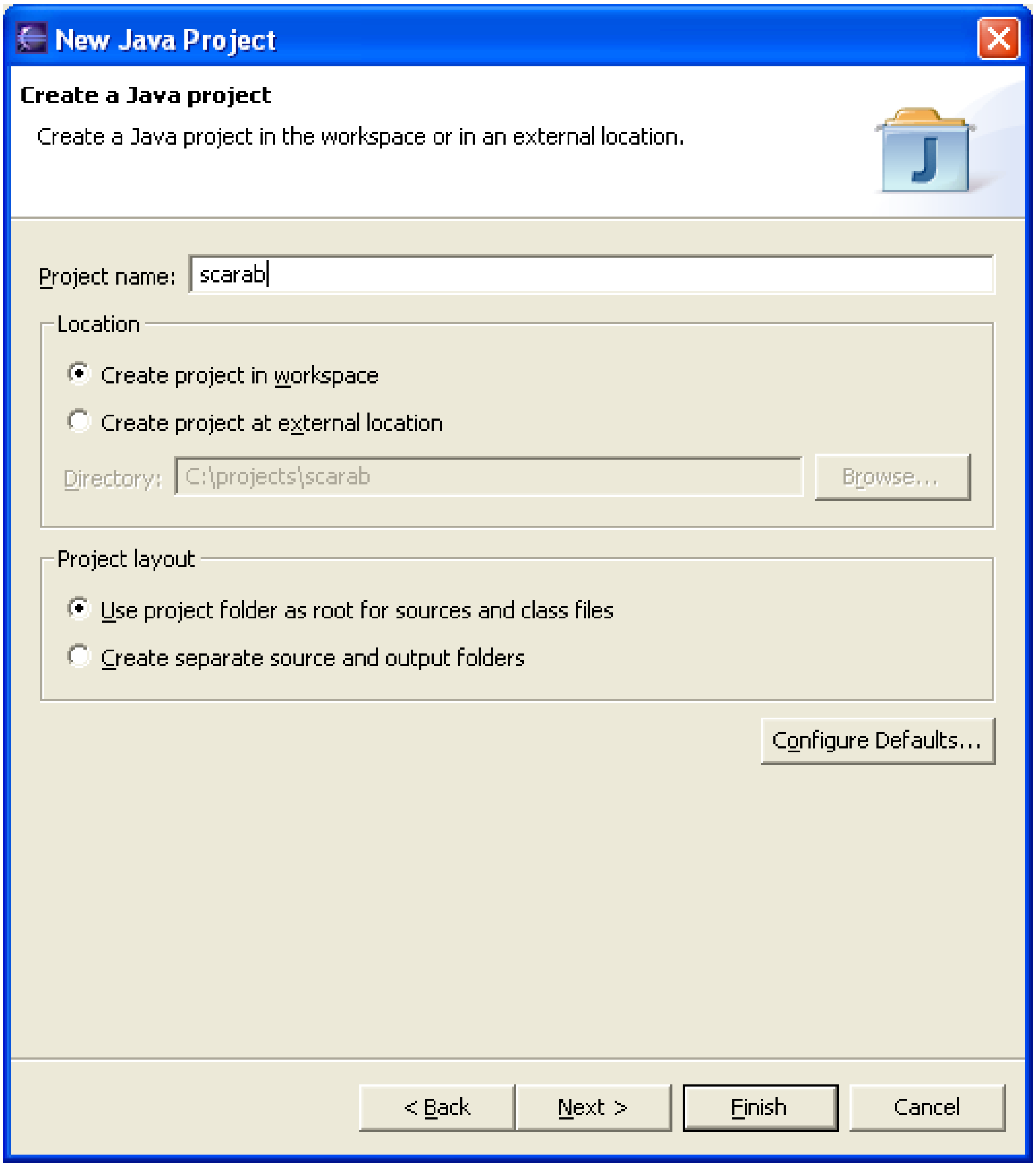
You can then use 'Checkout As Project', which allows you to rename the project (otherwise Eclipse would name it *trunk*, which is not that meaningful).



The 'New Project' wizard appears. Choose 'Java project'.



You now have to fill in the project name (you may choose *scarab* or any other name that suits you best).


New Java Project

Create a Java project

Create a Java project in the workspace or in an external location.

Project name: scarab

Location

(●) Create project in workspace

( ) Create project at external location

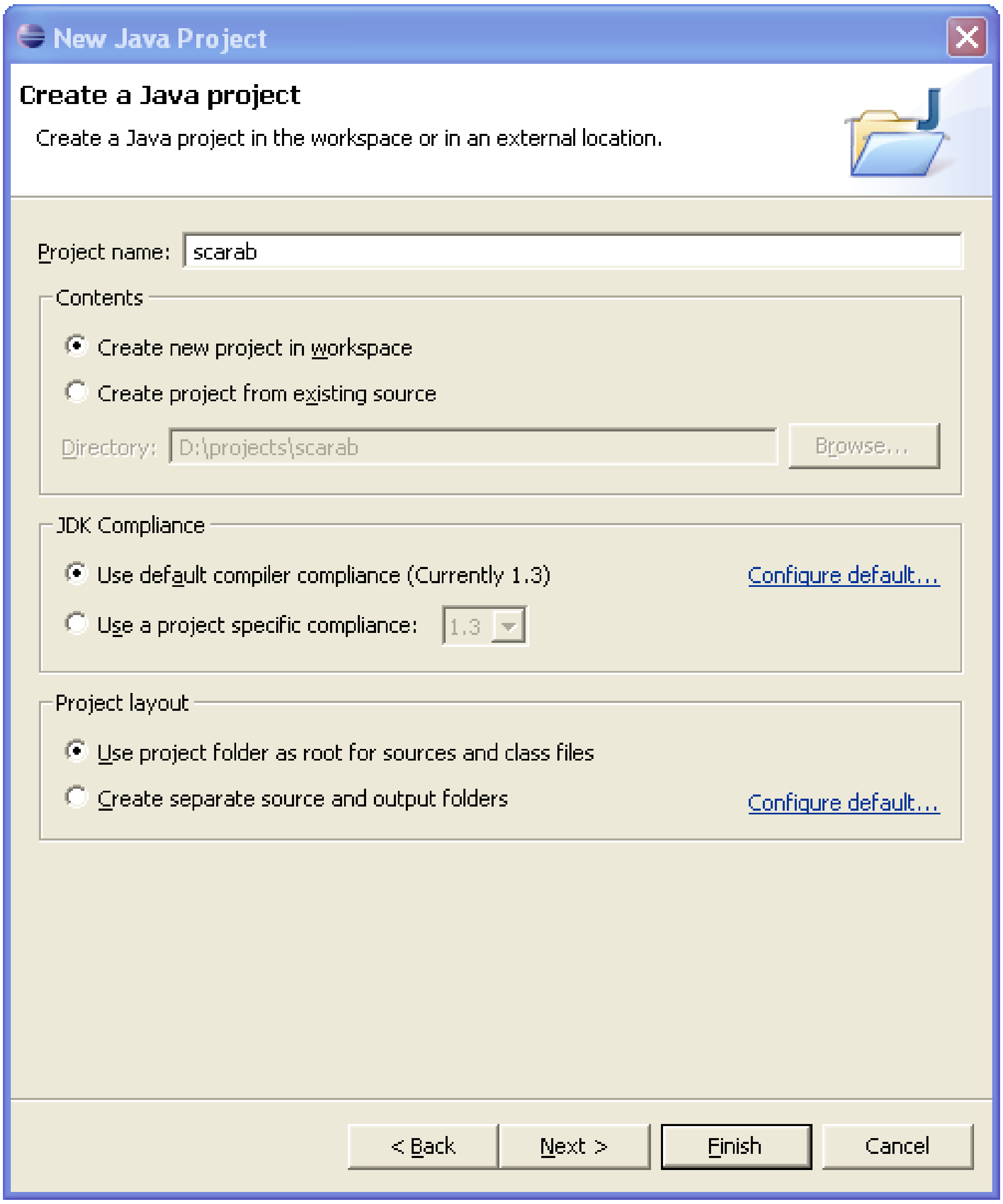
Directory: C:\projects\scarab

Project layout

(●) Use project folder as root for sources and class files

( ) Create separate source and output folders

If you are using Eclipse 3.1, you must choose J2SDK compatibility. At the moment, the development team has chosen to preserve JDK 1.3 compatibility.



Create a Java project

Create a Java project in the workspace or in an external location.

Project name: scarab

Contents

(●) Create new project in workspace

( ) Create project from existing source

Directory: D:\projects\scarab

JDK Compliance

(●) Use default compiler compliance (Currently 1.3)

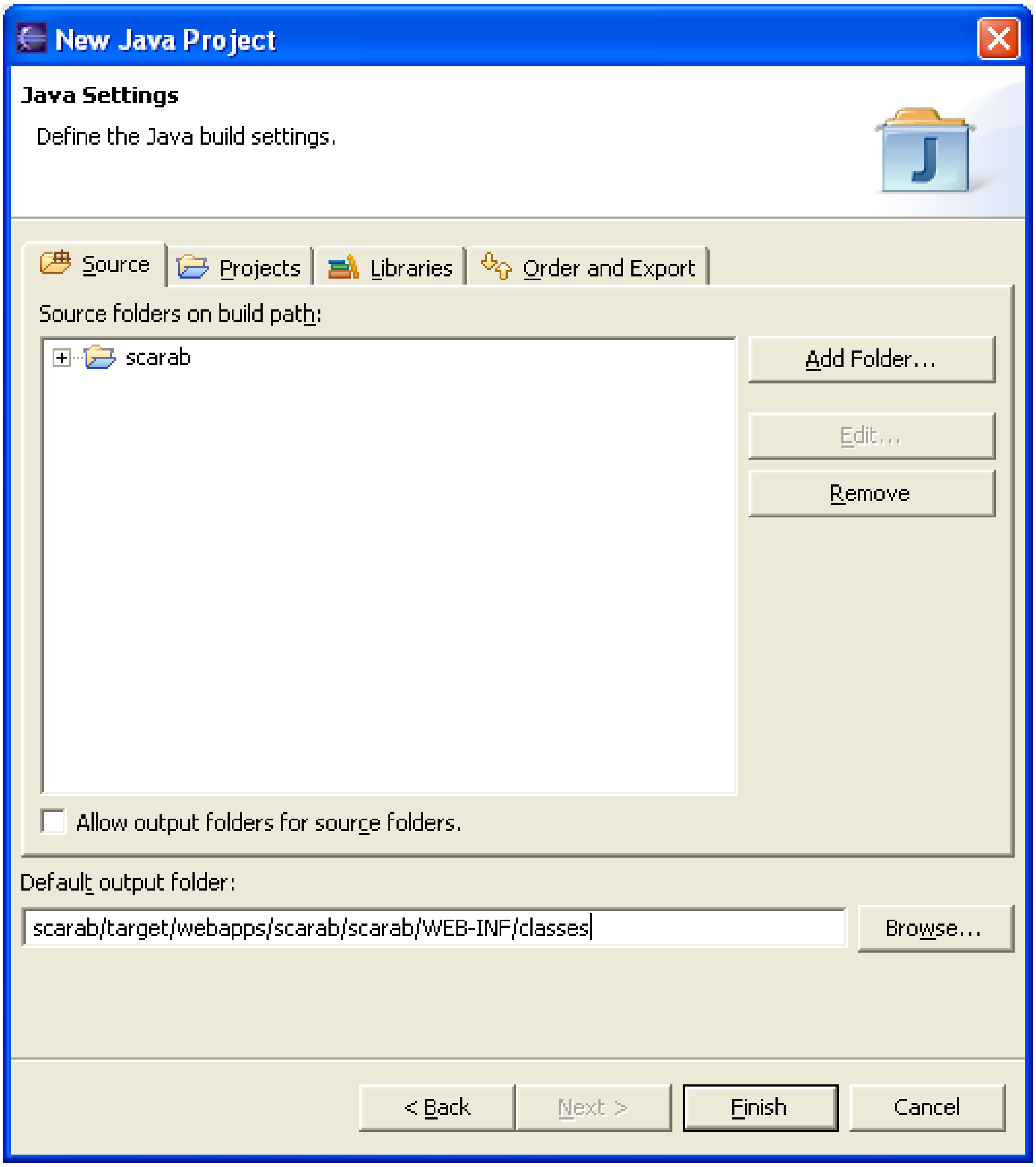
( ) Use a project specific compliance: 1.3

Project layout

(●) Use project folder as root for sources and class files

( ) Create separate source and output folders

Last step : it is best to change the *Default output folder* to `:scarab/target/webapps/scarab/scarab/WEB-INF/classes`



Source folders on build path:

scarab

Allow output folders for source folders. [ ]

Default output folder: scarab/target/webapps/scarab/scarab/WEB-INF/classes

Scarab in-place development

Installing the Sysdeo Tomcat Plugin

The Sysdeo Tomcat plugin allows you to start and stop Tomcat from Eclipse and makes debugging the application under development easier.

Download the 3.1 beta version from this URL: http://www.sysdeo.com/eclipse/tomcatplugin.

To install this plugin, all you need is to uncompress the ZIP archive in the `plugins` directory of your Eclipse installation.

Generating the required files with Maven

At the root of your project, type the following commands :

```
maven war:inplace
```

```
maven eclipse
```

This command generates the .project file used by Eclipse.

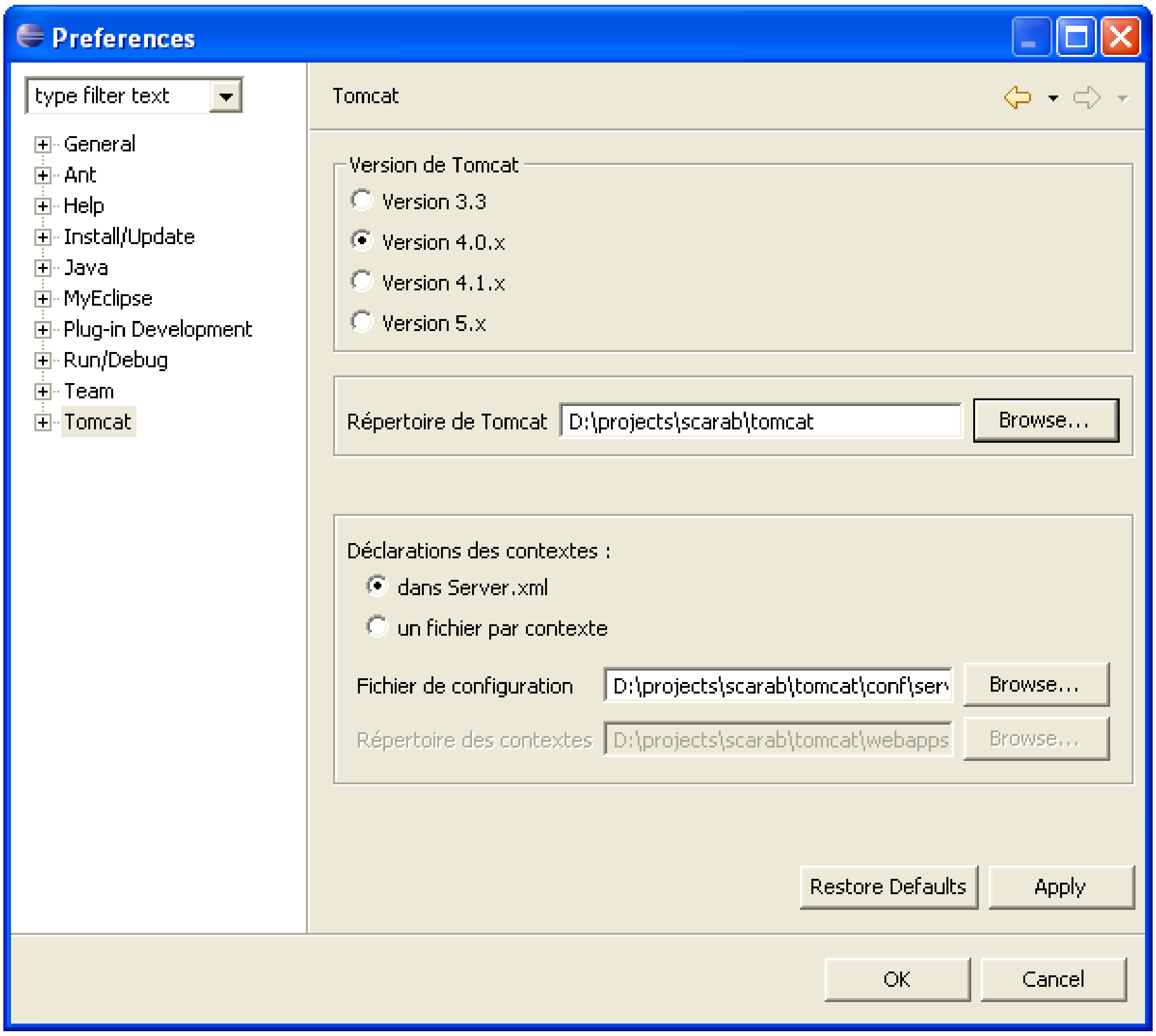
Configuring and using the Tomcat plugin

Rebuilding the project

You may now launch the build of the Scarab project in Eclipse.

Configuring the Tomcat preferences

In Window | Preferences... | Tomcat :



Preferences

type filter text

General
Ant
Help
Install/Update
Java
MyEclipse
Plug-in Development
Run/Debug
Team
Tomcat

Tomcat

Version de Tomcat
( ) Version 3.3
(•) Version 4.0.x
( ) Version 4.1.x
( ) Version 5.x

Répertoire de Tomcat: D:\projects\scarab\tomcat [Browse...]

Déclarations des contextes :
(•) dans Server.xml
( ) un fichier par contexte

Fichier de configuration: D:\projects\scarab\tomcat\conf\serv [Browse...]
Répertoire des contextes: D:\projects\scarab\tomcat\webapps [Browse...]

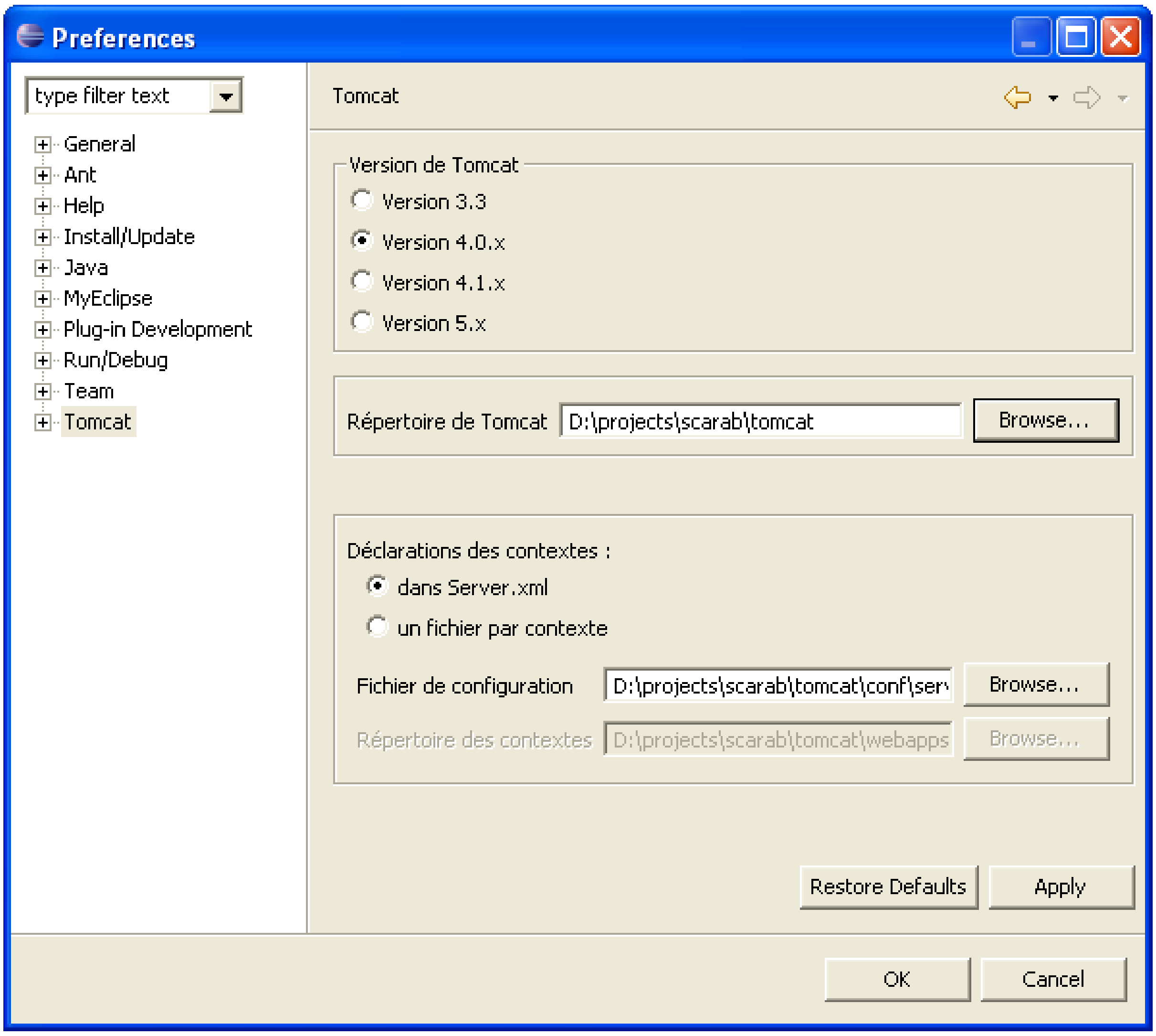
[Restore Defaults] [Apply]

[OK] [Cancel]

- Select the 4.0.x version
- Tomcat directory : the `tomcat` directory in your Scarab project
- Context declaration in server.xml

Setting the project Tomcat properties

In Project | Properties | Tomcat :


Preferences

type filter text

- General
- Ant
- Help
- Install/Update
- Java
- MyEclipse
- Plug-in Development
- Run/Debug
- Team
- Tomcat

Tomcat

Version de Tomcat
- Version 3.3
- [✓ Version 4.0.x]
- Version 4.1.x
- Version 5.x

Répertoire de Tomcat: D:\projects\scarab\tomcat Browse...

Déclarations des contextes :
- [✓ dans Server.xml]
- un fichier par contexte

Fichier de configuration: D:\projects\scarab\tomcat\conf\serv Browse...

Répertoire des contextes: D:\projects\scarab\tomcat\webapps Browse...

Restore Defaults Apply

OK Cancel

- Check the 'Is Tomcat project'
- Enter 'scarab' as context name
- Uncheck 'can update server.xml'
- Check 'Mark context as reloadable'
- Check 'Redirect context logger to Eclipse console...'

Chapter 5. Scarab development with Netbeans

Getting the Scarab sources

The current implementation of a Subversion client for Netbeans (code named "teepee", available from http://subversion.netbeans.org/) is only a prototype. So, at the moment, you need to use another client to check the sources out of the Scarab Subversion repository. If you use Windows as your development environment, you'll probably want to use TortoiseSVN [http://tortoisesvn.tigris.org/] - another Tigris project.

Getting the Mevenide plug-in (Maven-Netbeans integration)

Your best option to develop Scarab in Netbeans is certainly to use the Mevenide plug-in (which is, by the way, much more stable and functional in Netbeans than in Eclipse, not to mention JBuilder).

Download Mevenide for Netbeans at this URL: http://mevenide.codehaus.org/mevenide-netbeans-project/index.html

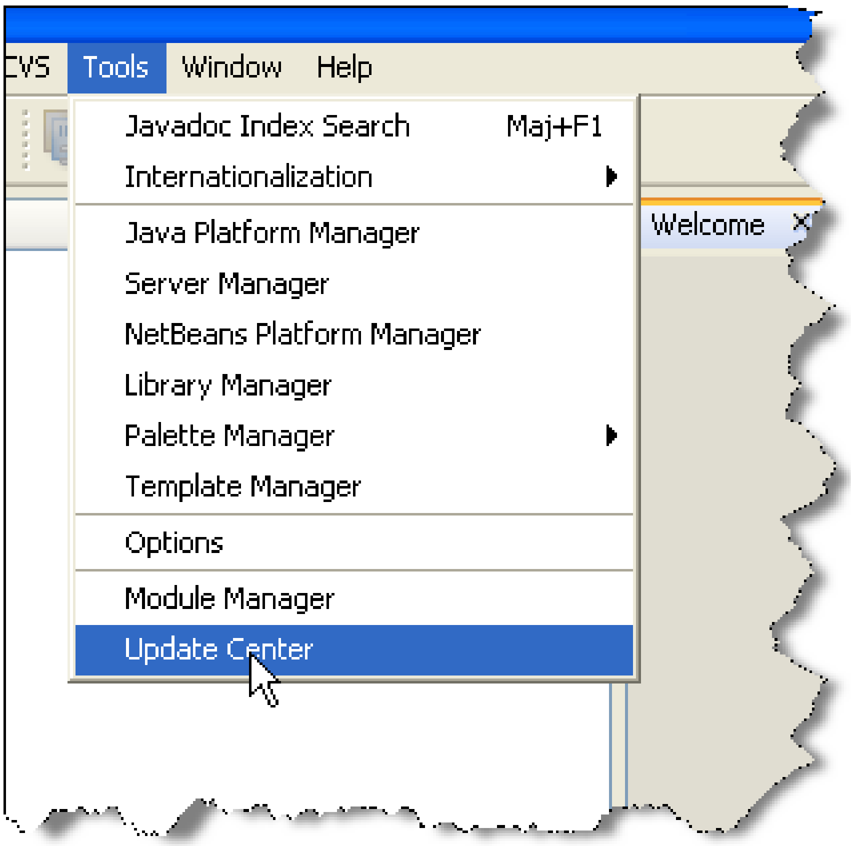
You will get a ZIP file with 16 nbm (Netbeans modules) inside. Unzip the file to a temporary directory.

Installing the Mevenide plug-in

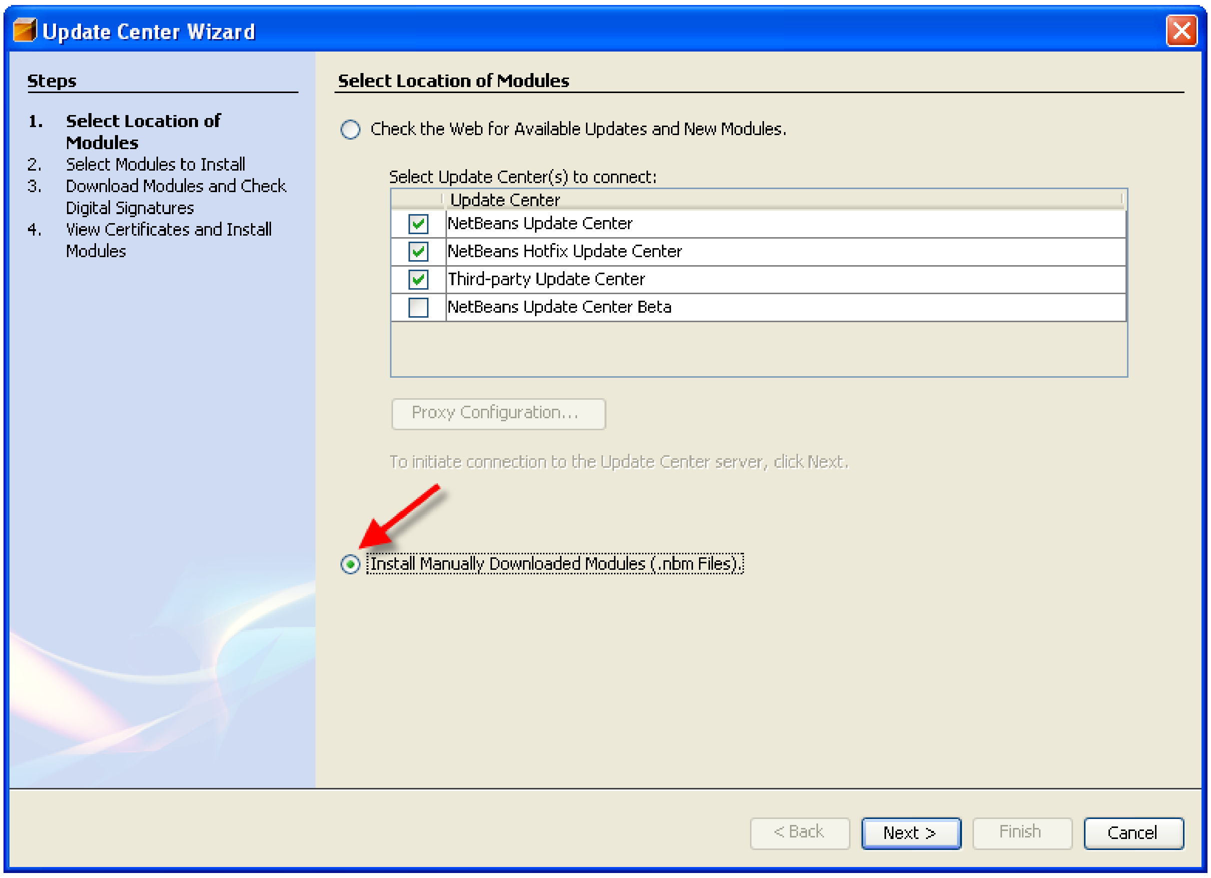
Tip

If you are familiar with Netbeans modules installation, you will probably want to skip this section.

In the Tools menu, choose Update Center



Check 'Install Manually Downloaded Modules (.nbm Files)'


Update Center Wizard

Steps

1. **Select Location of Modules**
2. Select Modules to Install
3. Download Modules and Check Digital Signatures
4. View Certificates and Install Modules

Select Location of Modules

( ) Check the Web for Available Updates and New Modules.

Select Update Center(s) to connect:

	Update Center
✓	NetBeans Update Center
✓	NetBeans Hotfix Update Center
✓	Third-party Update Center
	NetBeans Update Center Beta

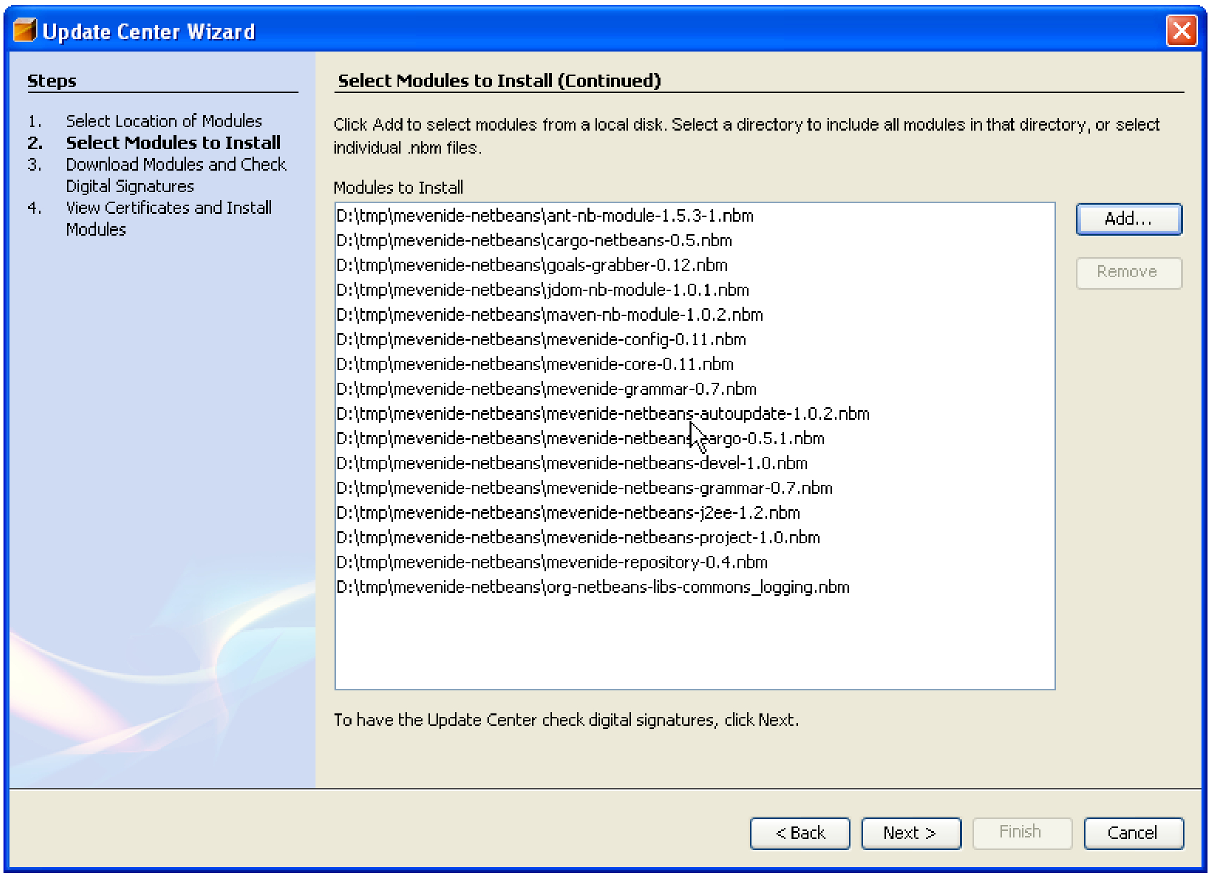
Proxy Configuration...

To initiate connection to the Update Center server, click Next.

[✓ Install Manually Downloaded Modules (.nbm Files).]

< Back | Next > | Finish | Cancel

Use the 'Add...' button to add the 16 files you have downloaded.


Update Center Wizard

Steps
1. Select Location of Modules
2. **Select Modules to Install**
3. Download Modules and Check Digital Signatures
4. View Certificates and Install Modules

Select Modules to Install (Continued)

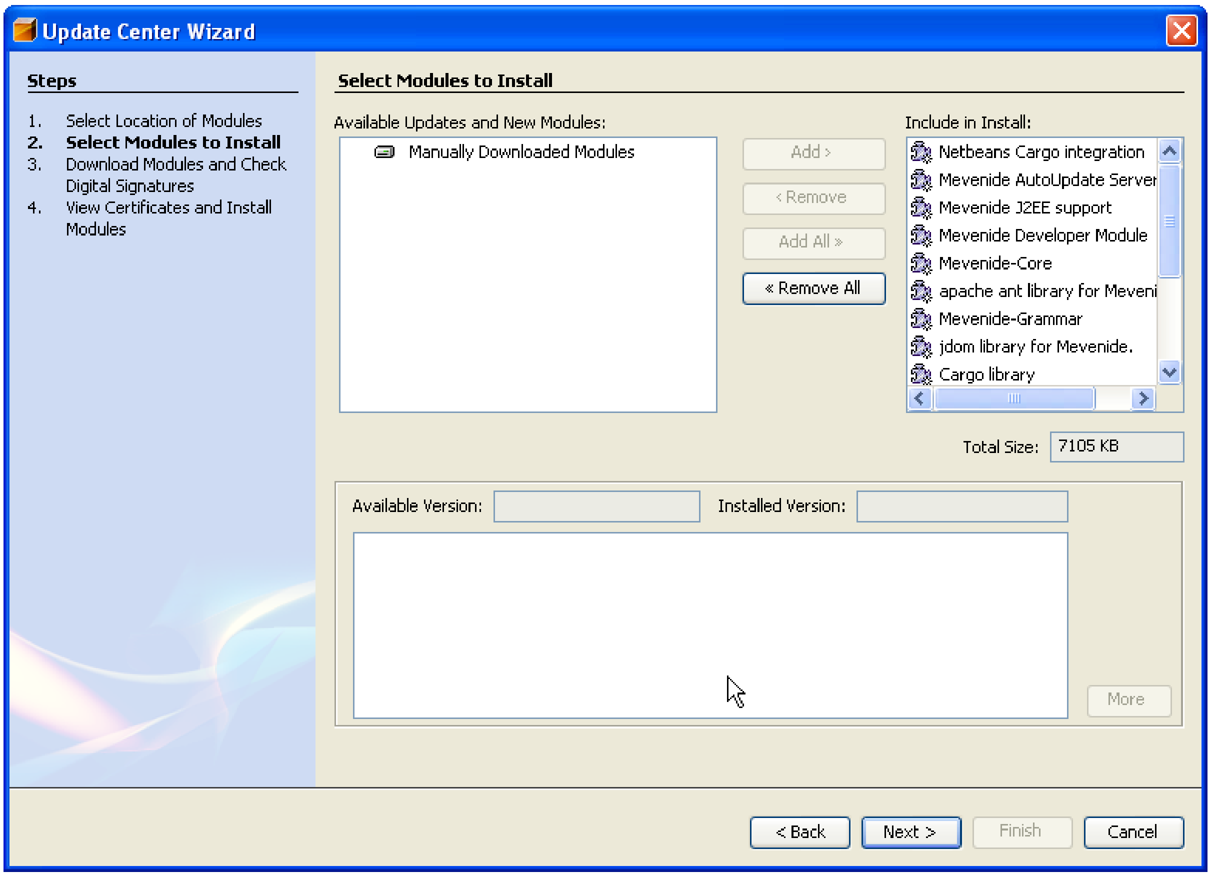
Click Add to select modules from a local disk. Select a directory to include all modules in that directory, or select individual .nbm files.

Modules to Install

```
D:\tmp\mevenide-netbeans\ant-nb-module-1.5.3-1.nbm
D:\tmp\mevenide-netbeans\cargo-netbeans-0.5.nbm
D:\tmp\mevenide-netbeans\goals-grabber-0.12.nbm
D:\tmp\mevenide-netbeans\jdom-nb-module-1.0.1.nbm
D:\tmp\mevenide-netbeans\maven-nb-module-1.0.2.nbm
D:\tmp\mevenide-netbeans\mevenide-config-0.11.nbm
D:\tmp\mevenide-netbeans\mevenide-core-0.11.nbm
D:\tmp\mevenide-netbeans\mevenide-grammar-0.7.nbm
D:\tmp\mevenide-netbeans\mevenide-netbeans-autoupdate-1.0.2.nbm
D:\tmp\mevenide-netbeans\mevenide-netbeans-cargo-0.5.1.nbm
D:\tmp\mevenide-netbeans\mevenide-netbeans-devel-1.0.nbm
D:\tmp\mevenide-netbeans\mevenide-netbeans-grammar-0.7.nbm
D:\tmp\mevenide-netbeans\mevenide-netbeans-j2ee-1.2.nbm
D:\tmp\mevenide-netbeans\mevenide-netbeans-project-1.0.nbm
D:\tmp\mevenide-netbeans\mevenide-repository-0.4.nbm
D:\tmp\mevenide-netbeans\org-netbeans-libs-commons_logging.nbm
```

To have the Update Center check digital signatures, click Next.

Just click 'Next >'.


Update Center Wizard

Steps
1. Select Location of Modules
2. **Select Modules to Install**
3. Download Modules and Check Digital Signatures
4. View Certificates and Install Modules

Select Modules to Install

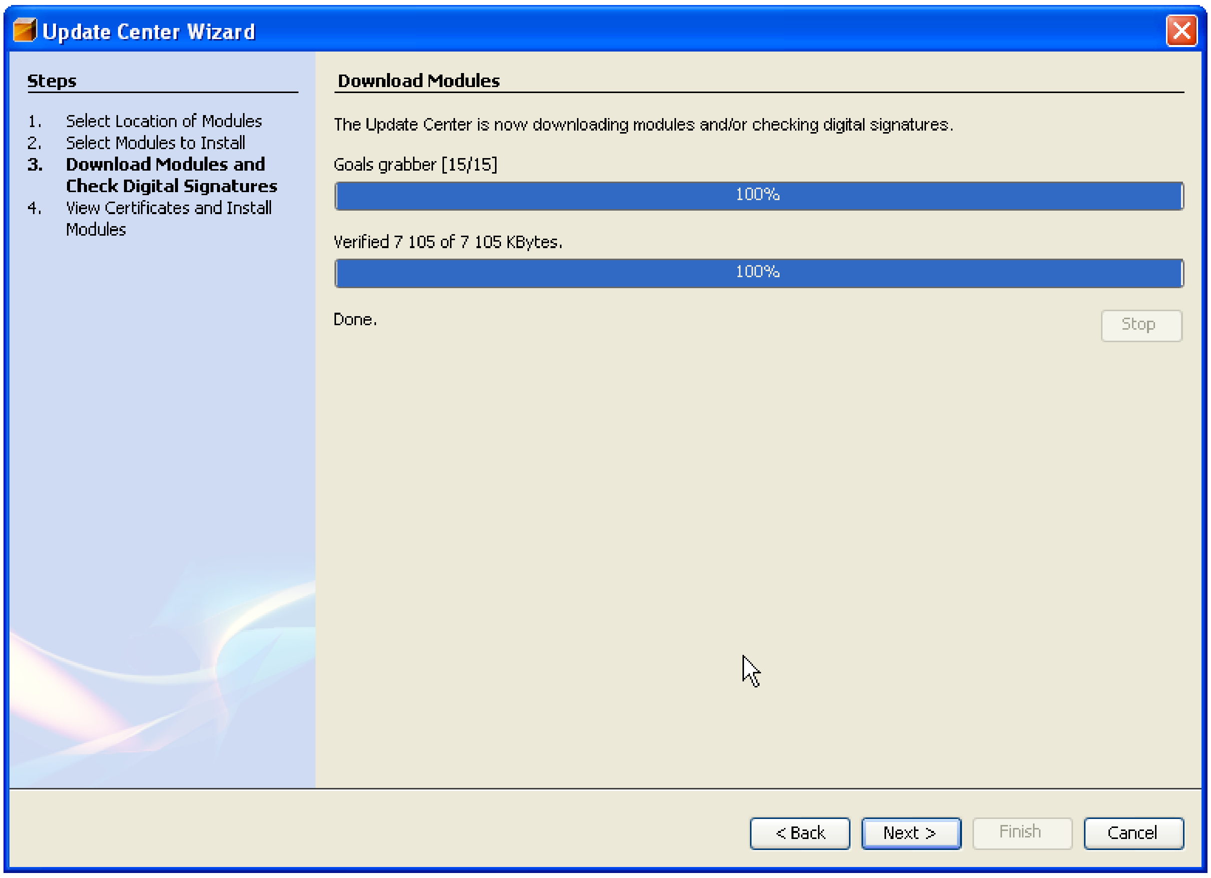
Available Updates and New Modules:
- Manually Downloaded Modules

Include in Install:
- Netbeans Cargo integration
- Mevenide AutoUpdate Server
- Mevenide J2EE support
- Mevenide Developer Module
- Mevenide-Core
- apache ant library for Meveni
- Mevenide-Grammar
- jdom library for Mevenide.
- Cargo library

Total Size: 7105 KB

Available Version:
Installed Version:

You will have to accept the different licences. The modules are then ready for installation.


Update Center Wizard
Steps
1. Select Location of Modules
2. Select Modules to Install
3. Download Modules and Check Digital Signatures
4. View Certificates and Install Modules
Download Modules
100%
100%
Stop
< Back
Next >
Finish
Cancel

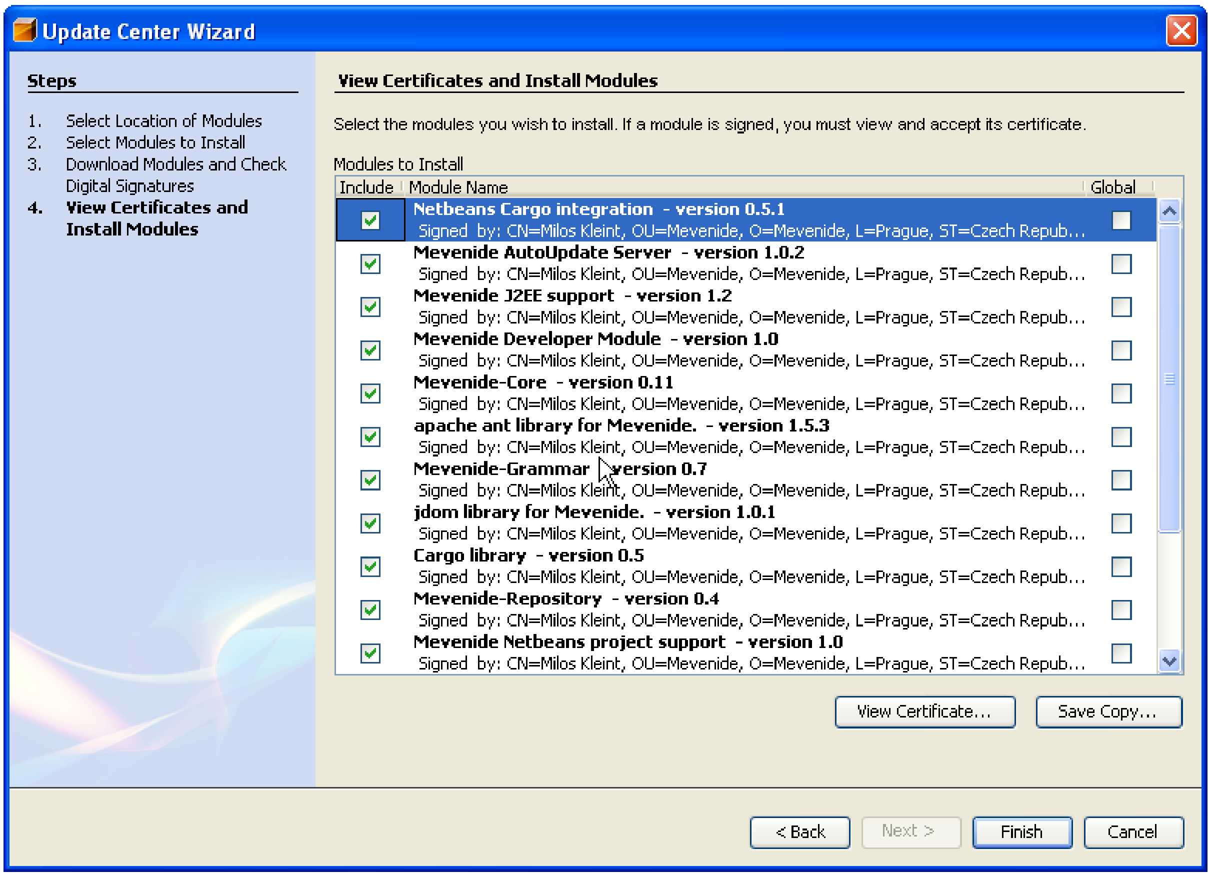
The Update Center is now downloading modules and/or checking digital signatures.

Goals grabber [15/15]

Verified 7 105 of 7 105 KBytes.

Done.

View the certificate (there is only one here) and click 'Finish' to install the modules.


Update Center Wizard
Steps
1. Select Location of Modules
2. Select Modules to Install
3. Download Modules and Check Digital Signatures
4. View Certificates and Install Modules
View Certificates and Install Modules
View Certificate...
Save Copy...
< Back
Next >
Finish
Cancel

Select the modules you wish to install. If a module is signed, you must view and accept its certificate.

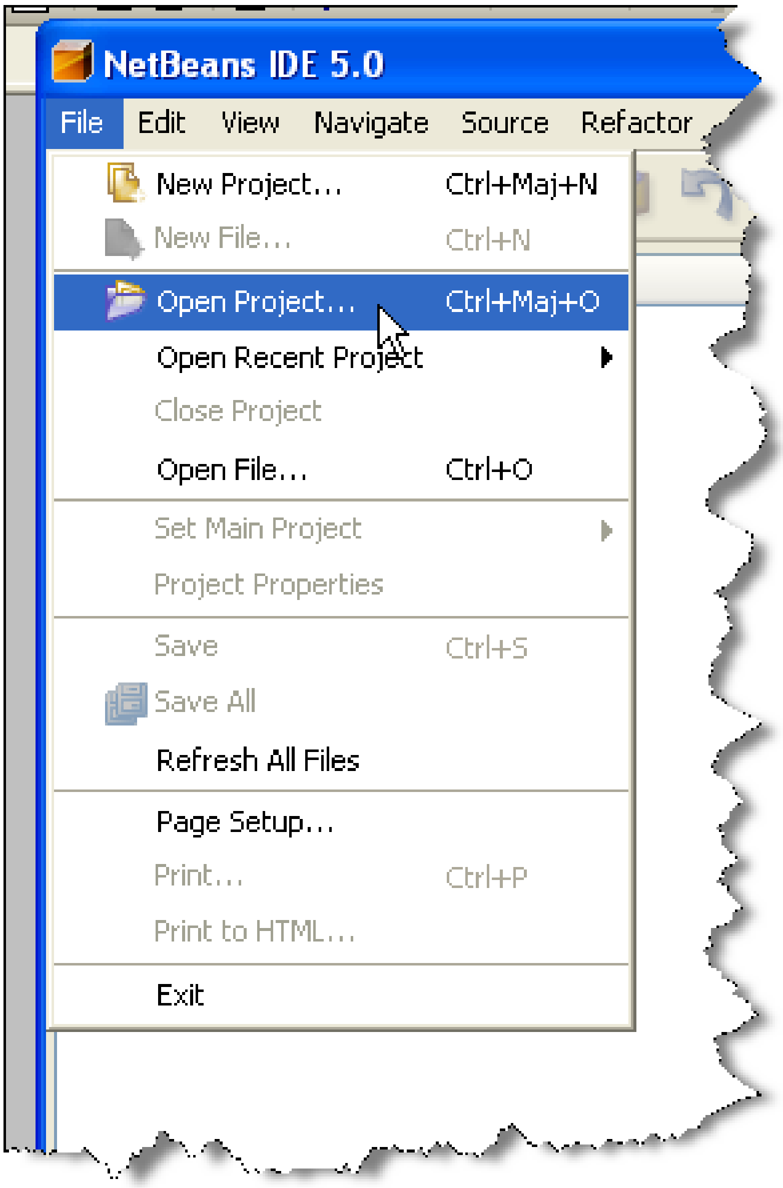
Modules to Install

Include	Module Name	Global
✓	**Netbeans Cargo integration - version 0.5.1** Signed by: CN=Milos Kleint, OU=Mevenide, O=Mevenide, L=Prague, ST=Czech Repub...	☐
✓	**Mevenide AutoUpdate Server - version 1.0.2** Signed by: CN=Milos Kleint, OU=Mevenide, O=Mevenide, L=Prague, ST=Czech Repub...	☐
✓	**Mevenide J2EE support - version 1.2** Signed by: CN=Milos Kleint, OU=Mevenide, O=Mevenide, L=Prague, ST=Czech Repub...	☐
✓	**Mevenide Developer Module - version 1.0** Signed by: CN=Milos Kleint, OU=Mevenide, O=Mevenide, L=Prague, ST=Czech Repub...	☐
✓	**Mevenide-Core - version 0.11** Signed by: CN=Milos Kleint, OU=Mevenide, O=Mevenide, L=Prague, ST=Czech Repub...	☐
✓	**apache ant library for Mevenide. - version 1.5.3** Signed by: CN=Milos Kleint, OU=Mevenide, O=Mevenide, L=Prague, ST=Czech Repub...	☐
✓	**Mevenide-Grammar - version 0.7** Signed by: CN=Milos Kleint, OU=Mevenide, O=Mevenide, L=Prague, ST=Czech Repub...	☐
✓	**jdom library for Mevenide. - version 1.0.1** Signed by: CN=Milos Kleint, OU=Mevenide, O=Mevenide, L=Prague, ST=Czech Repub...	☐
✓	**Cargo library - version 0.5** Signed by: CN=Milos Kleint, OU=Mevenide, O=Mevenide, L=Prague, ST=Czech Repub...	☐
✓	**Mevenide-Repository - version 0.4** Signed by: CN=Milos Kleint, OU=Mevenide, O=Mevenide, L=Prague, ST=Czech Repub...	☐
✓	**Mevenide Netbeans project support - version 1.0** Signed by: CN=Milos Kleint, OU=Mevenide, O=Mevenide, L=Prague, ST=Czech Repub...	☐

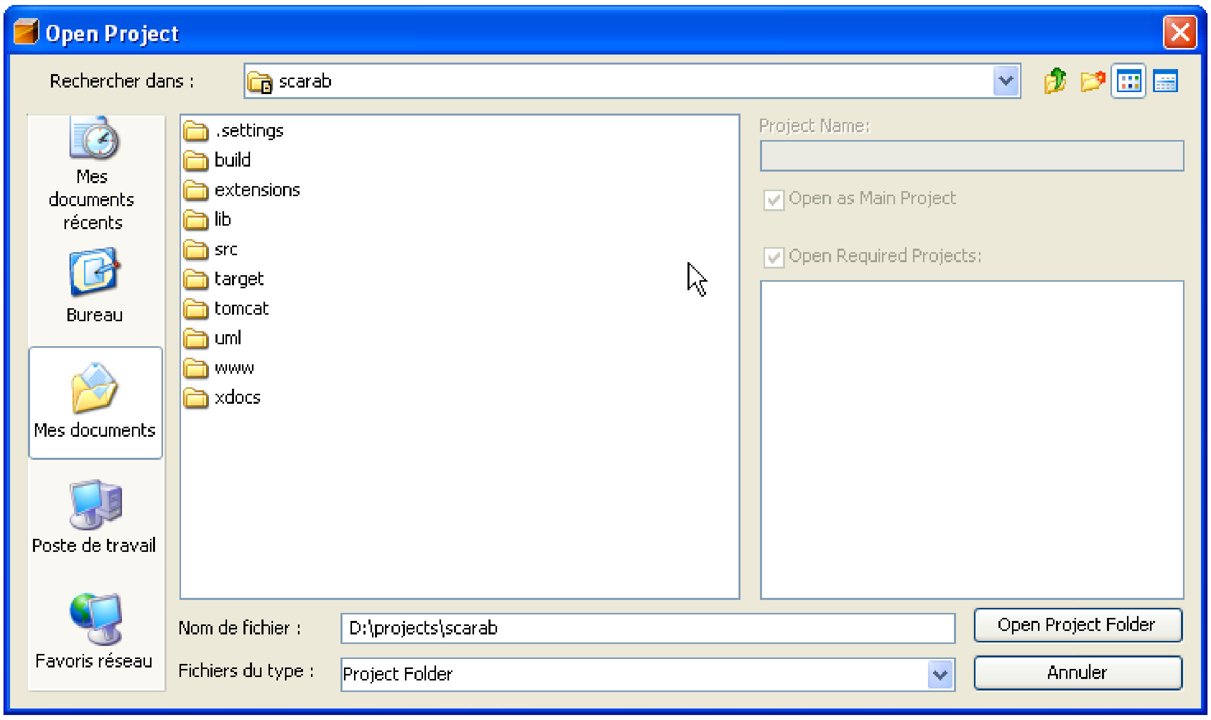
You need now to restart Netbeans to use the Mevenide plug-in you have just installed.

Scarab in-place development

Accessing the Scarab project is now as easy as it can be. In the File menu, choose 'Open Project...'



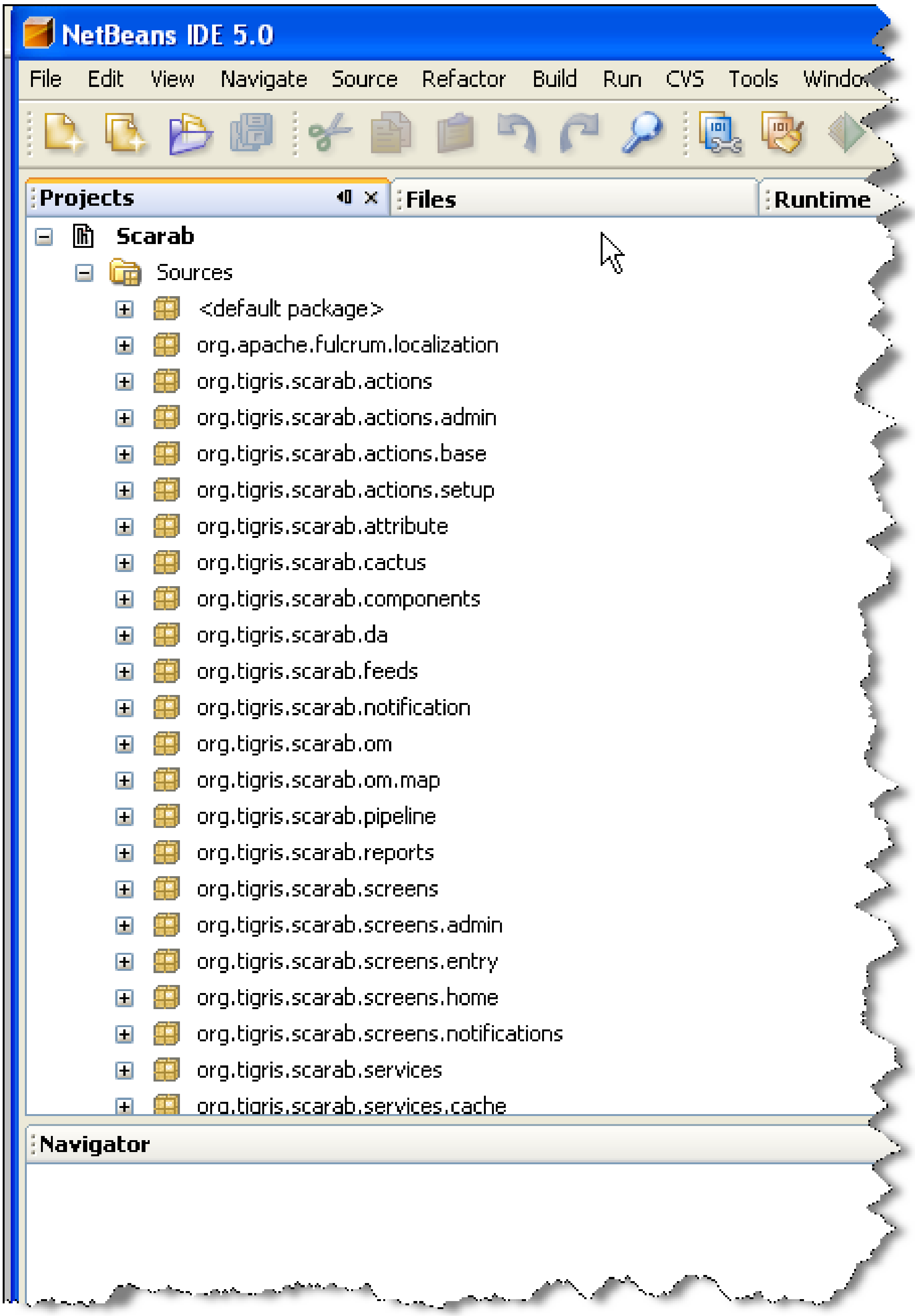
Choose the directory in which you checked out Scarab from the Subversion repository.



Nom de fichier : D:\projects\scarab

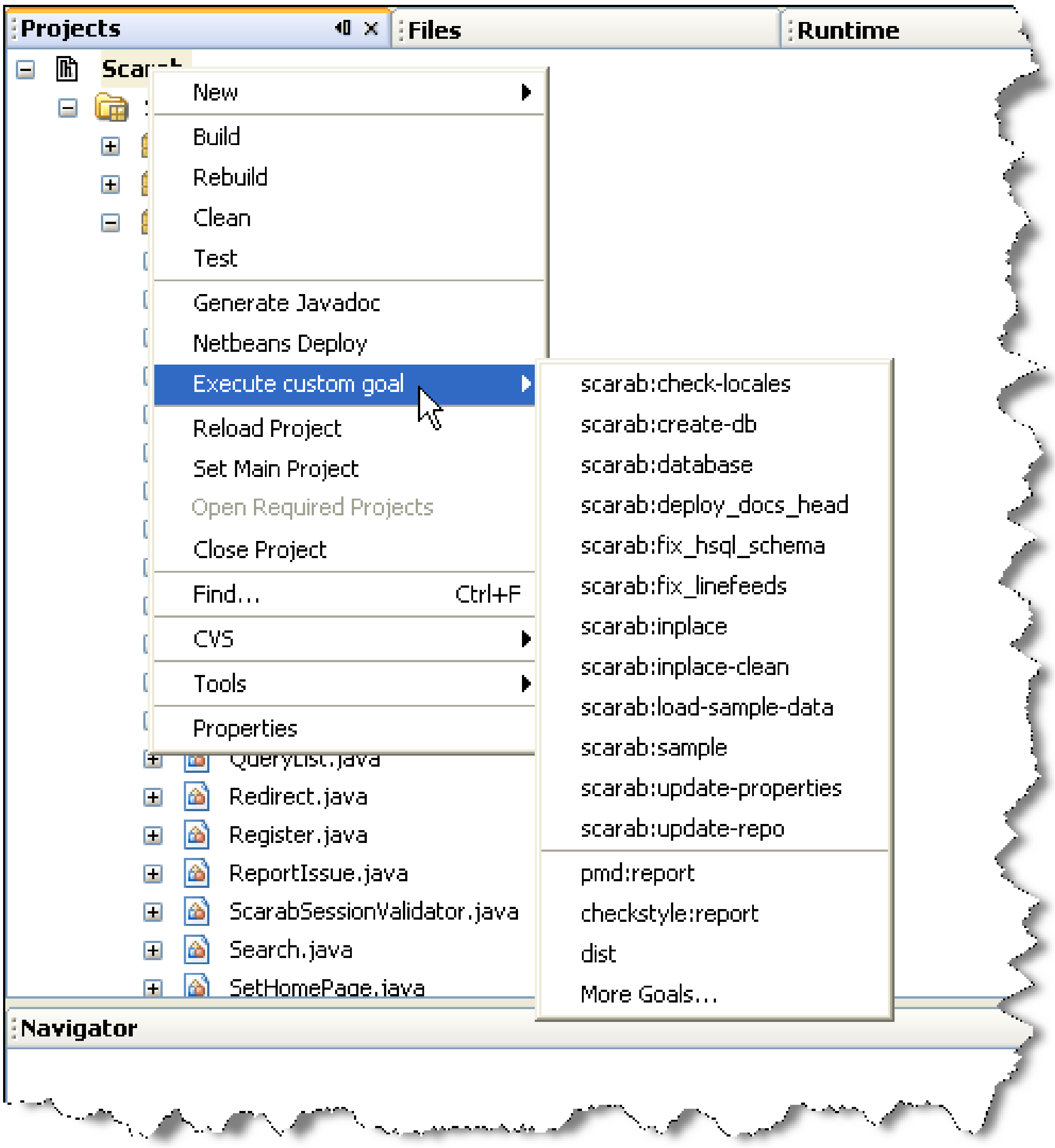
Fichiers du type : Project Folder

There it is !


NetBeans IDE 5.0
File Edit View Navigate Source Refactor Build Run CVS Tools Windo
Projects
Files
Runtime
Navigator

```
Scarab
  Sources
    <default package>
    org.apache.fulcrum.localization
    org.tigris.scarab.actions
    org.tigris.scarab.actions.admin
    org.tigris.scarab.actions.base
    org.tigris.scarab.actions.setup
    org.tigris.scarab.attribute
    org.tigris.scarab.cactus
    org.tigris.scarab.components
    org.tigris.scarab.da
    org.tigris.scarab.feeds
    org.tigris.scarab.notification
    org.tigris.scarab.om
    org.tigris.scarab.om.map
    org.tigris.scarab.pipeline
    org.tigris.scarab.reports
    org.tigris.scarab.screens
    org.tigris.scarab.screens.admin
    org.tigris.scarab.screens.entry
    org.tigris.scarab.screens.home
    org.tigris.scarab.screens.notifications
    org.tigris.scarab.services
    org.tigris.scarab.services.cache
```

You have direct access to Scarab custom Maven goals by right-clicking the Scarab project.


Projects
Files
Runtime
Scarab
New
Build
Rebuild
Clean
Test
Generate Javadoc
Netbeans Deploy
Execute custom goal
Reload Project
Set Main Project
Open Required Projects
Close Project
Find... Ctrl+F
CVS
Tools
Properties
scarab:check-locales
scarab:create-db
scarab:database
scarab:deploy_docs_head
scarab:fix_hsql_schema
scarab:fix_linefeeds
scarab:inplace
scarab:inplace-clean
scarab:load-sample-data
scarab:sample
scarab:update-properties
scarab:update-repo
pmd:report
checkstyle:report
dist
More Goals...
QueryList.java
Redirect.java
Register.java
ReportIssue.java
ScarabSessionValidator.java
Search.java
SetHomePage.java
Navigator

Chapter 6. The Scarab data model

Introduction

Scarab data are accessed via Torque, the OR mapping developed for and with the Turbine framework (but Torque is today independent in its way).

The Scarab database schema is formally described in XML. This schema will be explained in three parts, corresponding to the three XML files under `./src/schema`.

- the table associated to the primary key broker (`id-table-schema.xml`);
- the tables used to manage the Scarab users (for authentification and other purposes) and the permissions in the Turbine framework (`turbine-schema.xml`);
- the tables of the Scarab application itself (`scarab-schema.xml`).

From this formal definition of the database schema, Torque generates:

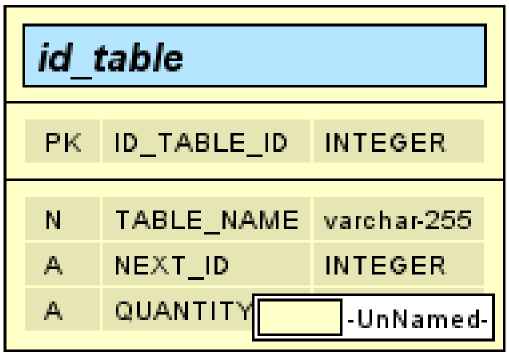
- SQL (DDL) scripts to create tables in the various RDBMS;
- a set of java classes that map the database entities as objects (the so called "peers").

Unique IDs generation: the `id_table` table

Most tables in the Scarab schema have meaningless (large) integers as primary keys. Many RDBMS offer a native mechanism to generate such primary keys (integers, not necessarily in sequence): Oracle SEQUENCE's, auto-incremented columns in MySQL or MSSQL, etc.

One of the Scarab design goals was to be portable across many RDBMS; thus, the generation of primary keys could hardly rely on such native mechanisms. So Scarab uses for this a Torque functionality ("id broker").

The **`id_table`** table is used to generate primary keys for the various tables in the Scarab schema.



Reserved ID ranges

As a rule, everything under 10000 is reserved for development use (default data, sample data etc.)

Table 6.1. ID ranges

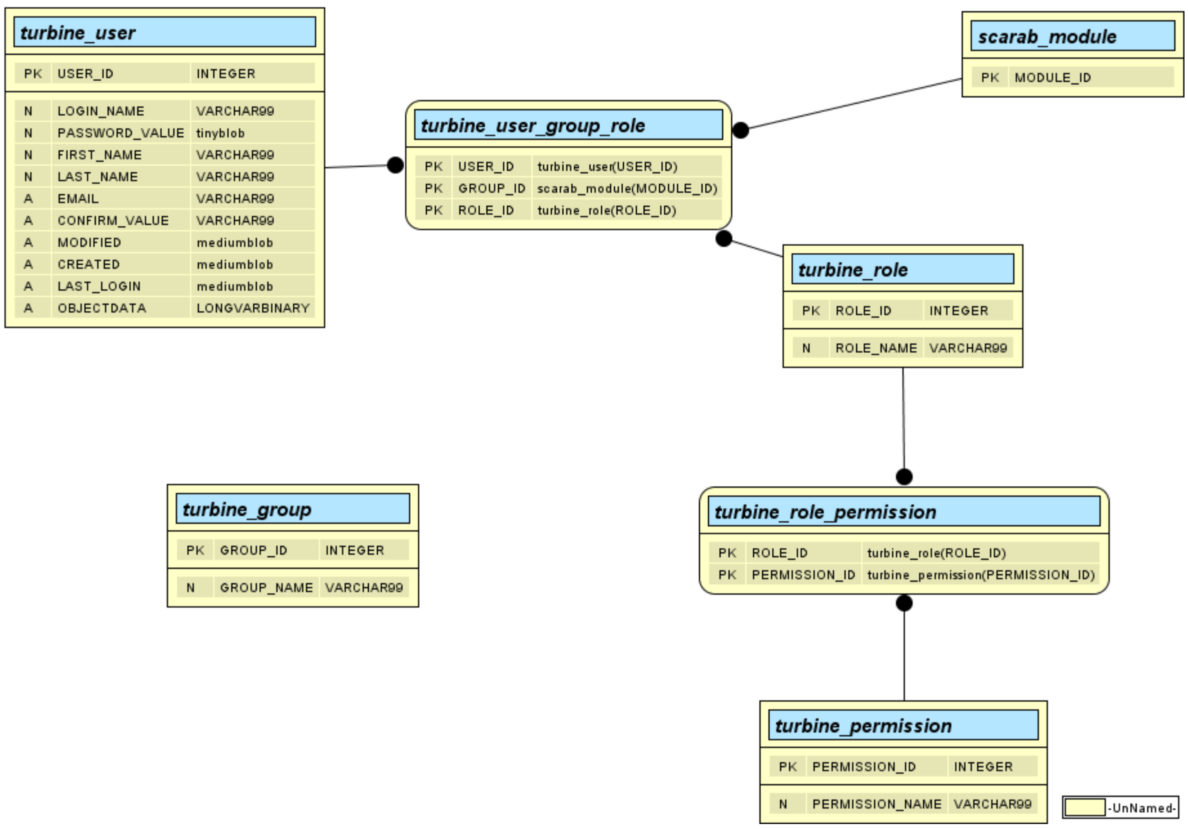
ID range	Use
0-99	Required data.
100-189	Default data.
190-199	Anonymous user data.
200-299	JIRA templates.
300-399	Bugzilla templates.
1000-1999	Sample data.
2000-8999	(Reserved for future use.)
9000-9999	User custom (custom user templates, etc.)

Turbine framework tables

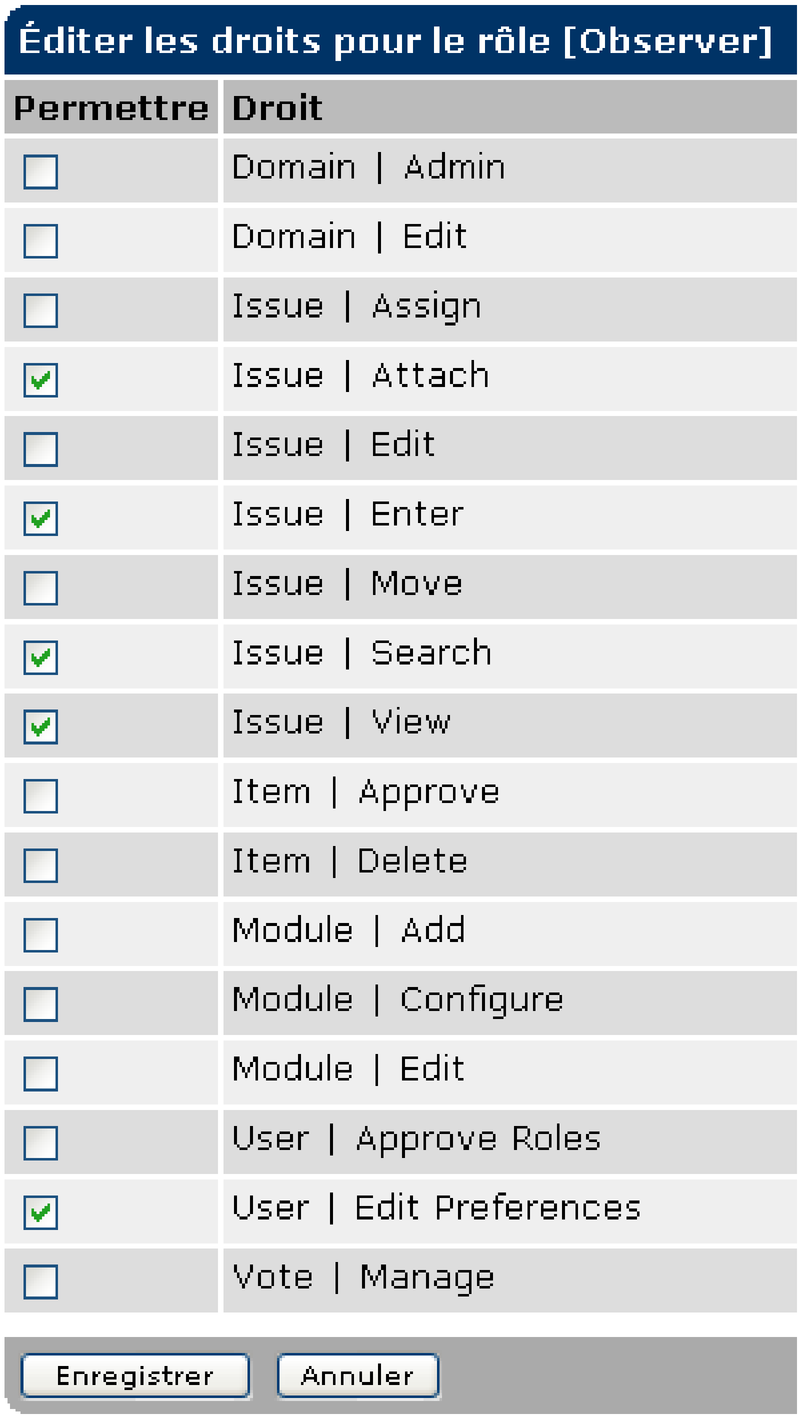
These tables are used to authenticate and manage the Scarab users, their roles and permissions in the different modules.

The tables in this part of the model are:

- **`turbine_user`** : stores data about the Scarab users, their logins, passwords, etc..
- **`turbine_role`** : stores the list of roles. Roles are common to all Scarab modules.
- **`turbine_permission`** : stores the different permissions defined in the Scarab application.
- **`turbine_group`** : this table may be required by Turbine at runtime but it is not used by Scarab. It is always empty.
- The **`turbine_user_group_role`** and **`turbine_role_permission`** tables are many-to-many relations. Their role is explained in the ERD below:



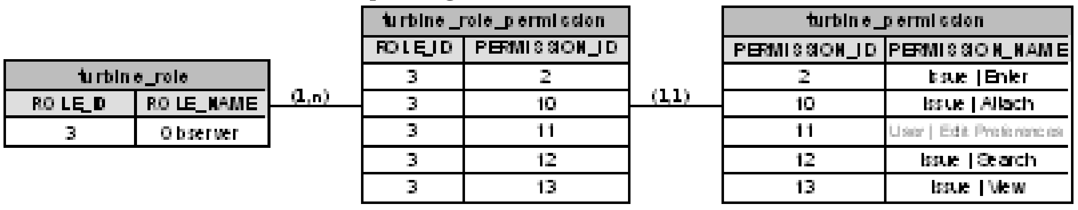
The following screenshot shows the definition of roles in Scarab (with the example data):



Éditer les droits pour le rôle [Observer]

Permettre	Droit
☐	Domain | Admin
☐	Domain | Edit
☐	Issue | Assign
✓	Issue | Attach
☐	Issue | Edit
✓	Issue | Enter
☐	Issue | Move
✓	Issue | Search
✓	Issue | View
☐	Item | Approve
☐	Item | Delete
☐	Module | Add
☐	Module | Configure
☐	Module | Edit
☐	User | Approve Roles
✓	User | Edit Preferences
☐	Vote | Manage

This is an illustration of the corresponding data model:



turbine_role

ROLE_ID	ROLE_NAME
3	Observer

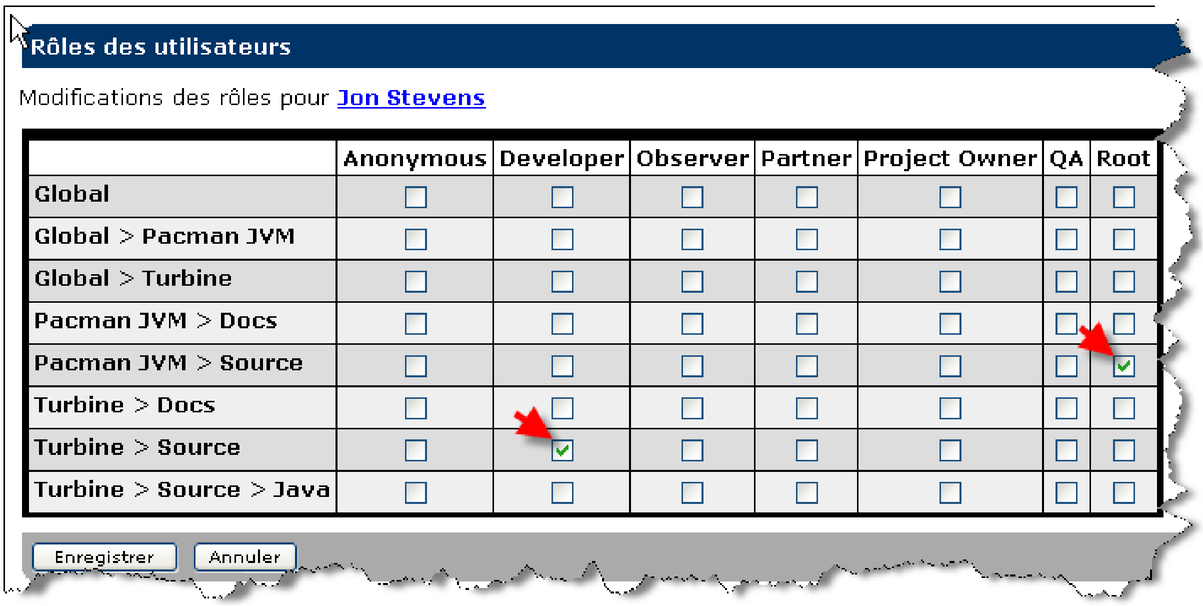
turbine_role_permission

ROLE_ID	PERMISSION_ID
3	2
3	10
3	11
3	12
3	13

turbine_permission

PERMISSION_ID	PERMISSION_NAME
2	Issue | Enter
10	Issue | Attach
11	User | Edit Preferences
12	Issue | Search
13	Issue | View

The following screenshot shows the definition of a user's roles (in the example data):

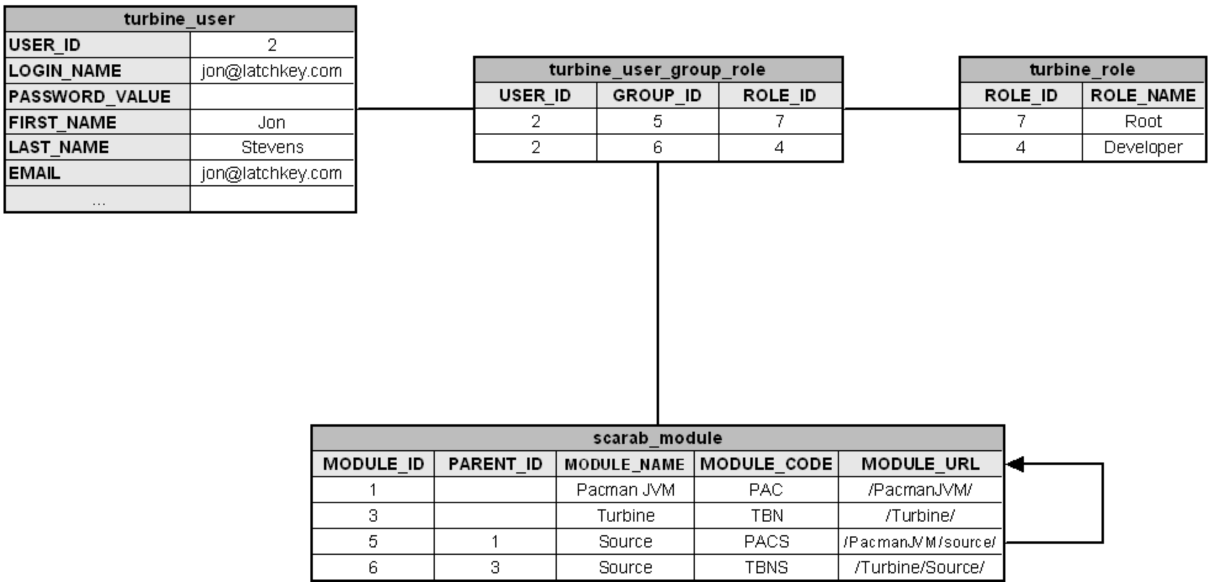


Rôles des utilisateurs

Modifications des rôles pour Jon Stevens

	Anonymous	Developer	Observer	Partner	Project Owner	QA	Root
Global	☐	☐	☐	☐	☐	☐	☐
Global > Pacman JVM	☐	☐	☐	☐	☐	☐	☐
Global > Turbine	☐	☐	☐	☐	☐	☐	☐
Pacman JVM > Docs	☐	☐	☐	☐	☐	☐	☐
Pacman JVM > Source	☐	☐	☐	☐	☐	☐	✓
Turbine > Docs	☐	☐	☐	☐	☐	☐	☐
Turbine > Source	☐	✓	☐	☐	☐	☐	☐
Turbine > Source > Java	☐	☐	☐	☐	☐	☐	☐

This is an illustration of the corresponding data model:



turbine_user	
USER_ID	2
LOGIN_NAME	jon@latchkey.com
PASSWORD_VALUE	
FIRST_NAME	Jon
LAST_NAME	Stevens
EMAIL	jon@latchkey.com
...	

turbine_user_group_role

USER_ID	GROUP_ID	ROLE_ID
2	5	7
2	6	4

turbine_role

ROLE_ID	ROLE_NAME
7	Root
4	Developer

scarab_module

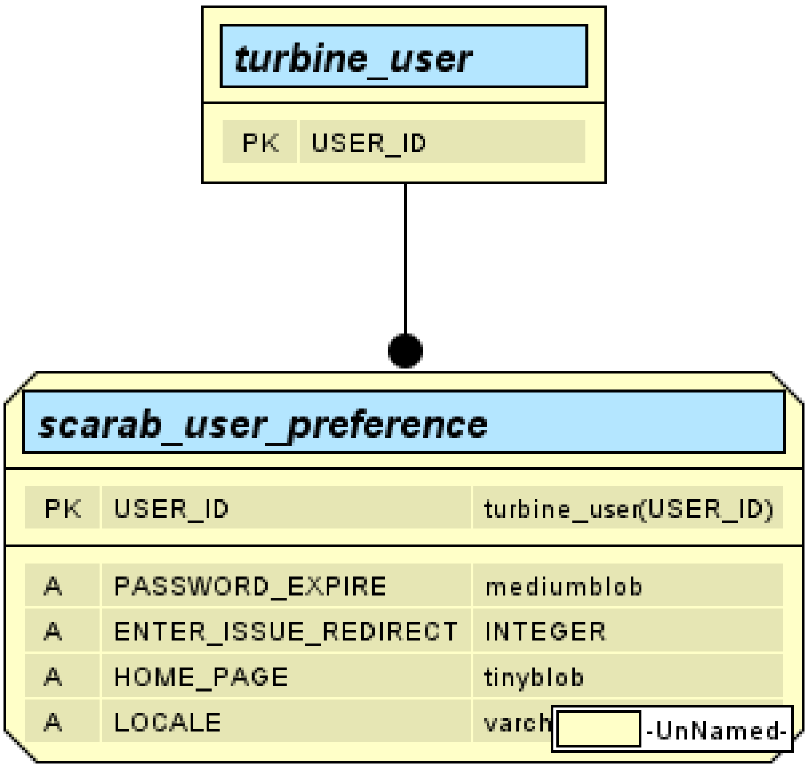
MODULE_ID	PARENT_ID	MODULE_NAME	MODULE_CODE	MODULE_URL
1		Pacman JVM	PAC	/PacmanJVM/
3		Turbine	TBN	/Turbine/
5	1	Source	PACS	/PacmanJVM/source/
6	3	Source	TBNS	/Turbine/Source/

Scarab tables

Entities related to users

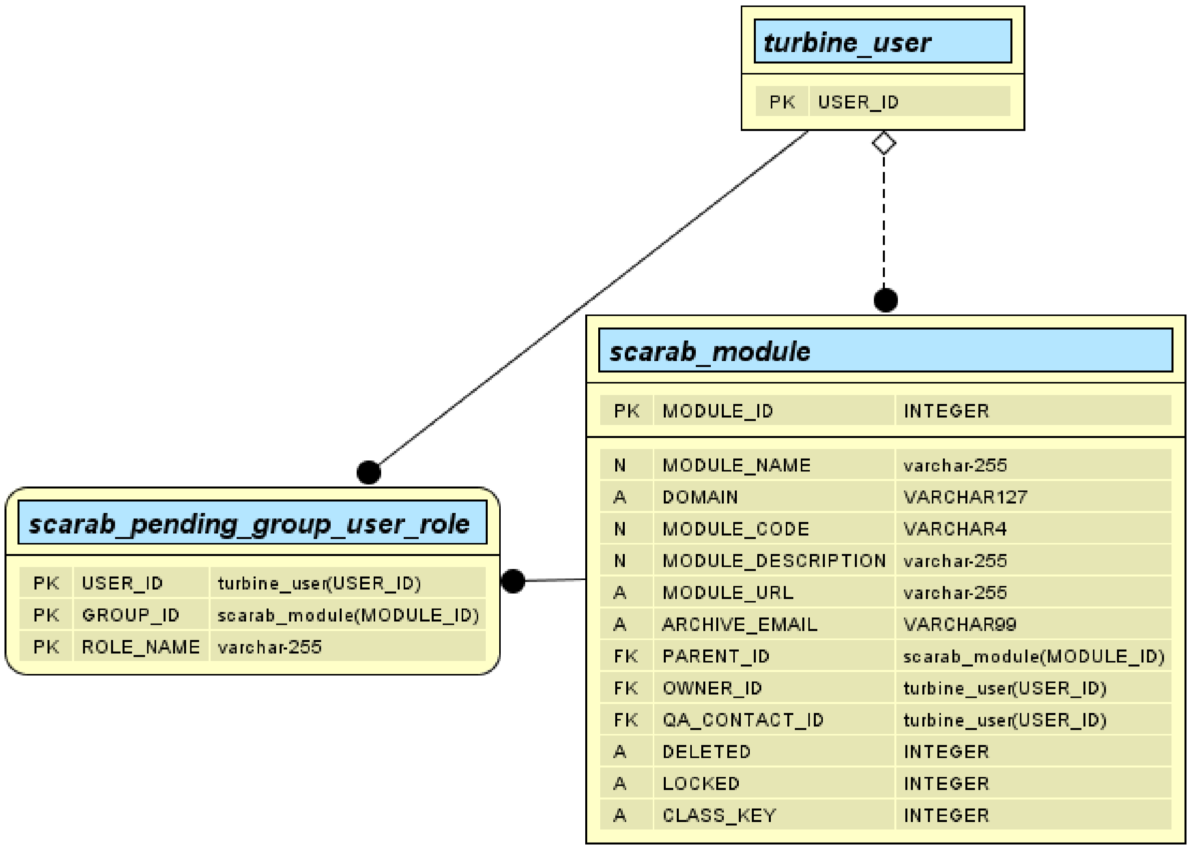
Before entering the heart of the Scarab data model, let us mention two series of simple entities related to the users as defined just above.

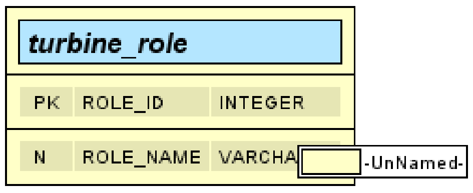
Preferences


turbine_user
PK USER_ID
scarab_user_preference
PK USER_ID turbine_user(USER_ID)
A PASSWORD_EXPIRE mediumblob
A ENTER_ISSUE_REDIRECT INTEGER
A HOME_PAGE tinyblob
A LOCALE varch
-UnNamed-

Pending role demands

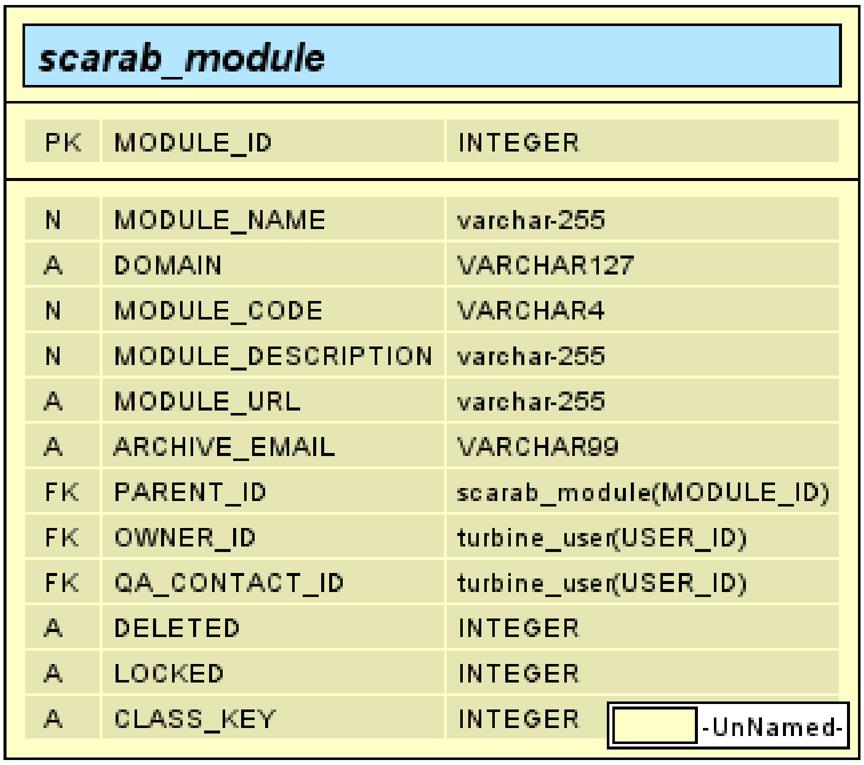
The `scarab_pending_group_user_role` table stores temporarily the role demands as supplied by users. These demands must be approved by the module administrator.


turbine_user
PK USER_ID
scarab_module
PK MODULE_ID INTEGER
N MODULE_NAME varchar-255
A DOMAIN VARCHAR127
N MODULE_CODE VARCHAR4
N MODULE_DESCRIPTION varchar-255
A MODULE_URL varchar-255
A ARCHIVE_EMAIL VARCHAR99
FK PARENT_ID scarab_module(MODULE_ID)
FK OWNER_ID turbine_user(USER_ID)
FK QA_CONTACT_ID turbine_user(USER_ID)
A DELETED INTEGER
A LOCKED INTEGER
A CLASS_KEY INTEGER
scarab_pending_group_user_role
PK USER_ID turbine_user(USER_ID)
PK GROUP_ID scarab_module(MODULE_ID)
PK ROLE_NAME varchar-255



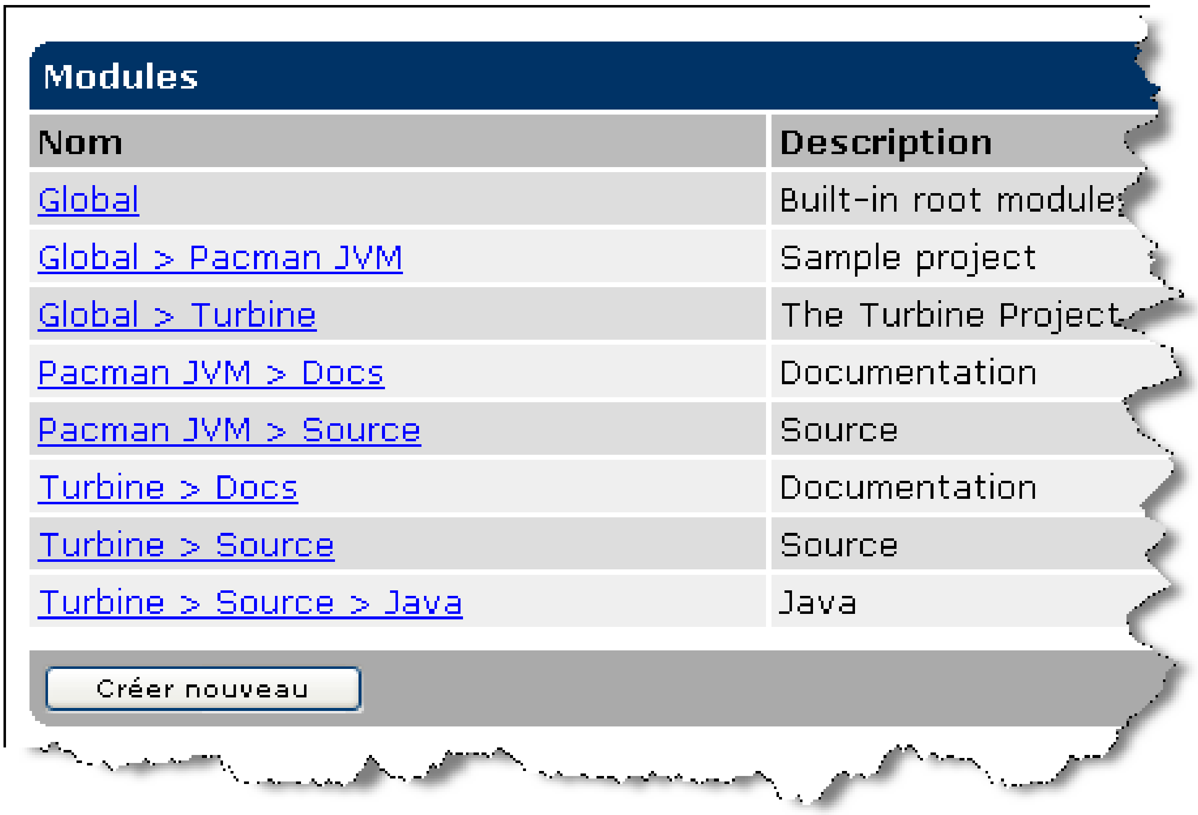
Modules

The list of modules and the corresponding data are stored in the `scarab_module` table.



scarab_module

PK	MODULE_ID	INTEGER
N	MODULE_NAME	varchar-255
A	DOMAIN	VARCHAR127
N	MODULE_CODE	VARCHAR4
N	MODULE_DESCRIPTION	varchar-255
A	MODULE_URL	varchar-255
A	ARCHIVE_EMAIL	VARCHAR99
FK	PARENT_ID	scarab_module(MODULE_ID)
FK	OWNER_ID	turbine_user(USER_ID)
FK	QA_CONTACT_ID	turbine_user(USER_ID)
A	DELETED	INTEGER
A	LOCKED	INTEGER
A	CLASS_KEY	INTEGER



Modules

Nom	Description
Global	Built-in root module
Global > Pacman JVM	Sample project
Global > Turbine	The Turbine Project
Pacman JVM > Docs	Documentation
Pacman JVM > Source	Source
Turbine > Docs	Documentation
Turbine > Source	Source
Turbine > Source > Java	Java

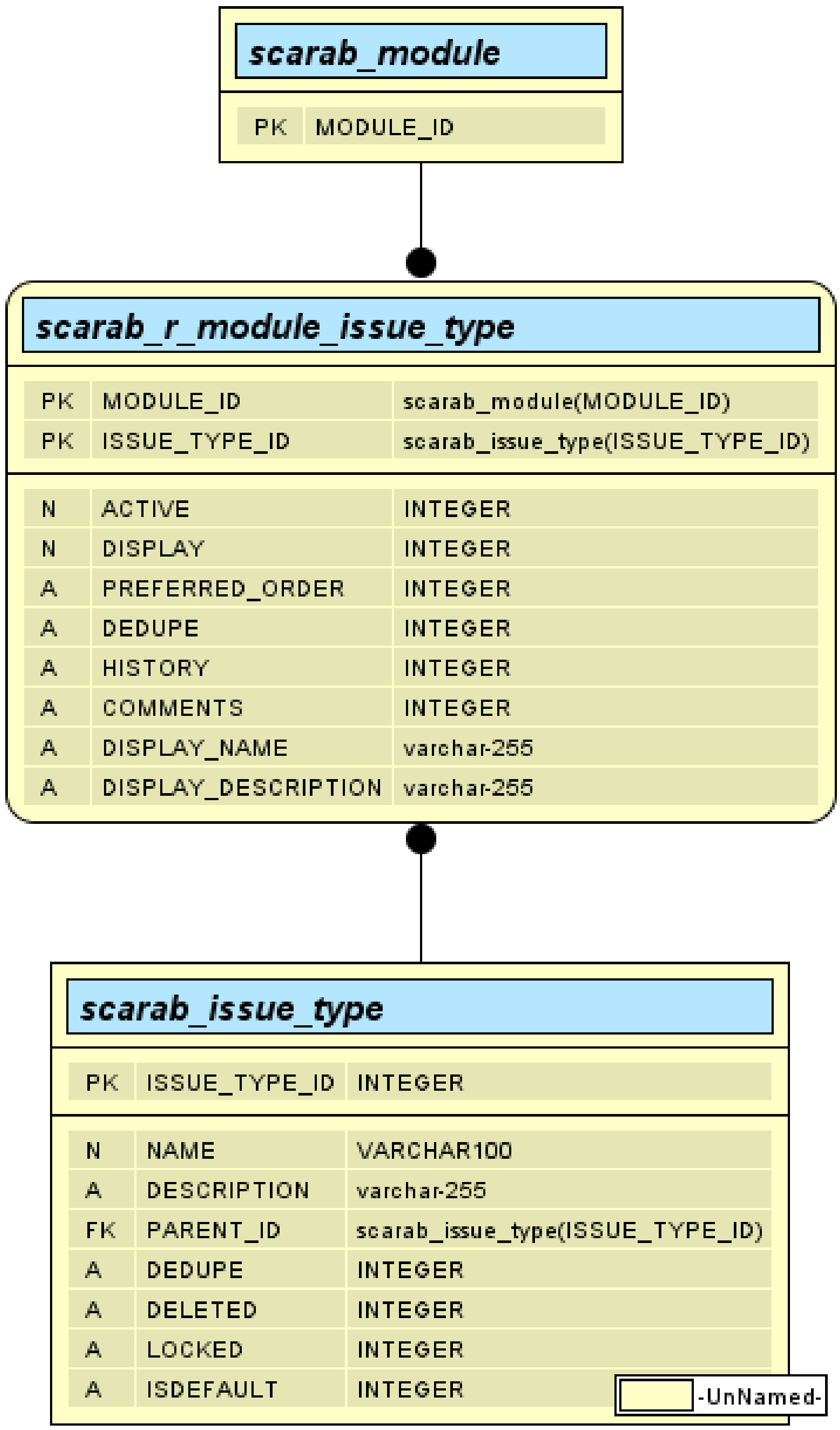
Tip

Our QA manager asked one day for a table that would give a correspondance between the module codes and their names. The slight difficulty here is that the module name is often meaningful with the name of the parent module. It is easy to obtain this kind of table from SCARAB_MODULE with the following SQL request:

```
select M1.MODULE_CODE, M2.MODULE_NAME, M1.MODULE_NAME from SCARAB_MODULE M1, SCARAB_MODULE M2 where M1.PARENT_ID = M2.MODULE_ID order by M1.MODULE_CODE;
```

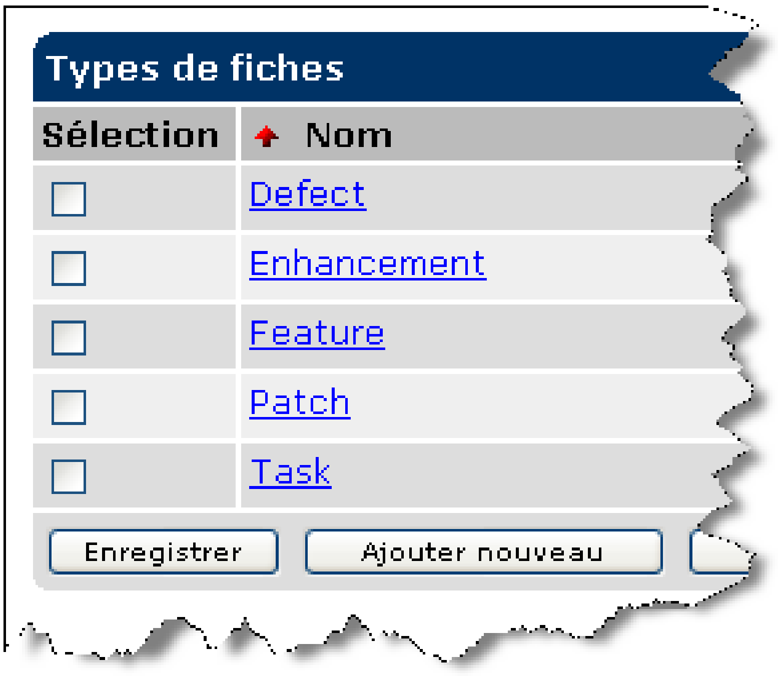
There it is!

Issue types


scarab_module
PK MODULE_ID
scarab_r_module_issue_type
PK MODULE_ID scarab_module(MODULE_ID)
PK ISSUE_TYPE_ID scarab_issue_type(ISSUE_TYPE_ID)
N ACTIVE INTEGER
N DISPLAY INTEGER
A PREFERRED_ORDER INTEGER
A DEDUPE INTEGER
A HISTORY INTEGER
A COMMENTS INTEGER
A DISPLAY_NAME varchar-255
A DISPLAY_DESCRIPTION varchar-255
scarab_issue_type
PK ISSUE_TYPE_ID INTEGER
N NAME VARCHAR100
A DESCRIPTION varchar-255
FK PARENT_ID scarab_issue_type(ISSUE_TYPE_ID)
A DEDUPE INTEGER
A DELETED INTEGER
A LOCKED INTEGER
A ISDEFAULT INTEGER
-UnNamed-

Issue types are stored in the `scarab_issue_type` table. Modules and issue types are related to each other in a many-to-many relationship through the `scarab_r_module_issue_type` relation table.

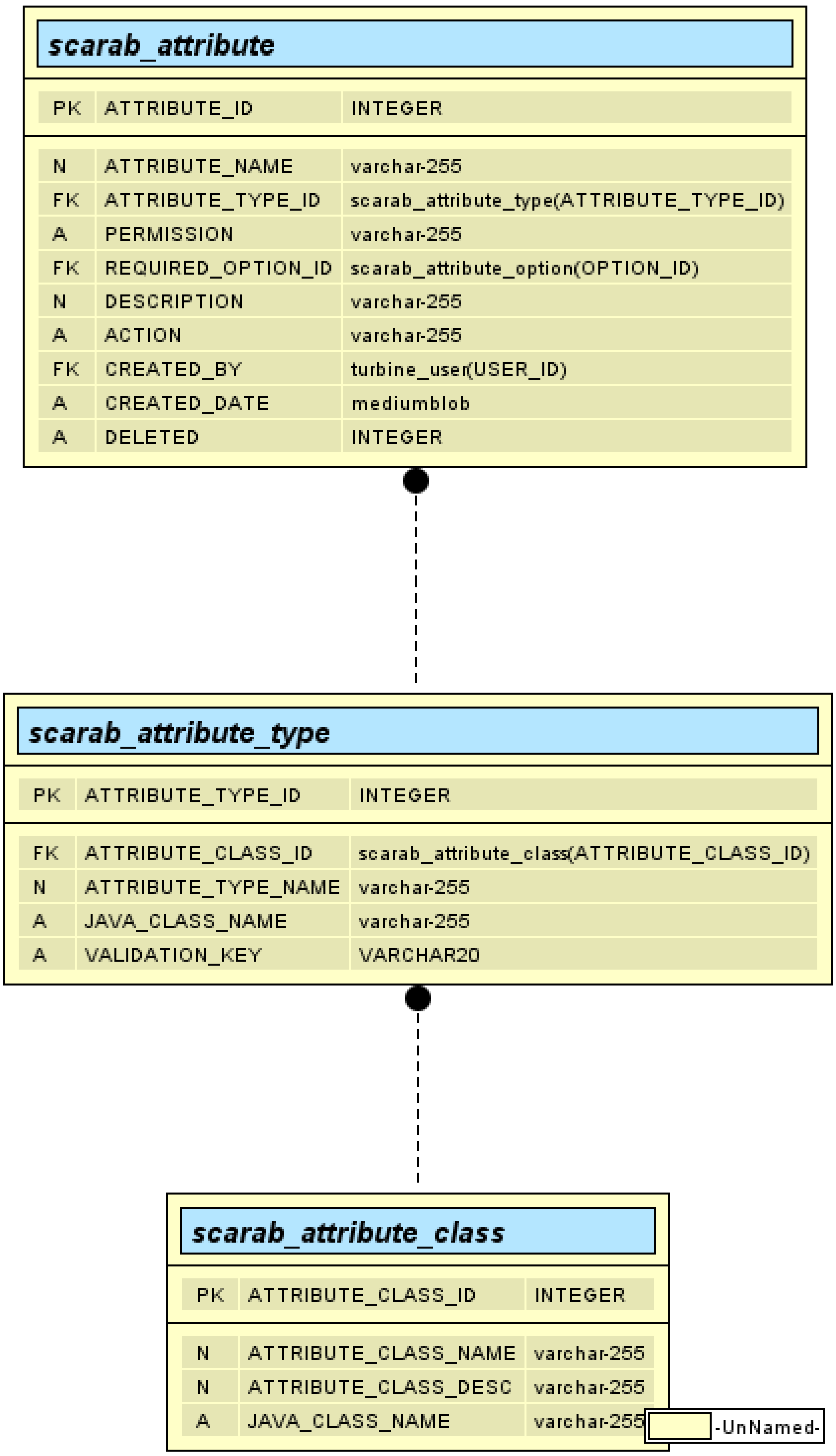
The global issue types, as illustrated below, are related to the "Global" module (the ID of this module is always 0).



Types de fiches

Sélection	Nom
☐	Defect
☐	Enhancement
☐	Feature
☐	Patch
☐	Task

Attributes



scarab_attribute

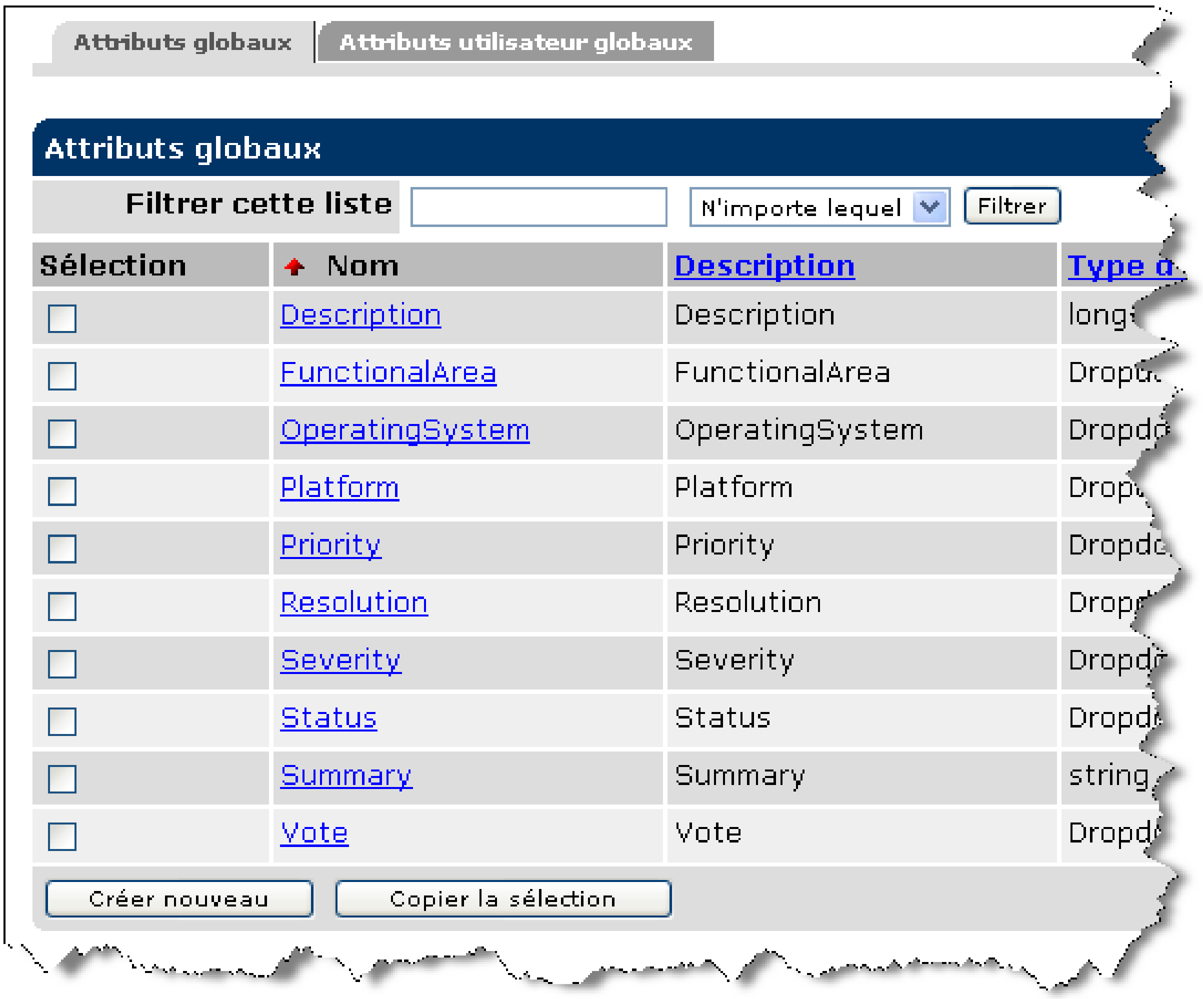
PK	ATTRIBUTE_ID	INTEGER
N	ATTRIBUTE_NAME	varchar-255
FK	ATTRIBUTE_TYPE_ID	scarab_attribute_type(ATTRIBUTE_TYPE_ID)
A	PERMISSION	varchar-255
FK	REQUIRED_OPTION_ID	scarab_attribute_option(OPTION_ID)
N	DESCRIPTION	varchar-255
A	ACTION	varchar-255
FK	CREATED_BY	turbine_user(USER_ID)
A	CREATED_DATE	mediumblob
A	DELETED	INTEGER

scarab_attribute_type

PK	ATTRIBUTE_TYPE_ID	INTEGER
FK	ATTRIBUTE_CLASS_ID	scarab_attribute_class(ATTRIBUTE_CLASS_ID)
N	ATTRIBUTE_TYPE_NAME	varchar-255
A	JAVA_CLASS_NAME	varchar-255
A	VALIDATION_KEY	VARCHAR20

scarab_attribute_class

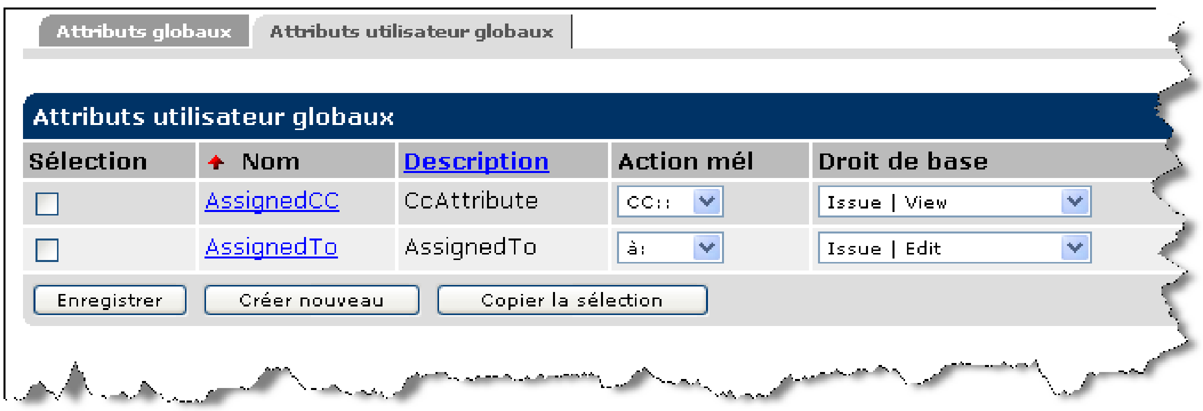
PK	ATTRIBUTE_CLASS_ID	INTEGER
N	ATTRIBUTE_CLASS_NAME	varchar-255
N	ATTRIBUTE_CLASS_DESC	varchar-255
A	JAVA_CLASS_NAME	varchar-255



Attributs globaux

Filtrer cette liste [ ] N'importe lequel Filtrer

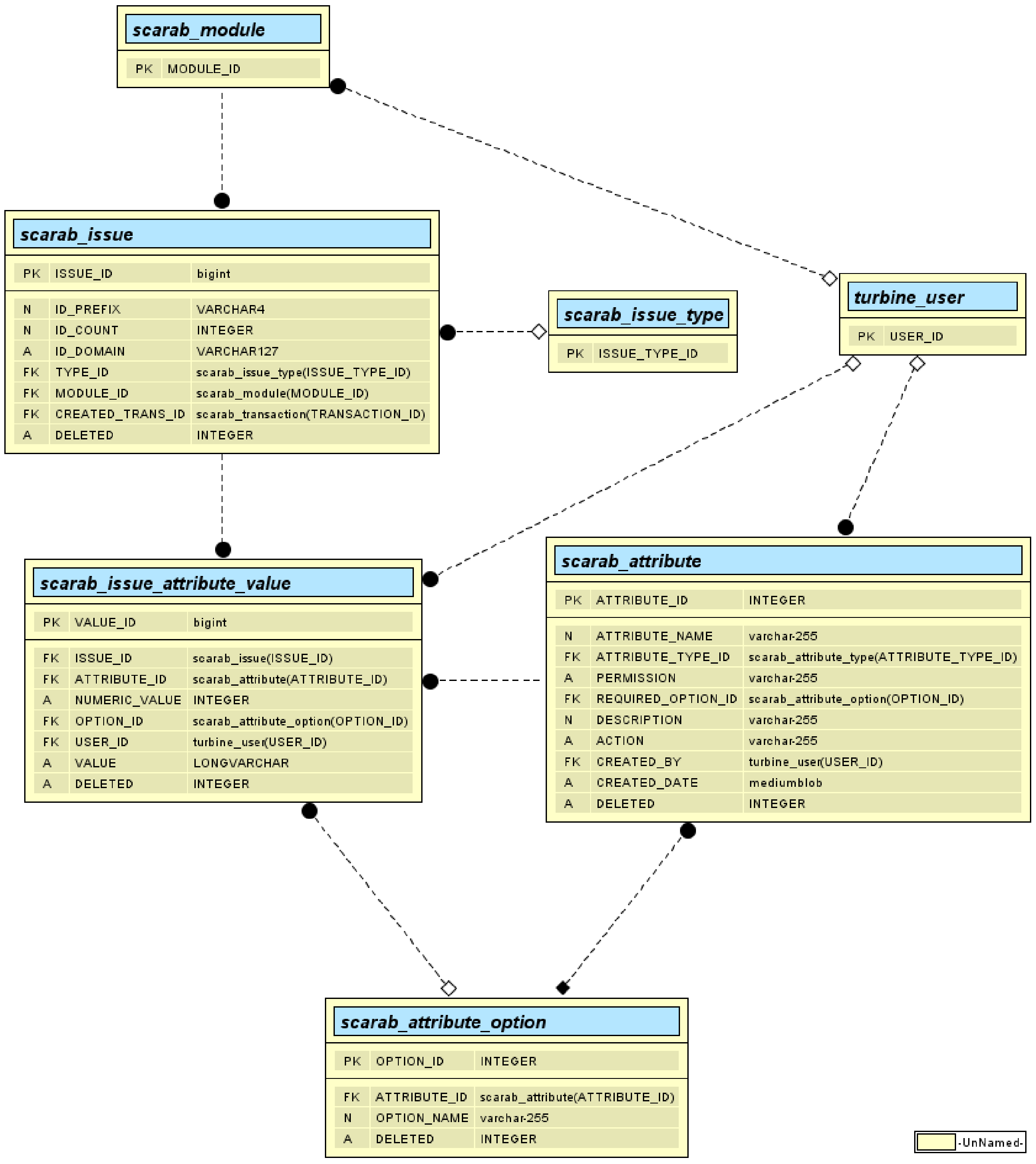
Sélection	Nom	Description	Type d
☐	Description	Description	long
☐	FunctionalArea	FunctionalArea	Dropd
☐	OperatingSystem	OperatingSystem	Dropdo
☐	Platform	Platform	Drop
☐	Priority	Priority	Dropdo
☐	Resolution	Resolution	Drop
☐	Severity	Severity	Dropdr
☐	Status	Status	Dropd
☐	Summary	Summary	string
☐	Vote	Vote	Dropd



Attributs utilisateur globaux

Sélection	Nom	Description	Action mél	Droit de base
☐	AssignedCC	CcAttribute	CC::	Issue | View
☐	AssignedTo	AssignedTo	à:	Issue | Edit

Issues



scarab_module

PK	MODULE_ID	

scarab_issue

PK	ISSUE_ID	bigint
N	ID_PREFIX	VARCHAR4
N	ID_COUNT	INTEGER
A	ID_DOMAIN	VARCHAR127
FK	TYPE_ID	scarab_issue_type(ISSUE_TYPE_ID)
FK	MODULE_ID	scarab_module(MODULE_ID)
FK	CREATED_TRANS_ID	scarab_transaction(TRANSACTION_ID)
A	DELETED	INTEGER

scarab_issue_type

PK	ISSUE_TYPE_ID	

turbine_user

PK	USER_ID	

scarab_issue_attribute_value

PK	VALUE_ID	bigint
FK	ISSUE_ID	scarab_issue(ISSUE_ID)
FK	ATTRIBUTE_ID	scarab_attribute(ATTRIBUTE_ID)
A	NUMERIC_VALUE	INTEGER
FK	OPTION_ID	scarab_attribute_option(OPTION_ID)
FK	USER_ID	turbine_user(USER_ID)
A	VALUE	LONGVARCHAR
A	DELETED	INTEGER

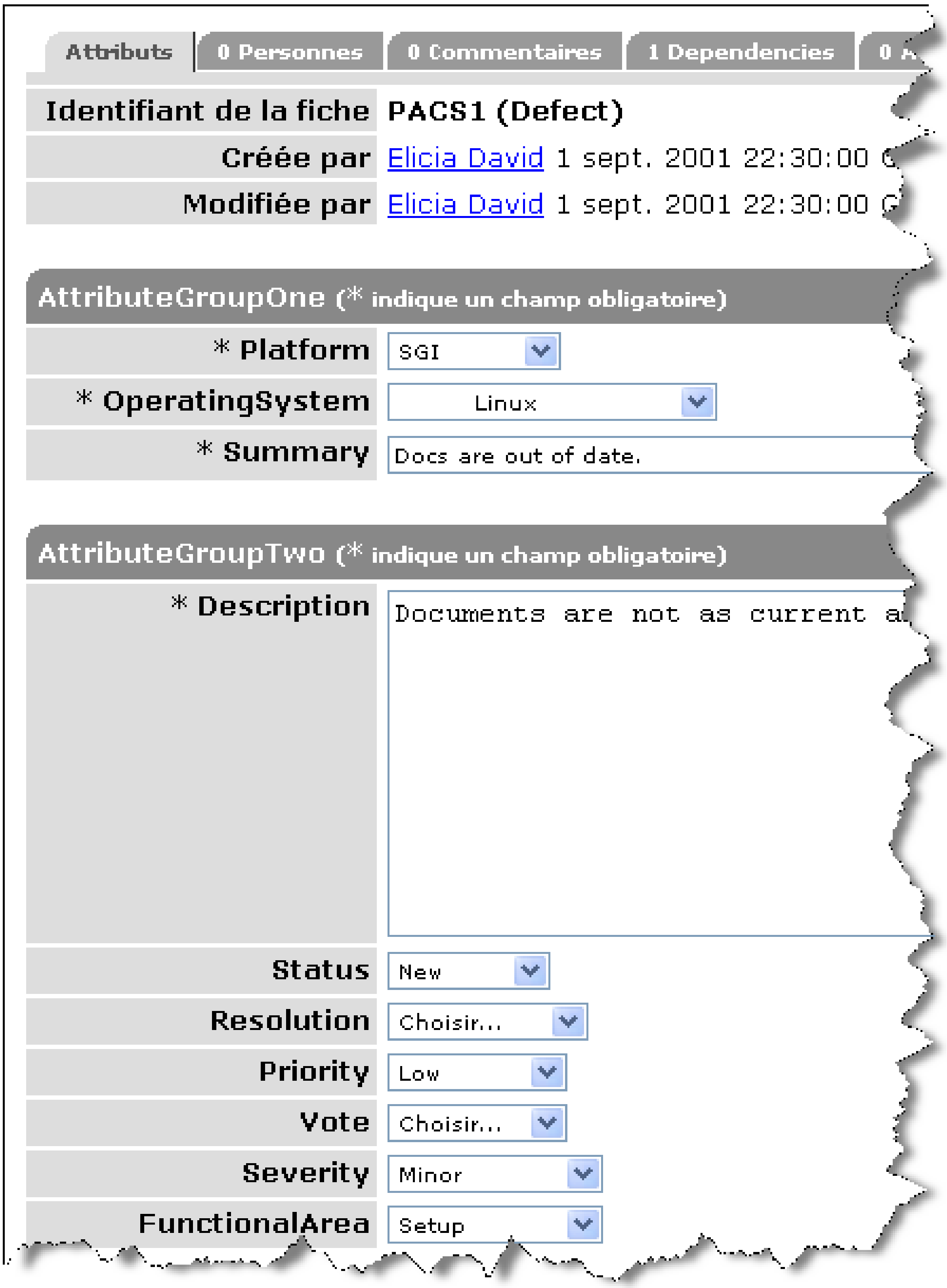
scarab_attribute

PK	ATTRIBUTE_ID	INTEGER
N	ATTRIBUTE_NAME	varchar255
FK	ATTRIBUTE_TYPE_ID	scarab_attribute_type(ATTRIBUTE_TYPE_ID)
A	PERMISSION	varchar255
FK	REQUIRED_OPTION_ID	scarab_attribute_option(OPTION_ID)
N	DESCRIPTION	varchar255
A	ACTION	varchar255
FK	CREATED_BY	turbine_user(USER_ID)
A	CREATED_DATE	mediumblob
A	DELETED	INTEGER

scarab_attribute_option

PK	OPTION_ID	INTEGER
FK	ATTRIBUTE_ID	scarab_attribute(ATTRIBUTE_ID)
N	OPTION_NAME	varchar255
A	DELETED	INTEGER

Attributes



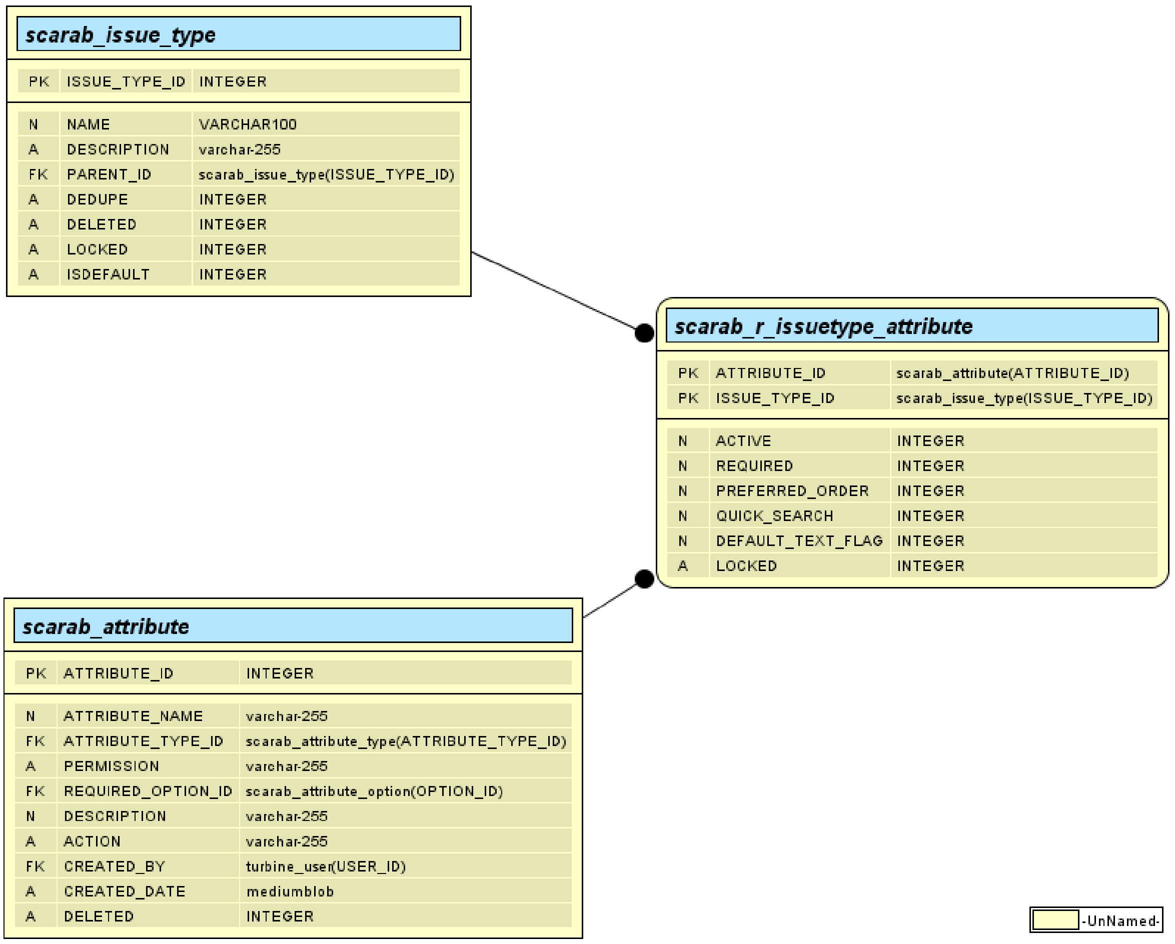
Identifiant de la fiche	PACS1 (Defect)
Créée par	Elicia David 1 sept. 2001 22:30:00 G
Modifiée par	Elicia David 1 sept. 2001 22:30:00 G

AttributeGroupOne (* indique un champ obligatoire)

* Platform	SGI
* OperatingSystem	Linux
* Summary	Docs are out of date.

AttributeGroupTwo (* indique un champ obligatoire)

* Description	Documents are not as current a
Status	New
Resolution	Choisir...
Priority	Low
Vote	Choisir...
Severity	Minor
FunctionalArea	Setup



scarab_issue_type

PK	ISSUE_TYPE_ID	INTEGER
N	NAME	VARCHAR100
A	DESCRIPTION	varchar-255
FK	PARENT_ID	scarab_issue_type(ISSUE_TYPE_ID)
A	DEDUPE	INTEGER
A	DELETED	INTEGER
A	LOCKED	INTEGER
A	ISDEFAULT	INTEGER

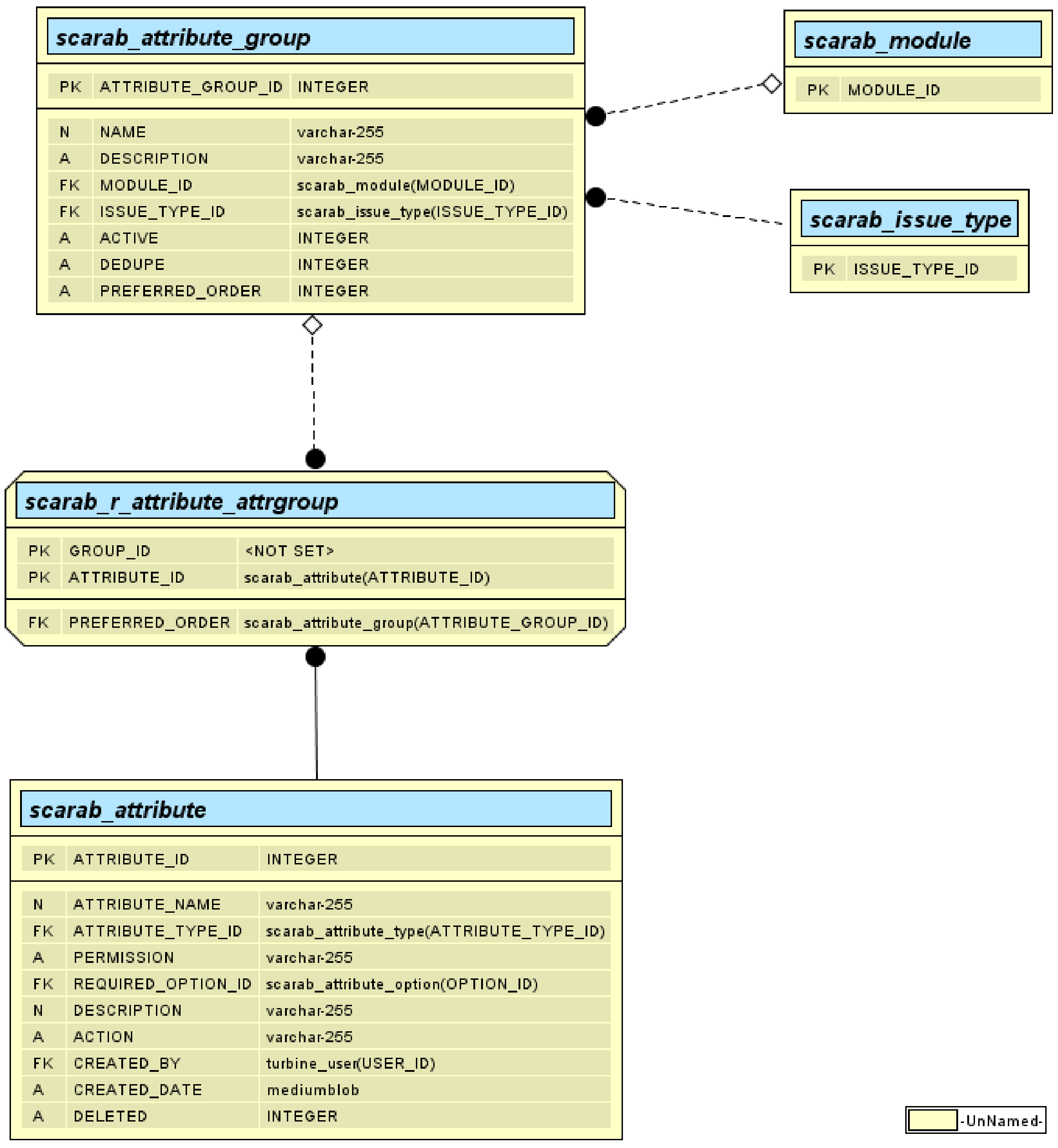
scarab_r_issuetype_attribute

PK	ATTRIBUTE_ID	scarab_attribute(ATTRIBUTE_ID)
PK	ISSUE_TYPE_ID	scarab_issue_type(ISSUE_TYPE_ID)
N	ACTIVE	INTEGER
N	REQUIRED	INTEGER
N	PREFERRED_ORDER	INTEGER
N	QUICK_SEARCH	INTEGER
N	DEFAULT_TEXT_FLAG	INTEGER
A	LOCKED	INTEGER

scarab_attribute

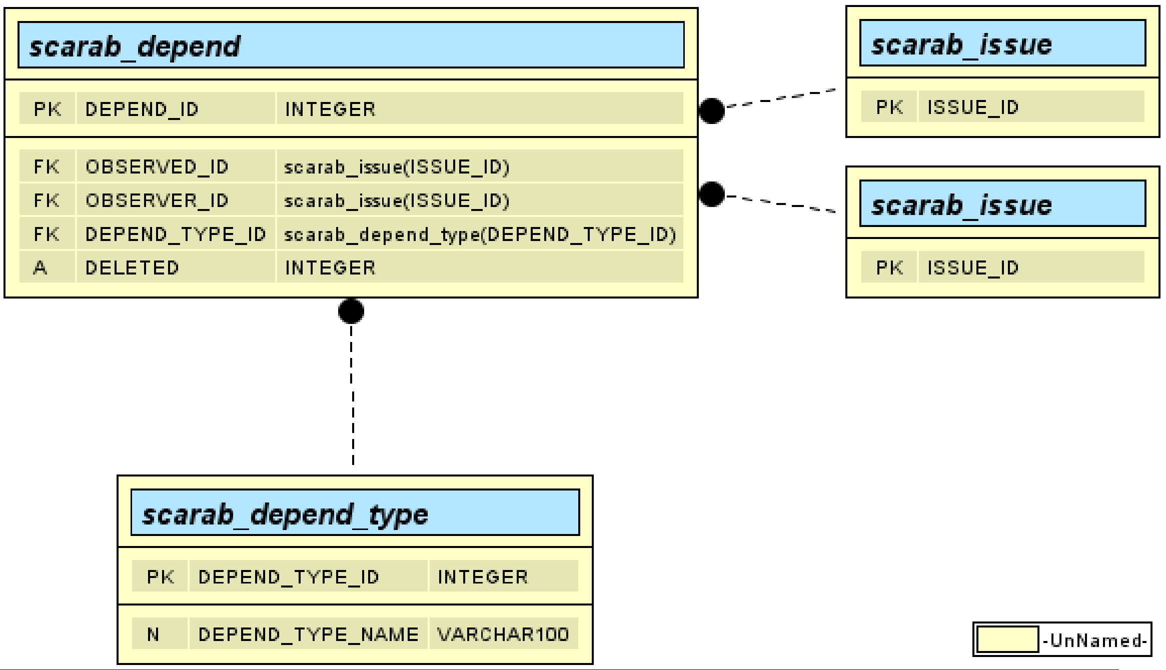
PK	ATTRIBUTE_ID	INTEGER
N	ATTRIBUTE_NAME	varchar-255
FK	ATTRIBUTE_TYPE_ID	scarab_attribute_type(ATTRIBUTE_TYPE_ID)
A	PERMISSION	varchar-255
FK	REQUIRED_OPTION_ID	scarab_attribute_option(OPTION_ID)
N	DESCRIPTION	varchar-255
A	ACTION	varchar-255
FK	CREATED_BY	turbine_user(USER_ID)
A	CREATED_DATE	mediumblob
A	DELETED	INTEGER

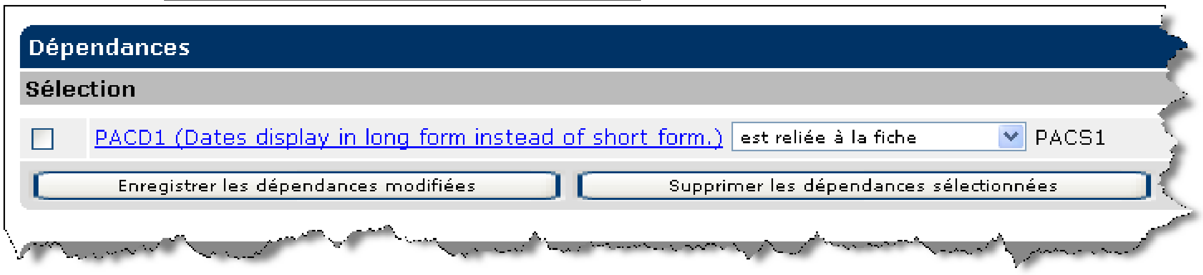
Attribute groups


scarab_attribute_group
PK ATTRIBUTE_GROUP_ID INTEGER
N NAME varchar-255
A DESCRIPTION varchar-255
FK MODULE_ID scarab_module(MODULE_ID)
FK ISSUE_TYPE_ID scarab_issue_type(ISSUE_TYPE_ID)
A ACTIVE INTEGER
A DEDUPE INTEGER
A PREFERRED_ORDER INTEGER
scarab_module
PK MODULE_ID
scarab_issue_type
PK ISSUE_TYPE_ID
scarab_r_attribute_attrgroup
PK GROUP_ID <NOT SET>
PK ATTRIBUTE_ID scarab_attribute(ATTRIBUTE_ID)
FK PREFERRED_ORDER scarab_attribute_group(ATTRIBUTE_GROUP_ID)
scarab_attribute
PK ATTRIBUTE_ID INTEGER
N ATTRIBUTE_NAME varchar-255
FK ATTRIBUTE_TYPE_ID scarab_attribute_type(ATTRIBUTE_TYPE_ID)
A PERMISSION varchar-255
FK REQUIRED_OPTION_ID scarab_attribute_option(OPTION_ID)
N DESCRIPTION varchar-255
A ACTION varchar-255
FK CREATED_BY turbine_user(USER_ID)
A CREATED_DATE mediumblob
A DELETED INTEGER
-UnNamed-

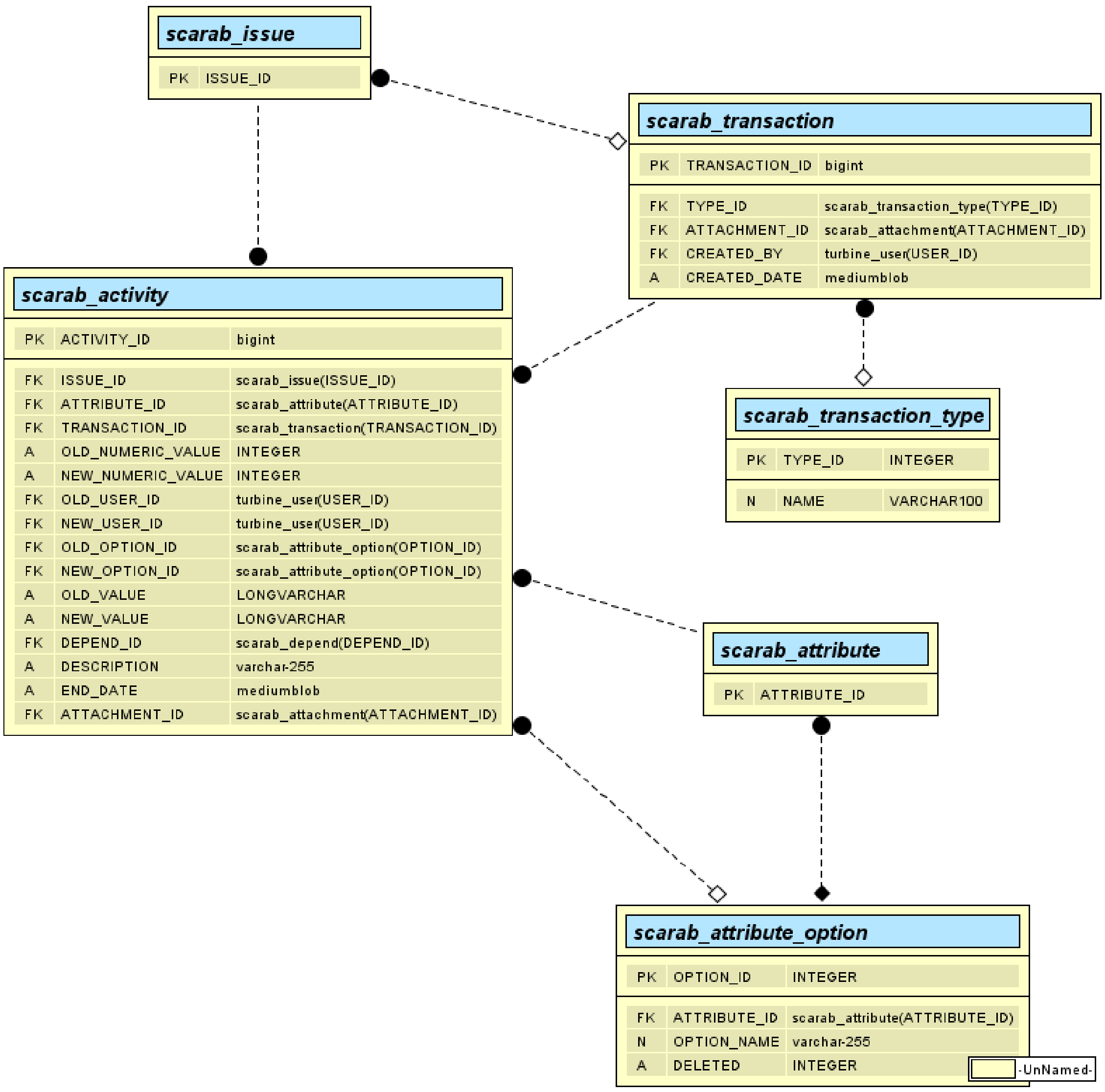
Attachments

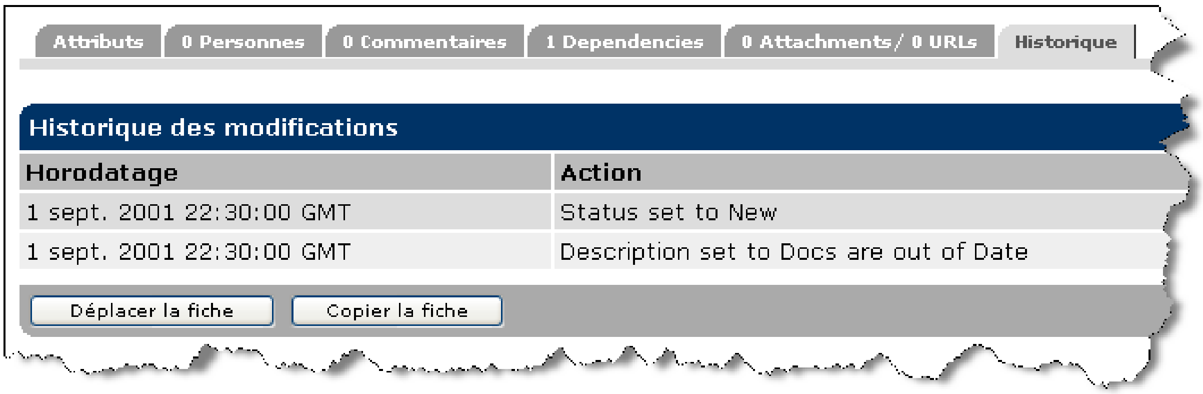
Dependencies


scarab_depend
PK DEPEND_ID INTEGER
FK OBSERVED_ID scarab_issue(ISSUE_ID)
FK OBSERVER_ID scarab_issue(ISSUE_ID)
FK DEPEND_TYPE_ID scarab_depend_type(DEPEND_TYPE_ID)
A DELETED INTEGER
scarab_issue
PK ISSUE_ID
scarab_issue
PK ISSUE_ID
scarab_depend_type
PK DEPEND_TYPE_ID INTEGER
N DEPEND_TYPE_NAME VARCHAR100
-UnNamed-


Dépendances
Sélection
PACD1 (Dates display in long form instead of short form.)
est reliée à la fiche
PACS1
Enregistrer les dépendances modifiées
Supprimer les dépendances sélectionnées

History

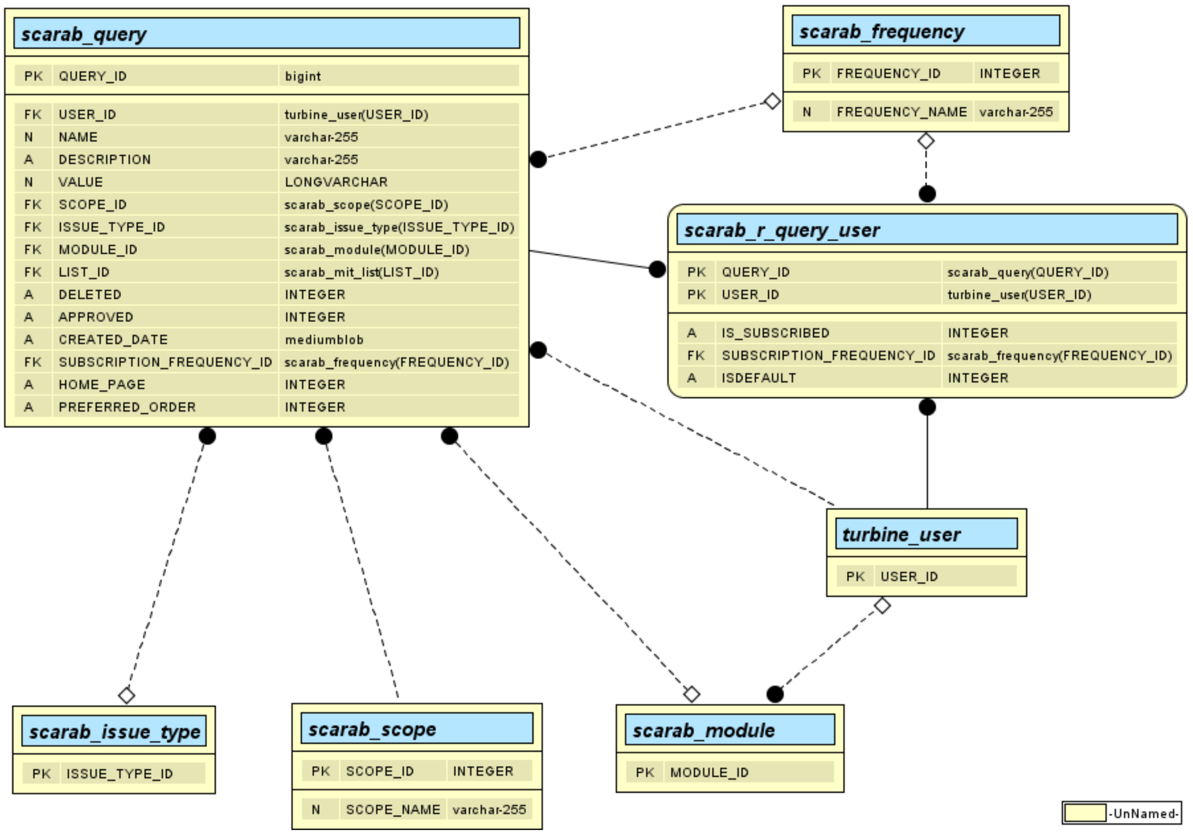

scarab_issue
PK ISSUE_ID
scarab_transaction
PK TRANSACTION_ID bigint
FK TYPE_ID scarab_transaction_type(TYPE_ID)
FK ATTACHMENT_ID scarab_attachment(ATTACHMENT_ID)
FK CREATED_BY turbine_user(USER_ID)
A CREATED_DATE mediumblob
scarab_activity
PK ACTIVITY_ID bigint
FK ISSUE_ID scarab_issue(ISSUE_ID)
FK ATTRIBUTE_ID scarab_attribute(ATTRIBUTE_ID)
FK TRANSACTION_ID scarab_transaction(TRANSACTION_ID)
A OLD_NUMERIC_VALUE INTEGER
A NEW_NUMERIC_VALUE INTEGER
FK OLD_USER_ID turbine_user(USER_ID)
FK NEW_USER_ID turbine_user(USER_ID)
FK OLD_OPTION_ID scarab_attribute_option(OPTION_ID)
FK NEW_OPTION_ID scarab_attribute_option(OPTION_ID)
A OLD_VALUE LONGVARCHAR
A NEW_VALUE LONGVARCHAR
FK DEPEND_ID scarab_depend(DEPEND_ID)
A DESCRIPTION varchar-255
A END_DATE mediumblob
FK ATTACHMENT_ID scarab_attachment(ATTACHMENT_ID)
scarab_transaction_type
PK TYPE_ID INTEGER
N NAME VARCHAR100
scarab_attribute
PK ATTRIBUTE_ID
scarab_attribute_option
PK OPTION_ID INTEGER
FK ATTRIBUTE_ID scarab_attribute(ATTRIBUTE_ID)
N OPTION_NAME varchar-255
A DELETED INTEGER
-UnNamed-



Historique des modifications

Horodatage	Action
1 sept. 2001 22:30:00 GMT	Status set to New
1 sept. 2001 22:30:00 GMT	Description set to Docs are out of Date

Queries



scarab_query

PK	QUERY_ID	bigint
FK	USER_ID	turbine_user(USER_ID)
N	NAME	varchar-255
A	DESCRIPTION	varchar-255
N	VALUE	LONGVARCHAR
FK	SCOPE_ID	scarab_scope(SCOPE_ID)
FK	ISSUE_TYPE_ID	scarab_issue_type(ISSUE_TYPE_ID)
FK	MODULE_ID	scarab_module(MODULE_ID)
FK	LIST_ID	scarab_mit_list(LIST_ID)
A	DELETED	INTEGER
A	APPROVED	INTEGER
A	CREATED_DATE	mediumblob
FK	SUBSCRIPTION_FREQUENCY_ID	scarab_frequency(FREQUENCY_ID)
A	HOME_PAGE	INTEGER
A	PREFERRED_ORDER	INTEGER

scarab_frequency

PK	FREQUENCY_ID	INTEGER
N	FREQUENCY_NAME	varchar-255

scarab_r_query_user

PK	QUERY_ID	scarab_query(QUERY_ID)
PK	USER_ID	turbine_user(USER_ID)
A	IS_SUBSCRIBED	INTEGER
FK	SUBSCRIPTION_FREQUENCY_ID	scarab_frequency(FREQUENCY_ID)
A	ISDEFAULT	INTEGER

turbine_user

PK	USER_ID

scarab_issue_type

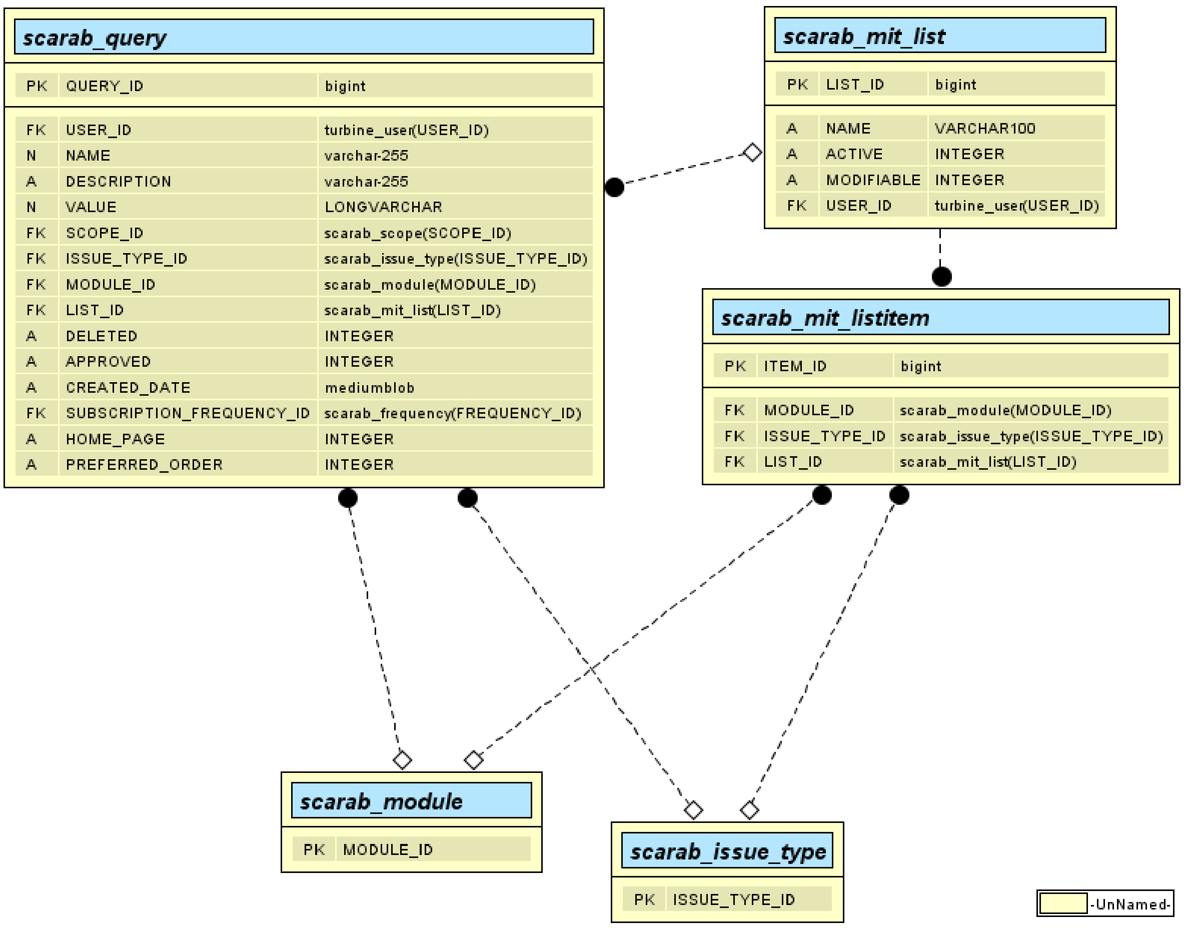
PK	ISSUE_TYPE_ID

scarab_scope

PK	SCOPE_ID	INTEGER
N	SCOPE_NAME	varchar-255

scarab_module

PK	MODULE_ID



scarab_query

PK	QUERY_ID	bigint
FK	USER_ID	turbine_user(USER_ID)
N	NAME	varchar-255
A	DESCRIPTION	varchar-255
N	VALUE	LONGVARCHAR
FK	SCOPE_ID	scarab_scope(SCOPE_ID)
FK	ISSUE_TYPE_ID	scarab_issue_type(ISSUE_TYPE_ID)
FK	MODULE_ID	scarab_module(MODULE_ID)
FK	LIST_ID	scarab_mit_list(LIST_ID)
A	DELETED	INTEGER
A	APPROVED	INTEGER
A	CREATED_DATE	mediumblob
FK	SUBSCRIPTION_FREQUENCY_ID	scarab_frequency(FREQUENCY_ID)
A	HOME_PAGE	INTEGER
A	PREFERRED_ORDER	INTEGER

scarab_mit_list

PK	LIST_ID	bigint
A	NAME	VARCHAR100
A	ACTIVE	INTEGER
A	MODIFIABLE	INTEGER
FK	USER_ID	turbine_user(USER_ID)

scarab_mit_listitem

PK	ITEM_ID	bigint
FK	MODULE_ID	scarab_module(MODULE_ID)
FK	ISSUE_TYPE_ID	scarab_issue_type(ISSUE_TYPE_ID)
FK	LIST_ID	scarab_mit_list(LIST_ID)

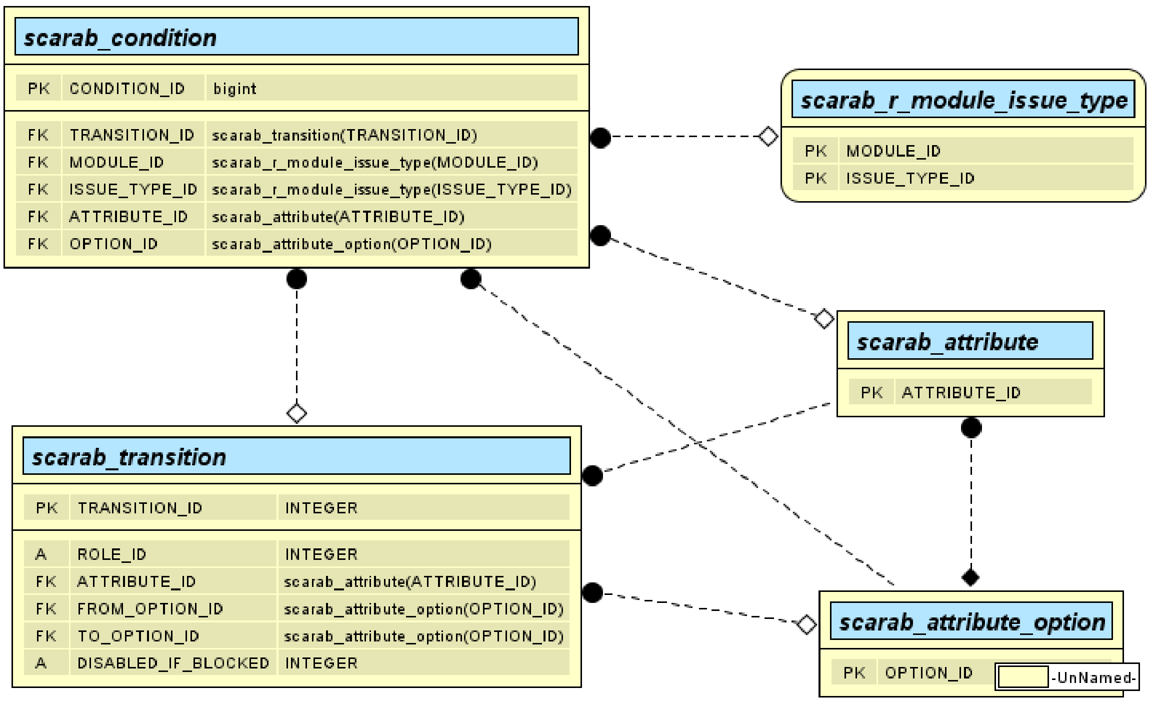
scarab_module

PK	MODULE_ID

scarab_issue_type

PK	ISSUE_TYPE_ID

Workflow



scarab_condition

PK	CONDITION_ID	bigint
FK	TRANSITION_ID	scarab_transition(TRANSITION_ID)
FK	MODULE_ID	scarab_r_module_issue_type(MODULE_ID)
FK	ISSUE_TYPE_ID	scarab_r_module_issue_type(ISSUE_TYPE_ID)
FK	ATTRIBUTE_ID	scarab_attribute(ATTRIBUTE_ID)
FK	OPTION_ID	scarab_attribute_option(OPTION_ID)

scarab_r_module_issue_type

PK	MODULE_ID
PK	ISSUE_TYPE_ID

scarab_attribute

PK	ATTRIBUTE_ID

scarab_transition

PK	TRANSITION_ID	INTEGER
A	ROLE_ID	INTEGER
FK	ATTRIBUTE_ID	scarab_attribute(ATTRIBUTE_ID)
FK	FROM_OPTION_ID	scarab_attribute_option(OPTION_ID)
FK	TO_OPTION_ID	scarab_attribute_option(OPTION_ID)
A	DISABLED_IF_BLOCKED	INTEGER

scarab_attribute_option

PK	OPTION_ID	-UnNamed-

Chapter 7. Workflow

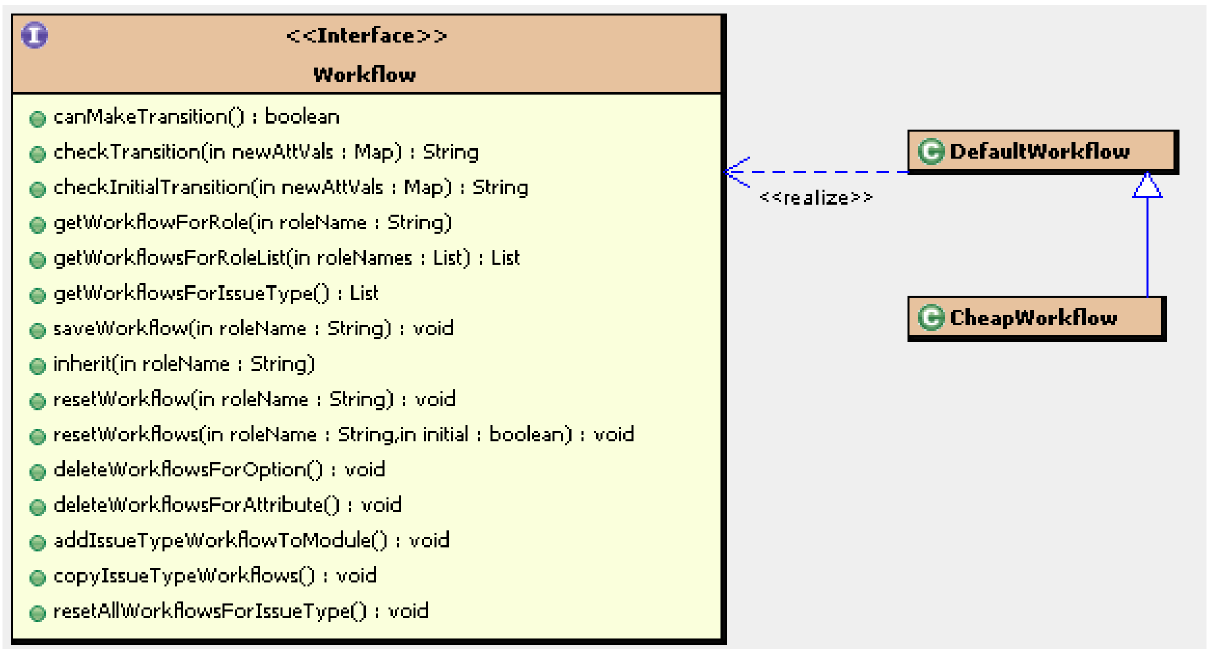
Introduction

As has been said in the *Workflow* chapter of the *Scarab User's Guide*, Scarab has no "hard-coded" workflow. When saving issues, all state transitions are allowed and there is no interaction with any external system ever.

Now let's have a look at how this is implemented.

Workflows in Scarab

At the programming level, a workflow is any java class that implements the `org.tigris.scarab.workflow.Workflow` interface (the source code of this interface is located under `src/java`).



<<Interface>>
Workflow
canMakeTransition() : boolean
checkTransition(in newAttVals : Map) : String
checkInitialTransition(in newAttVals : Map) : String
getWorkflowForRole(in roleName : String)
getWorkflowsForRoleList(in roleNames : List) : List
getWorkflowsForIssueType() : List
saveWorkflow(in roleName : String) : void
inherit(in roleName : String)
resetWorkflow(in roleName : String) : void
resetWorkflows(in roleName : String,in initial : boolean) : void
deleteWorkflowsForOption() : void
deleteWorkflowsForAttribute() : void
addIssueTypeWorkflowToModule() : void
copyIssueTypeWorkflows() : void
resetAllWorkflowsForIssueType() : void
<<realize>>
DefaultWorkflow
CheapWorkflow

The workflow is then instantiated dynamically at runtime by Turbine, according to a declaration in `Scarab.properties` (this file is located at runtime in the web application tree under `WEB-INF/conf`).

```
# The workflow tool
# scarab.workflow.classname=org.tigris.scarab.workflow.DefaultWorkflow
scarab.workflow.classname=org.tigris.scarab.workflow.CheapWorkflow
```

DefaultWorkflow

This was the default workflow for all versions of Scarab until milestone 19.

This workflow is implemented in the `org.tigris.scarab.workflow.DefaultWorkflow` class. This class is a stub, that does nothing and allows all transitions.

BasicWorkflow

This is the default workflow for all Scarab versions from milestone 20. The way it works from the user point of view is explained in the *User's Guide*.

This workflow is implemented in the `org.tigris.scarab.workflow.CheapWorkflow` class (againts the odds). It implements the *Workflow* interface indirectly by inheriting from the default workflow, `DefaultWorkflow` probably not to redefine certain default behaviours.

Defining a new workflow

To define a new workflow, using other mechanismes, communicating with one or several external systems or more adapted to the needs or processes of your organization, you just need to:

- write a java class that implements the `org.tigris.scarab.workflow.Workflow` interface (either directly or by inheriting from `org.tigris.scarab.workflow.DefaultWorkflow`);
- set the name of this class in the `Scarab.properties` mentioned above.

Using WfmOpen as a Scarab workflow engine

Someone wrote an interface between Scarab and the WfmOpen open-source workflow engine. This implementation, or the base of it, is currently stored in the SourceForge tracker at thie URL: http://sourceforge.net/tracker/index.php?func=detail&aid=961620&group_id=76143&atid=546206 [http://sourceforge.net/tracker/index.php?func=detail&aid=961620&group_id=76143&atid=546206l].

The status of this development is unknown.

Chapter 8. Accessing Scarab via XML-RPC

It is possible to access remotely the Scarab functionalities via XML-RPC.

An example is supplied in the Scarab sources (under `src/java`):

- la classe `org.tigris.scarab.util.SimpleHandler.java` (server, in Scarab) is used by the Scarab-Subversion integration (read the corresponding chapter in the *Scarab User's Guide*);
- la classe `org.tigris.scarab.util.SimpleHandlerClient.java` (client, in another application) is a simple example of using the server above.