MOMOCS

Model driven Modernisation of Complex Systems

DELIVERABLE # D51

DATA MODERNIZATION TOOL IMPLEMENTATION (XSM KNOWLEDGE BASE REPOSITORY)

Dissemination Level:	Public
Work package:	WP5
Lead Participant:	Atos Origin SAE
Contractual Delivery Date:	M17
Document status:	Final

Preparation Date: 15 February, 2008

Document Version: 1.0

Revisions

Document Version	Date	Author (Partner)	Description of Changes
0.1	22 Jan. 08	Yosu Gorroñogoitia (Atos)	ToC
0.2	24 Jan. 09	Luis Quijada (Atos)	Section 4 – ToC modification
0.4	12 Feb. 08	Yosu Gorroñogoitia (Atos)	Several sections fulfilled.
0.5	18 Feb. 08	Luis Quijada, Yosu Gorroñogoitia (Atos)	Several sections fulfilled.
1.0	19 Feb 08	Luis Quijada, Yosu Gorroñogoitia (Atos)	Final editing.

INDEX

INDEX OF TABLES

INDEX OF FIGURES

1 Scope

This deliverable, but also D5.2 and D5.3, does not fully comply with the organization described in the Description of work, where we were thinking of slicing our tools according to the distinction among data, *ware, and process. While designing our tools, we understood that that division was unnatural, and would have led to conceivable overlapping among the different tools. Our tools are functionality-specific, but they can easily work on the artifacts at the different levels of the hierarchy identified by the MOMOCS Description of Work.

This is why we opted for a different partitioning based on supplied features, instead of the abstraction level. In other words, we prefer to get rid of the horizontal layers originally proposed, and identify consistent and self-contained vertical segments that easily spread across data, *ware, and process.

1.1 Purpose of the Document

The purpose of the document is offer a complete guide for the XSM KB Repository Tool, which is part of the MOMOCS Tools Suite. This guide helps the users of this tool to install it, to configure it and to start using it. Besides, it offers a detailed reference to all the features provided by the tool.

The development was driven by [2007b] "D41 – XIRUP Supporting Tools Specification".

1.2 Document Structure

This document is organized as follows:

Section 2 introduces the XSM KB Repository describing its main features in the context of XIRUP Modernisation methodology. Section 3 is a getting started guide that explains the requirements needed to install the tool, how to install the tool and how start playing with it. Section 4 contains the main tool user's guide. Section 5 is a troubleshooting section, listing knows issues, bugs reported and a FAQ. Section 6 contains the appendixes, describing additional tool ongoing features under development or forthcoming features to be implemented. Finally, section 7 contains the annexes with the acronyms, glossary and reference documents.

2 KB Repository Introduction

2.1 Introduction

XSM Knowledge Base Repository is a container for XIRUP artefacts produced during the different phases of modernization process that converts the legacy system (TBMS) into the modernized system (MS), to promote their reuse whenever is possible. KB Repository will provide support for storing and retrieving those artefacts, which may be required later on by some activities of that modernization process, which can be performed following the XIRUP methodology 2007c. These main features facilitate the re-use of those artefacts in the same modernization process or in others, as it is suggested by the XIRUP.

The repository stores the following artefacts produced during some phases of the XIRUP methodology:

- A XSM created by the Analyst using the XSM Editor.
- A metamodel transformation rules set created by the XIRUP Analyst using the XSM Transformation tool.
- A XSM model transformation historic including a set of transformation mappings.

The XSMs produced during the modernisation process we can store into the KB Repository follow this classification:

- By the system target: we can classify models according into the legacy system (TBMS) models or the modernized system (MS) models.

- By the level of abstractness: we can classify the models into CIM, PIM or PSM, that is, different views describing the same system at different levels of detail or concerns.
- By the scope or extension of the model: we can consider complete models by they own, describing a whole system, or partial models describing only restricted regions of the system. In the latter case, it is included (among other ones) the "component type models", describing abstract components, which may be instantiated by particular components in the model, and "pattern models", a sort of well-known model solution for a particular problem .

All those kind of models are built by the XSM Editor, which can send and retrieve them to/from the XSM KB Repository.

A transformation rules set is a list of mappings between the elements of a source metamodel and those of a target metamodel[1]. A particular metamodel is XIRUP, which is used to describe XSM models. This transformation rules set is used to convert from one source model (as instance of the source metamodel) into the target model (instance of the target metamodel). XSM transformation rules set is created and edited by the XSM Transformation tool.

Finally, the XSM model transformation mappings are a set of links between objects belonging to the source model and those equivalent ones belonging to the target model after being apply to them the transformation rules set. They allow navigating from the objects of the source model to the objects of the target model and vice versa. A set of transformation mappings linking different models coming from an initial one constitute a XSM model transformation historic, also store into the XSM KB

[1] In general source and target metamodel are not the same. However, in MOMOCS transformations are defined between XIRUP metamodel and itself.

Repository. A XSM model transformation mapping (together with the source model, target model and XSM transformation rules set) constitutes a chain link of that transformation historic.

The main operations supported by the XSM KB Repository are the following:

- Manage the repository
- Store artefacts
- Browse the repository
- Search for artefacts
- Retrieve artefacts

XSM KB Repository is organized according with a hierarchical classification determined by the user. This classification is basically a taxonomical classification consisting of semantically annotated (using metadata and/or ontological concepts) classifiers. In other words, a tree-based structure of semantic annotated folders where are stored the XIRUP artefacts. These folders are semantically annotated by the user, when they are created, assisted by the XSM KB Repository tool, which make use of a set of available domain ontologies. Domain ontologies for MOMOCS case studies are supplied by their domain experts and come with the XSM KB Repository tool bundle. Semantic annotation will be used by the repository to search for artefacts using semantic reasoning.

XSM KB Repository management mainly comprises the organization of the repository hierarchical classification above described.

XIRUP artefacts are stored within the repository and annotated in a similar way than for the repository management.

Repository browsing permits to navigate the repository structure to discover the artefacts stored within. Its main purpose is to manually discover and retrieve XIRUP artefacts.

In a same manner, repository Searching is aimed to discover services that match with a searching criteria determined by the user. XSM KB Repository will support semantic searching by creating semantic queries that are used to reason within the available domain ontologies. The discovered candidate artefacts are also ranked , using semantic algorithms that determine the best artefact matching according with the semantic query criteria and the semantic restrictions established by the domain ontologies, so only best scored candidates are shown to the user. XSM KB Repository will assist the user to determine the searching criteria on the basis of the available domain ontologies, and, according with those criteria, it will automatically build the semantic query.

3 Repository Getting Started

3.1 Requirements

This section describes the technical requirements required by KB Repository Tool

For installing and running the KB Repository Tool some minimum requirements need to be satisfied:

- Operating system: Windows 2000/XP (forthcoming versions will also support any OS compatible with JVM).
- JRE 1.6 (integrated versions of KB Repository will required JRE 1.5).
- eXist [eXist] database installed and configured (this release provides a Windows install facility which also installs this database, but forthcoming Eclipse update site will not provide this requirement).

3.2 KB Repository tool license

KB Repository is released under the Eclipse Public License.

See the detailed document at http://www.eclipse.org/legal/epl-v10.html.

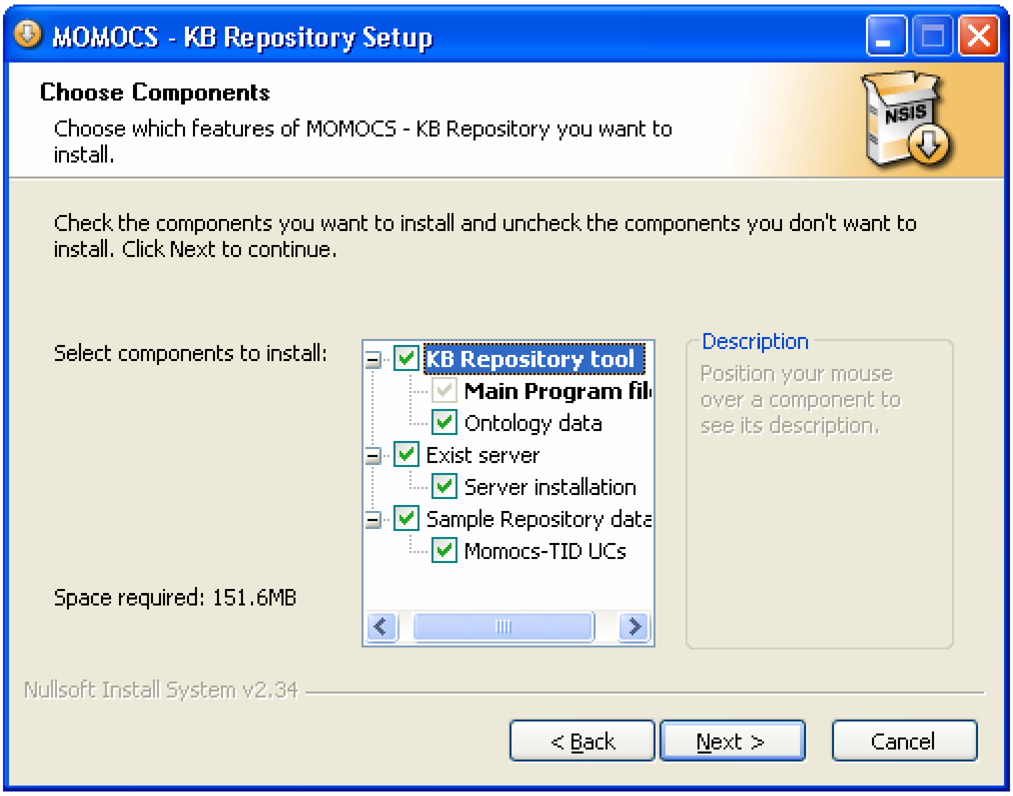
3.3 Installation Guide

KB Repository is released as Windows install facility that packs both KB Repository Client (Eclipse platform + KB Repository plugins) and KB Repository database server (eXist database).

3.3.1 Install options

To start the windows installer just double-click the executable file named: kbrepository_v1.0.0.exe. The installer assistant will be opened.

After the agreement of the KB Repository tool license (aforementioned) the next step consists of selecting which packages the user desires to be installed:



MOMOCS - KB Repository Setup

Choose Components

Choose which features of MOMOCS - KB Repository you want to install.

Check the components you want to install and uncheck the components you don't want to install. Click Next to continue.

Select components to install:

- ☑ KB Repository tool
  - ☑ Main Program fil
  - ☑ Ontology data
- ☑ Exist server
  - ☑ Server installation
- ☑ Sample Repository data
  - ☑ Momocs-TID UCs

Space required: 151.6MB

Description

Position your mouse over a component to see its description.

Nullsoft Install System v2.34

Figure 1 KB Repository installer. Choosing components

a) KB Repository tool main files (required) + ontology data (optional)

This option will install the minimum client requirements for running the KB Repository tool. That is an Eclipse 3.3.1 platform plus the KB Repository tool plugins. Please note that this minimum installation is mandatory.

Optionally the user can install de Ontology data files for additional Ontology annotation support.

Note: The Ontology files are accessible in the "OWL" folder within the installation root folder.

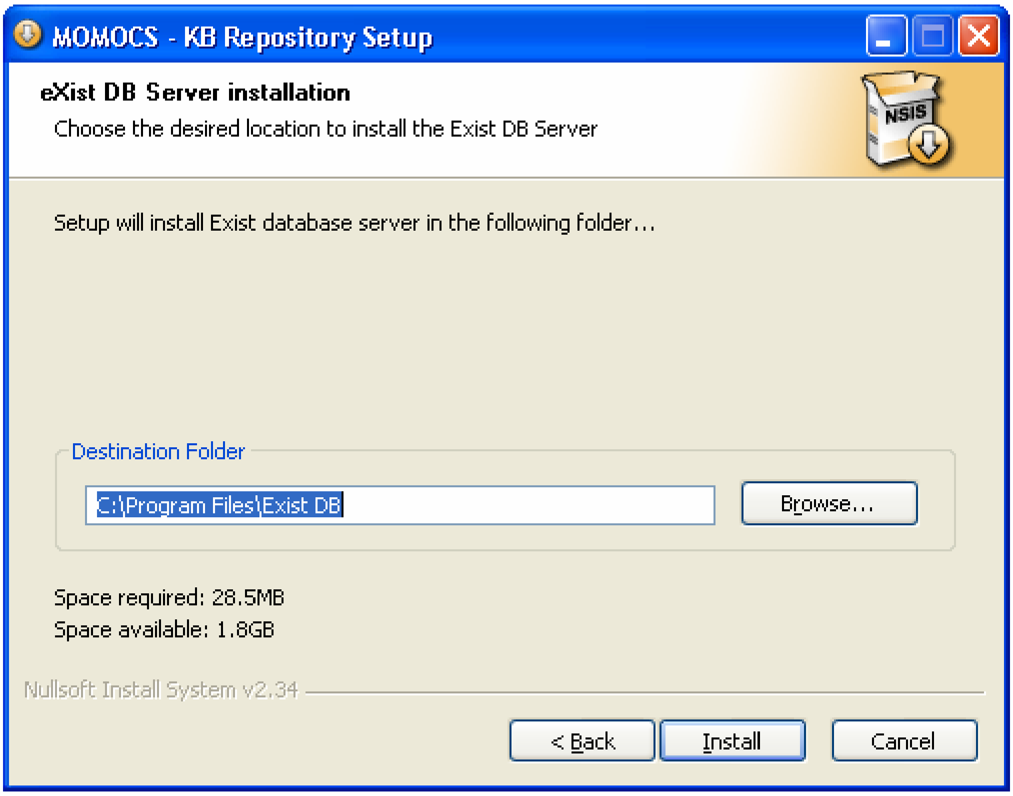
To install this section just provide to the installer the desired installation path and proceed normally.

b) eXist database server (optional)

The KB Repository tools needs an eXist database server to connect to. The installer provides an easy way to install and automatically configure the eXist server to be used within the KB Repository tool.

To proceed just tell the assistant where to install the eXist server and follow carefully further instructions.

eXist installation takes few minutes to be installed.



eXist DB Server installation

Choose the desired location to install the Exist DB Server

Setup will install Exist database server in the following folder...

Destination Folder: C:\Program Files\Exist DB

Space required: 28.5MB

Space available: 1.8GB

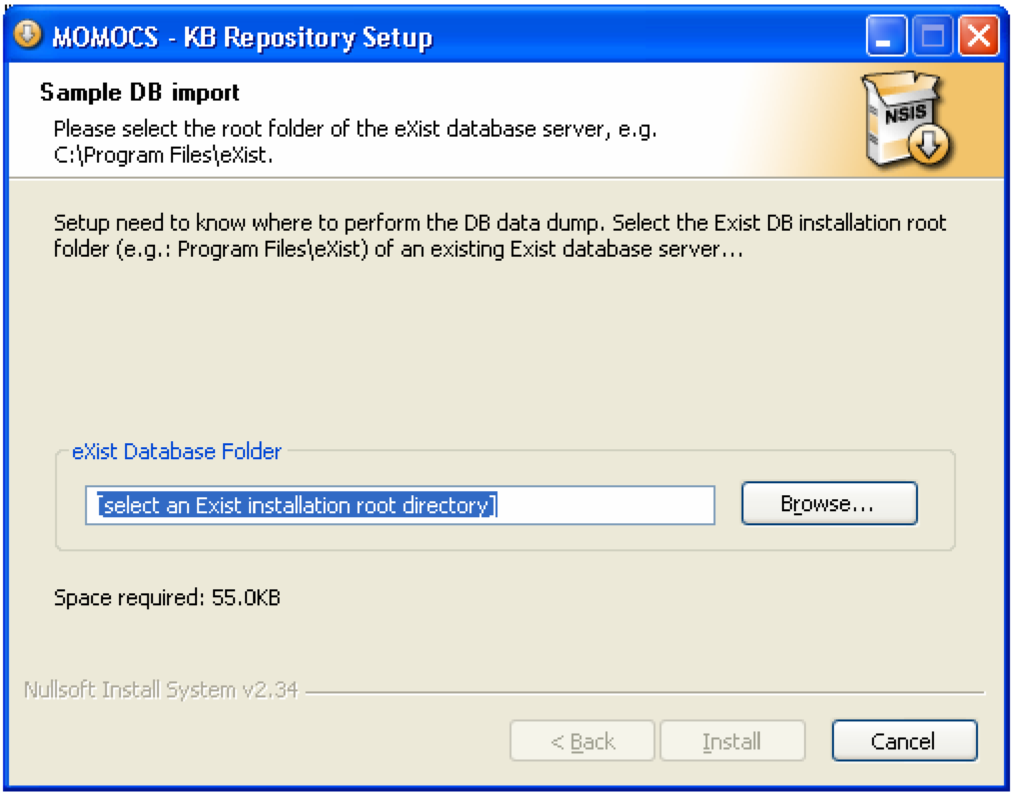
Figure 2 KB Repository installer. eXist DB Server installation

Important note: by default, the eXist database server is automatically configured to listen at port 8095.

c) Sample repository data (optional)

Once installed the full eXist server package, is also recommended to import some sample data.

To install this package select the root folder of an existing eXist installation by pressing the button "Browse" at the eXist DB Server installation section (see figure below). When installed via this installer, the default path of the eXist server is: C:\Program Files\Exist DB.



MOMOCS - KB Repository Setup

Sample DB import

Please select the root folder of the eXist database server, e.g.
C:\Program Files\eXist.

Setup need to know where to perform the DB data dump. Select the Exist DB installation root folder (e.g.: Program Files\eXist) of an existing Exist database server...

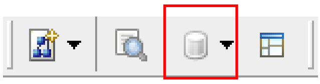
eXist Database Folder: [select an Exist installation root directory]

Space required: 55.0KB

Nullsoft Install System v2.34

Figure 3 KB Repository installer. Sample DB import

Follow the instructions given by the installer to successfully install the sample source data into the eXist server.

Note: At this step, the installer will try to start the database, import the sample data and automatically shutdown the server. Remember to, once the KB Repository tool is running, start the database server by using the Repository database management button: . See section 4.3.1 for more details.

3.4 Getting Started

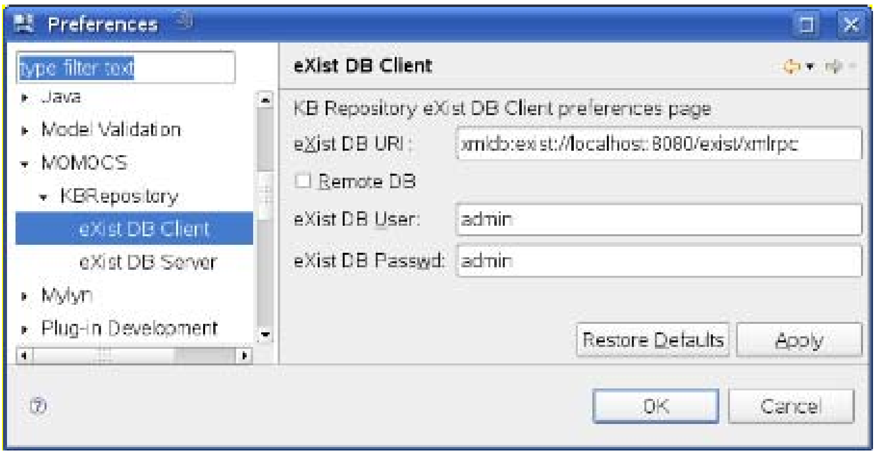
This section explains the initial steps to use KB Repository Tool.

To launch KB Repository Tool go to Start menu >> MOMOCS KB Repository tool >> KB Repository menu option or double click on the kbrepositorytool.exe icon on the installation directory (if using Windows explorer).

Important note: it is hardly recommended to select the menu option "KB Repository tool (clean)" at the KB Repository Start menu icon group, the first time the KB Repository is started.

3.4.1 KB Repository configuration

KB Repository Tool is preconfigured by the installation facility, so not additional configuration is required. However, the user can re-configure KB Repository whenever he/she wishes by accessing the Eclipse configuration. Choose Window>Preferences> to open the Preferences dialog. Then choose MOMOCS > KB Repository on the left tree view. Currently, it can be edited the eXist database configuration, both for the server and the client.



eXist DB Client

KB Repository eXist DB Client preferences page

eXist DB URI: xmldb:exist://localhost:8080/exist/xmlrpc

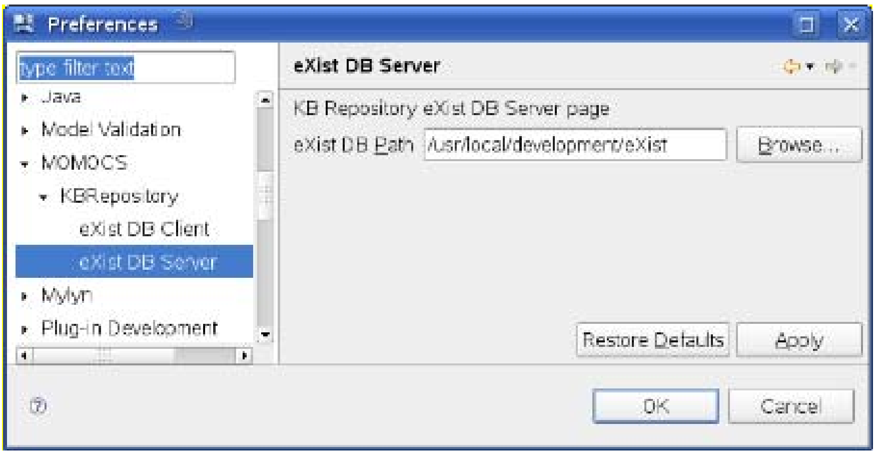
☐ Remote DB

eXist DB User: admin

eXist DB Passwd: admin

Figure 4 KB Repository preference page: eXist DB client

eXist DB Client configuration establishes the information required to connect with the KB Repository database: database URI, user and password. In case that KB Repository is accessing a remote database, the checkbox Remote DB has to be selected. Thus, database management support in KB Repository tool is disabled.



eXist DB Server

KB Repository eXist DB Server page

eXist DB Path /usr/local/development/eXist

Figure 5 KB Repository preference page: eXist DB Server

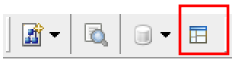
eXist DB Server configuration establishes the information required to access the local eXist database installation. This allows the KB Repository to manage the startup and the shutdown of the database (only if Remote DB checkbox is unmarked). See section 4.3.1 for repository database management.

3.4.2 KB Repository workbench

Within Eclipse workbench, KB Repository tool is accessible by opening the KB Repository perspective.

To open the KB Repository perspective:

Navigate through KB Repository tool menu >> Views >> Open KB Repository tool perspective.

Or press button  at the KB toolbar button group.

Or choose from Eclipse menu: Window > Open perspective > Other, select "KB Repository tool" option.

Once the application engines has been started the perspective is brought to the front opening as shown in the figure below:

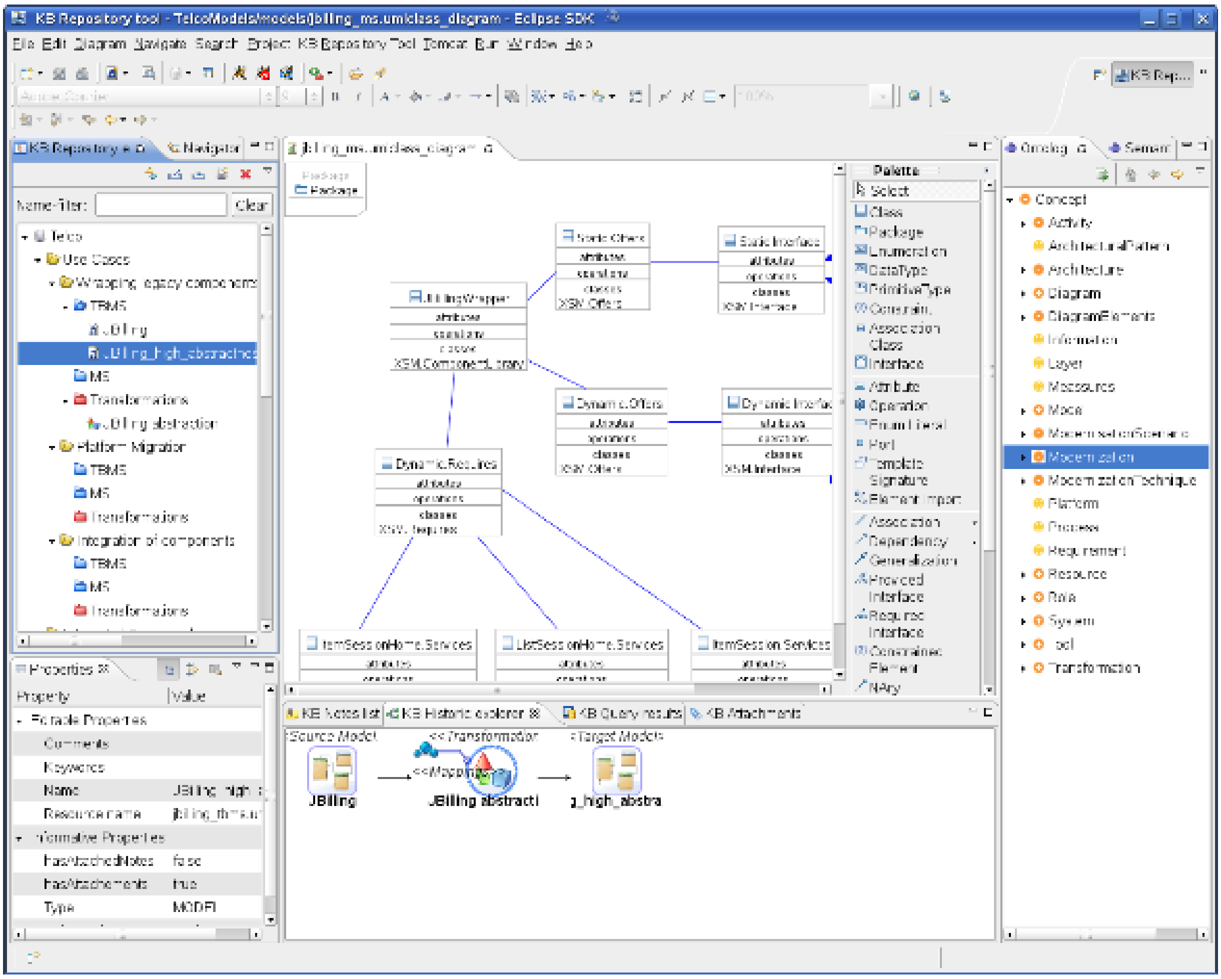


Figure 6 KB Repository workbench

4 KB Repository User's Guide

This section describes in detail all the features provided by KB Repository Tool. Besides, it describes the most relevant uses of KB Repository Tool.

4.1 KB Repository Workbench Perspective

The KB Repository offers its functionality via the "KB Repository tool" perspective.

Each workbench perspective is not more than a customized arrangement of windows and buttons in a particular way.

Those windows, formerly named "Views", are consequently distributed for providing the user a nice and "easy to use" interface to interact effectively with the KB Repository.

To open the KB Repository tool perspective see section 3.4.2: Once the application engines has been started the perspective is brought to the front opening as shown in the figure below:

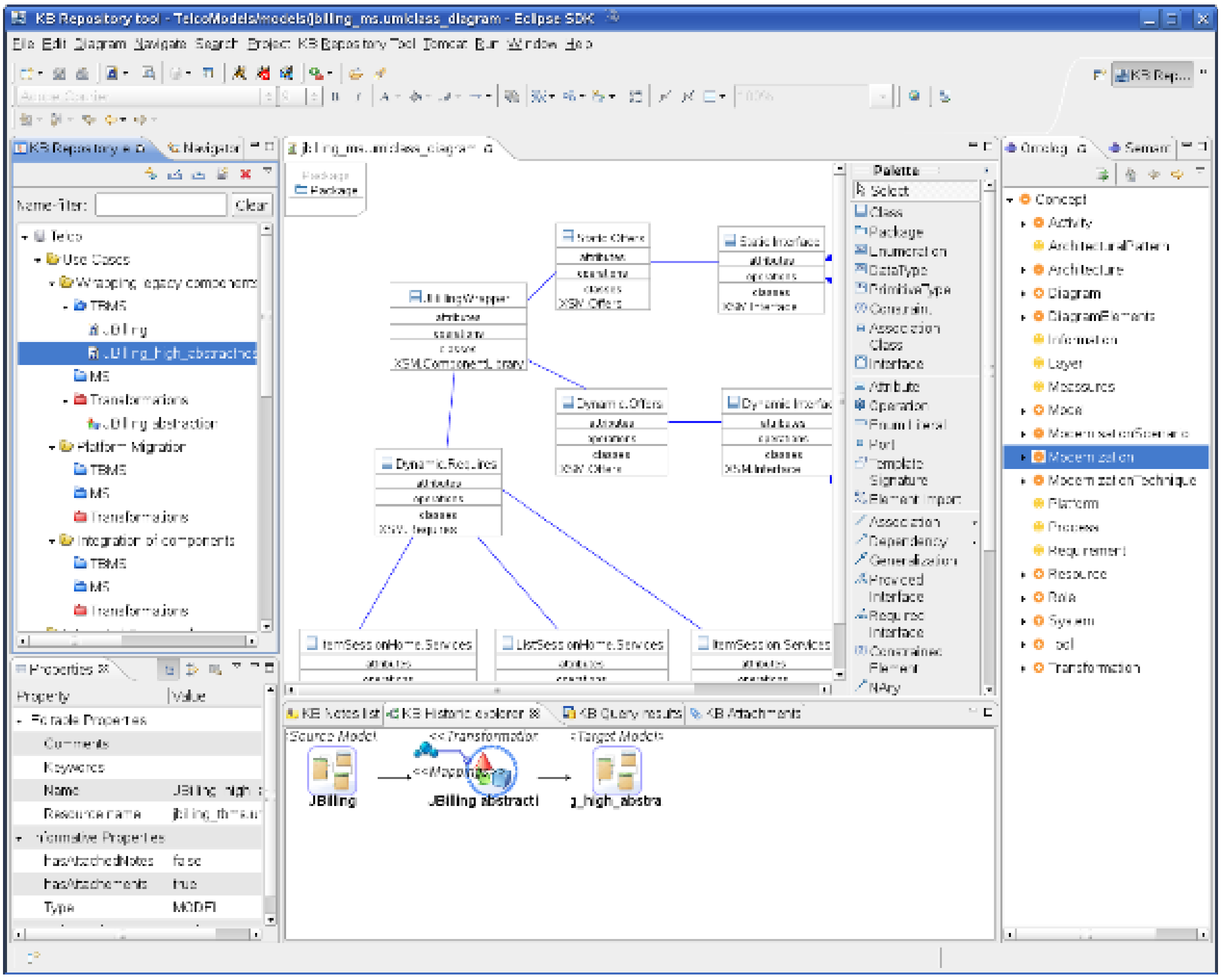


Figure 7 KB Repository perspective

As a normal perspective within the Eclipse platform, the user is able to close the entire perspective, even to reset it (helpful mechanism when some views are hidden accidentally), and hide / show some of the "Views" that compose it.

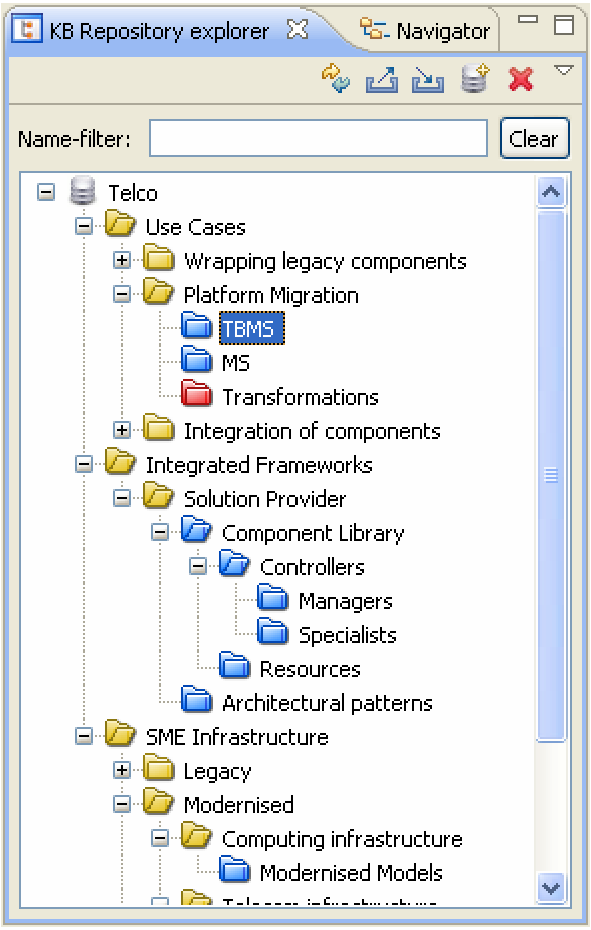
To close the KB Repository tool perspective: Right-click the KB Repository perspective button at perspective toolbar KB KB Repository… (top-right workbench-window corner) and select the close option.

To reset the KB Repository tool perspective to default configuration: Right-click the KB Repository perspective button at perspective toolbar KB and select the reset option.

The KB Repository tool perspective is made by a set of views, some of them are key views within the tool (managing views), and the others could be considered as auxiliary (informative views).

Management views:

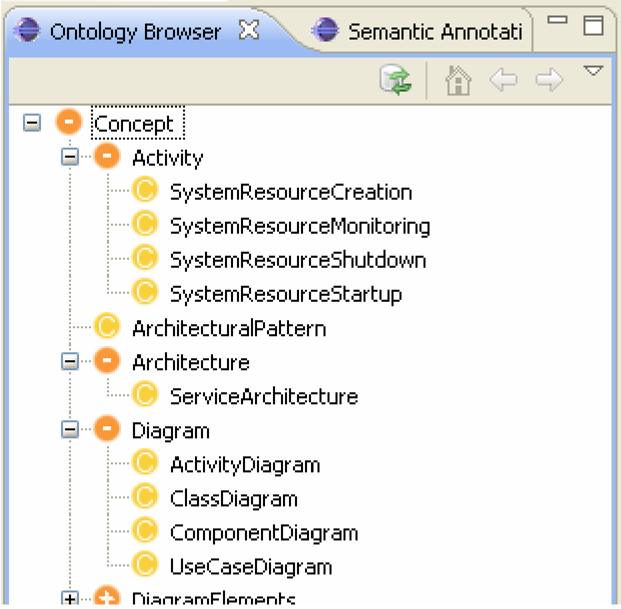
KB Explorer view (for detailed section see 4.1.1)



- Telco
  - Use Cases
    - Wrapping legacy components
    - Platform Migration
      - TBMS
      - MS
      - Transformations
    - Integration of components
  - Integrated Frameworks
    - Solution Provider
      - Component Library
        - Controllers
          - Managers
          - Specialists
        - Resources
      - Architectural patterns
  - SME Infrastructure
    - Legacy
    - Modernised
      - Computing infrastructure
        - Modernised Models

Figure 8 KB Repository explorer view

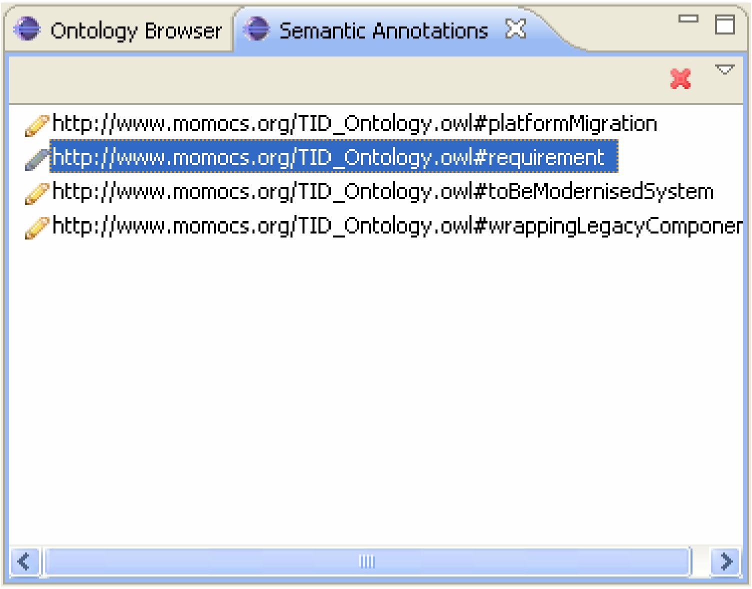
KB Repository Ontology Browser (for detailed section see 4.1.3)



Concept
- Activity
  - SystemResourceCreation
  - SystemResourceMonitoring
  - SystemResourceShutdown
  - SystemResourceStartup
- ArchitecturalPattern
- Architecture
  - ServiceArchitecture
- Diagram
  - ActivityDiagram
  - ClassDiagram
  - ComponentDiagram
  - UseCaseDiagram
- DiagramElements

Figure 9 KB Repository ontology browser

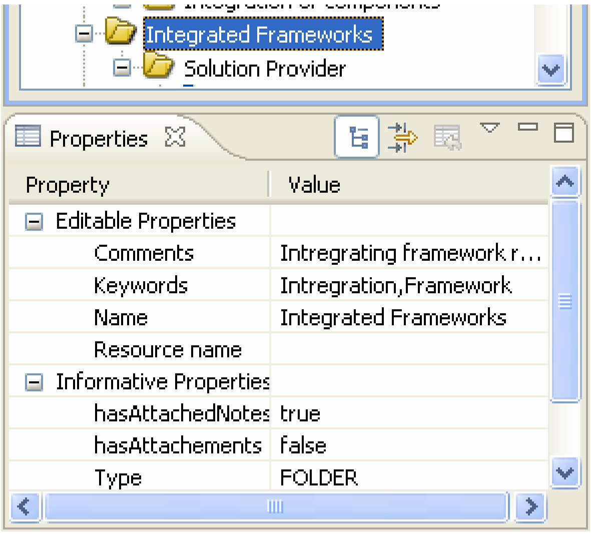
KB Repository semantic annotation browser (for detailed section see 4.1.4)



http://www.momocs.org/TID_Ontology.owl#platformMigration
http://www.momocs.org/TID_Ontology.owl#requirement
http://www.momocs.org/TID_Ontology.owl#toBeModernisedSystem
http://www.momocs.org/TID_Ontology.owl#wrappingLegacyComponer

Figure 10 KB Repository annotation browser

Properties editor (for detailed section see 4.1.2)



Property	Value
⊟ Editable Properties	
Comments	Intregrating framework r…
Keywords	Intregration,Framework
Name	Integrated Frameworks
Resource name	
⊟ Informative Properties	
hasAttachedNotes	true
hasAttachements	false
Type	FOLDER

Figure 11 Eclipse properties view

Informative views:

Historic (for detailed at section see 4.1.5.1)

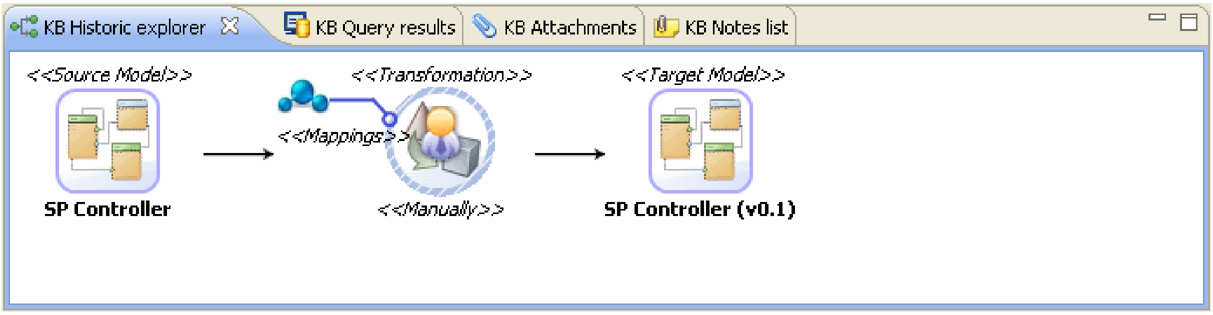
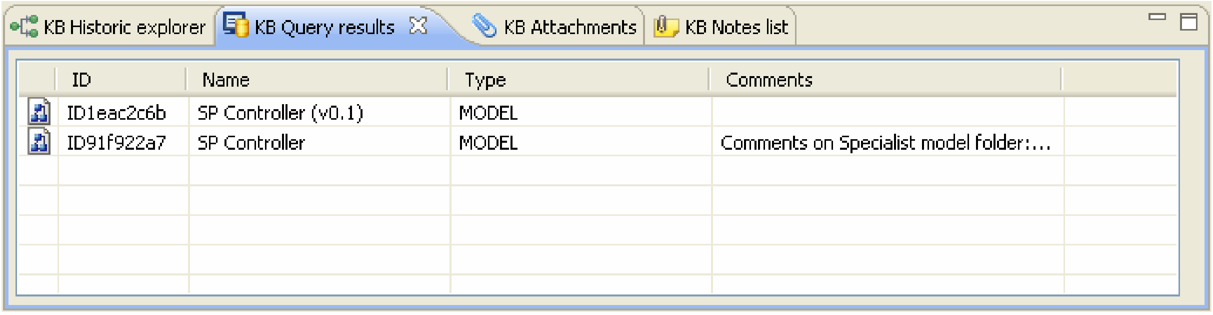


Figure 12 KB Historic explorer

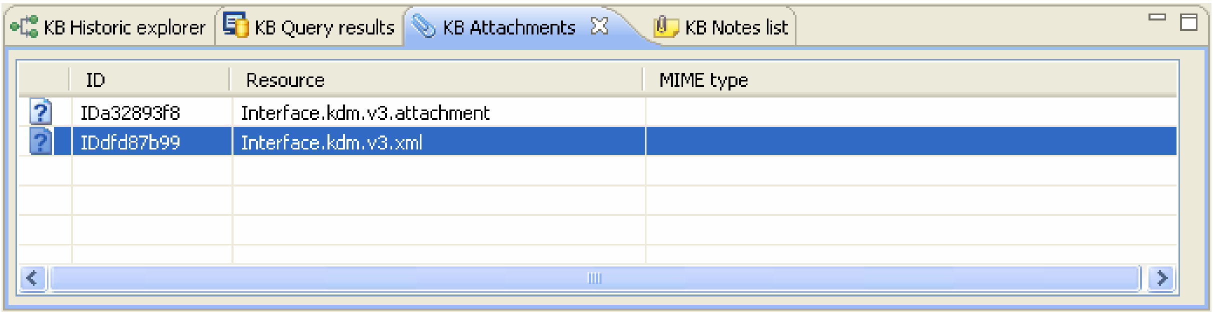
Query results (for detailed at section see 4.1.5.2)



ID	Name	Type	Comments
ID1eac2c6b	SP Controller (v0.1)	MODEL	
ID91f922a7	SP Controller	MODEL	Comments on Specialist model folder:...

Figure 13 KB Query results

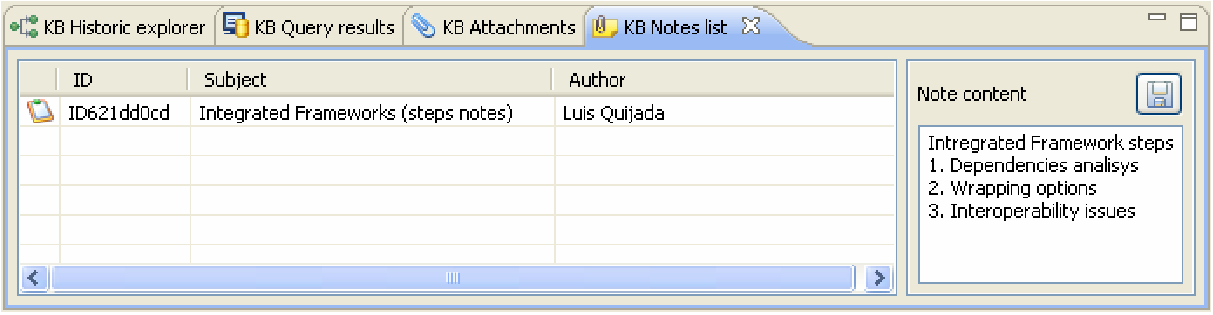
Attachments (for detailed at section see 4.1.5.3)



ID	Resource	MIME type
IDa32893f8	Interface.kdm.v3.attachment	
IDdfd87b99	Interface.kdm.v3.xml	

Figure 14 KB Attachments view

Notes (for a detailed at section see 4.1.5.4)



ID	Subject	Author
ID621dd0cd	Integrated Frameworks (steps notes)	Luis Quijada

Note content

Intregrated Framework steps
1. Dependencies analisys
2. Wrapping options
3. Interoperability issues

Figure 15 KB Notes list view

The tool perspective is divided into 3 different main areas: left, right, bottom where the KB Repository views are docked to. This initial configuration is the default one allowing the user to freely change the perspective arrangement.

KB Repository tool has also a dedicated menu and a toolbar button group located at the upper area of the workbench window.

By default the KB Explorer view (main KB Repository management view) appears at the left side with the common Properties view located at the bottom side in the same panel. KB Ontology Browser and Annotation editor are docked together composing the right panel. The rest of the views (informative views) are located at bottom area of the workbench window.

Figure 7 shows the default perspective configuration.

KB Repository perspective can be customised just by dragging and dropping views to another location, resizing them, closing them, open other available views.

Once customized the perspective, the arrangement configuration will be stored when closed and restored the next time the perspective is opened.

4.1.1 KB Repository Explorer View

As written before, KB Explorer is the most important view of the tool allowing us a complete management of the KB Repository structure by creating and removing KB Repository domain objects (see 4.2.1) and modifying the artefact hierarchy.

KB Repository Explorer layout description

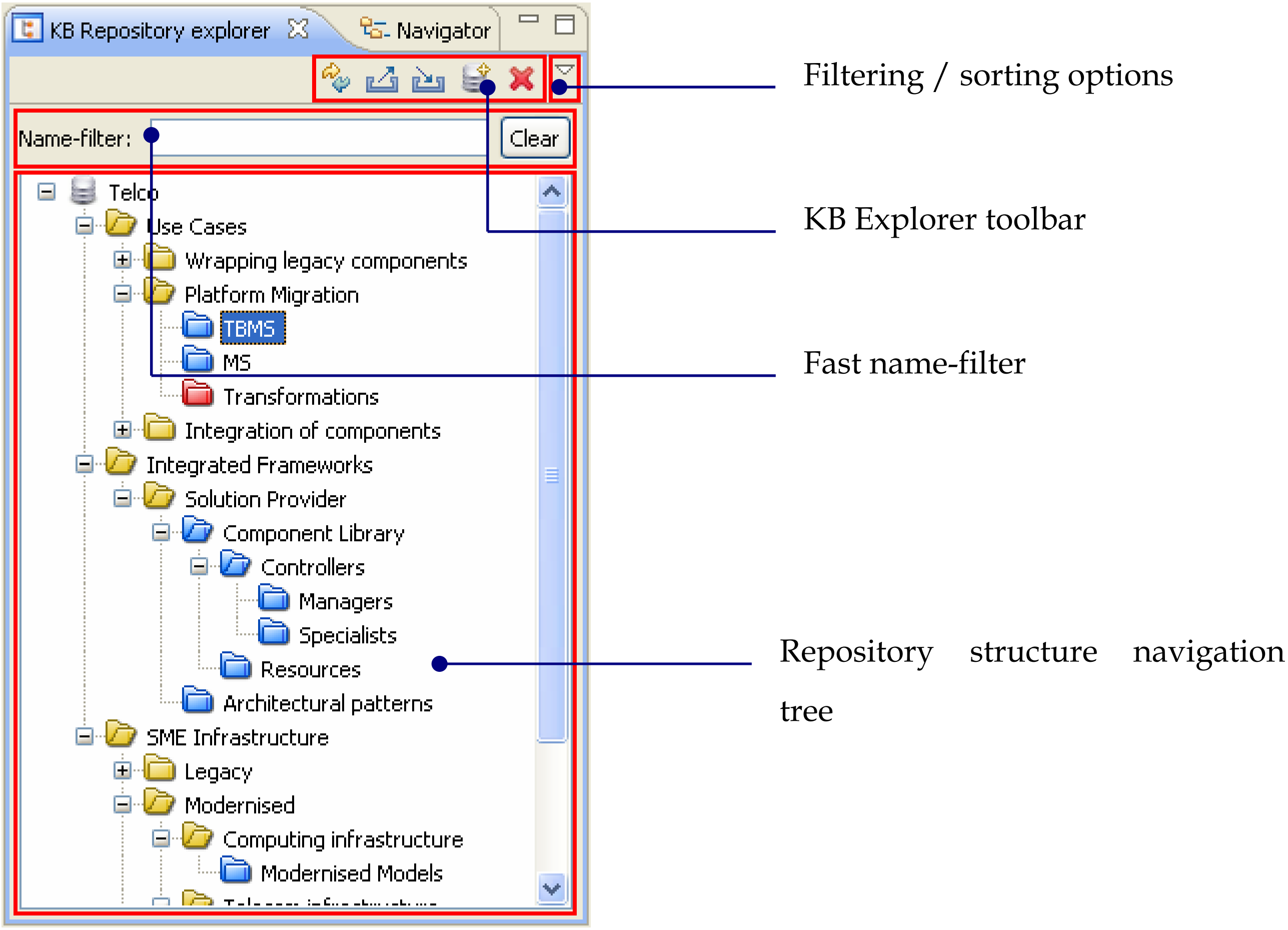


Figure 16 KB Repository explorer view

4.1.1.1 KB Repository navigation

The repository navigation tree, the core of the KB Explorer view, is a kind of file system virtualised browser. Due to the potential huge amount of KB artefacts to be stored in the repository, the KB Explorer navigation view offers the user filter capabilities, in order to reduce the number of displayed artefacts and sorting options.

Selecting an artefact in the navigations tree produces the informative views to be refreshed showing information related to the selected artefact, like artefact properties, transformation historic (in case of models), possible attachments, notes, and so on.

As a file system navigation tree, there are folders and non-container elements, like model, transformations, notes and so on.

Hide / show the container contents

In the navigation tree, each container has a small handle (⊞/ ⊟). Clicking on the handle when it looks like ⊞ causes folder contents to be made visible. Clicking on the handle when it is like ⊟ causes all sub-folders to be hidden.

Drag and drop

The Navigation tree allow drag'n'drop of elements for easy modification of the repository structure. There are some restrictions due to the domain model and error message could appear in case of not allowed operations, e.g. when trying to drag a repository-type element.

To move an element into another folder and perform such kinds of operations just left-click on the desired element to move it, holding the mouse left button clicked, drag the element into the desired location and release the mouse left button to drop it on.

4.1.1.2 Ordering and filtering

In a complex repository structure filtering and ordering is a key functionality for fast location of elements. The filtering does not modify at all the internal repository structure, just reduces the displayed elements using some criteria.

Built-in filters

KB Repository allows predefined filtering options like: show only models, show only transformations, hide/show notes elements (take into account that notes elements are hidden by default).

To access the predefined filters see Figure 17:

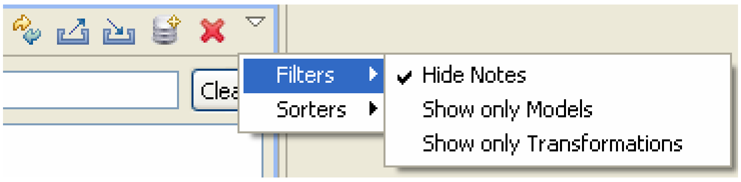


Figure 17 KB Repository explorer filters

Dynamic filtering

Dynamic filtering is performed through the "Name filter" command at the top of KB Explorer view below the main view toolbar . It allows the user to filter the displayed elements producing a reduced tree structure by inspecting each one of the model elements and matching its **name** property with the expression given at the "Name-filter" text box.

Wildcards and regular expressions

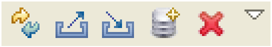
At dynamic filtering, two kind of standard wildcard are allowed: the asterisk (*) and the question mark (?). As world-wide known wildcard standards, the asterisk (*) would mach any character and the question mark (?) would match 1 or 0 characters.

More in general, the dynamic filtering capabilities evaluate the contents of the text box as a regular expression, allowing the user to build powerful and complex search patterns.

E.g.: user wants to locate some models elements within the repository structure which name begins with the expression "DB". Then the user would write at the name-filter text box: "DB*". The KB Repository explorer would refresh its data model in order to show all the elements (and their respective parents) which name starts with "DB".

4.1.1.3 KB Repository explorer view toolbar

Used to perform some specific explorer view related actions, it is located at the upper side of the KB Explorer view, appearing as shown below:



Refresh button: forces de KB Explorer to refresh its contents, re-building the repository tree structure

Export-tool: opens the Export-tool wizard (see detailed description at section 4.3.2)

Import-tool: opens the Import-tool wizard (see detailed description at section 4.3.2)

New repository-type element: opens the New Repository wizard (see detailed description at section 4.2.2).

✖ Delete selected artefact/s (see detailed description at section 4.2.6).

▽ Filtering / ordering sub-menus

4.1.1.4 KB Repository navigation tree contextual menu

To rapid access the allowed operations that could be applied to a KB Repository element, just select the desired element/s and right click.

The contextual menu appears showing the main actions related to the selected artefact.

See an example below:

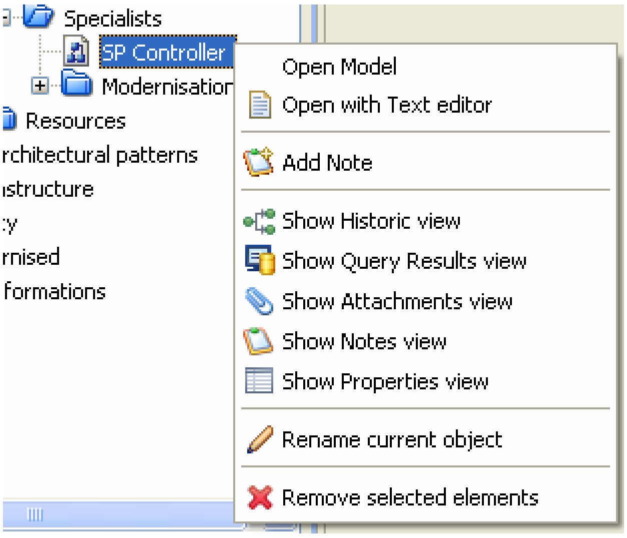
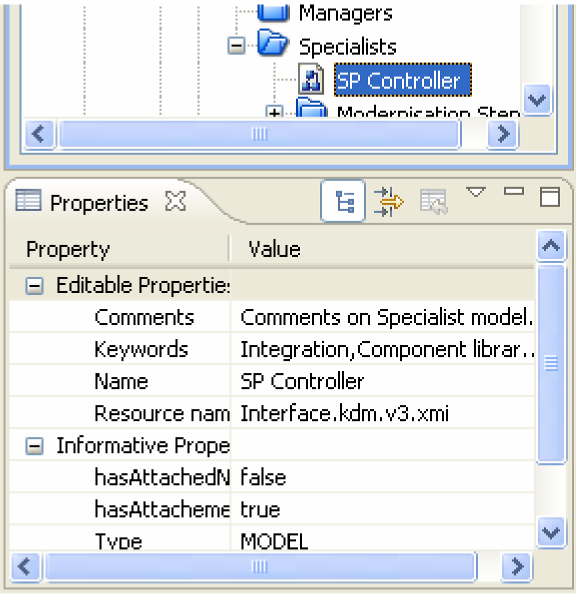


Figure 18 KB Repository explorer contextual menu

View the detailed command descriptions at the administration guide (see detailed description at section 4.2).

4.1.2 KB Repository artefacts properties editor

The Properties view is a common view in the form of a table widely used along most of the data manipulation applications. It is shown in the picture below.



Property	Value
⊟ Editable Propertie:	
Comments	Comments on Specialist model.
Keywords	Integration,Component librar..
Name	SP Controller
Resource nam	Interface.kdm.v3.xmi
⊟ Informative Prope	
hasAttachedN	false
hasAttacheme	true
Type	MODEL

Figure 19 Eclipse properties view

This view (as the informative views are) is selection context sensible. Selecting a KB Repository artefact using the KB Explorer or any of the informative views causes de properties table to be refreshed displaying the predefined artefact properties and its current values.

There are two kinds of properties, or categories:

- Informative: read-only properties initialized during artefact creation (e.g. “Unique Id.”) or indicating some states of the object (e.g. “has Attachments”).

- Editable: those of them which can be modified at any time, like “Name” or “Keywords”.

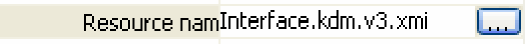
To edit an editable property:

1. Select the KB Repository artefact

2. Click on the actual value of the desired property

If the property is a string-value property, just write the new value and press the <<INTRO>> key or remove the focus off the field.

If it is a file system resource property, then click on the button that appears at the end of the property value field and a special resource selection dialog opens.

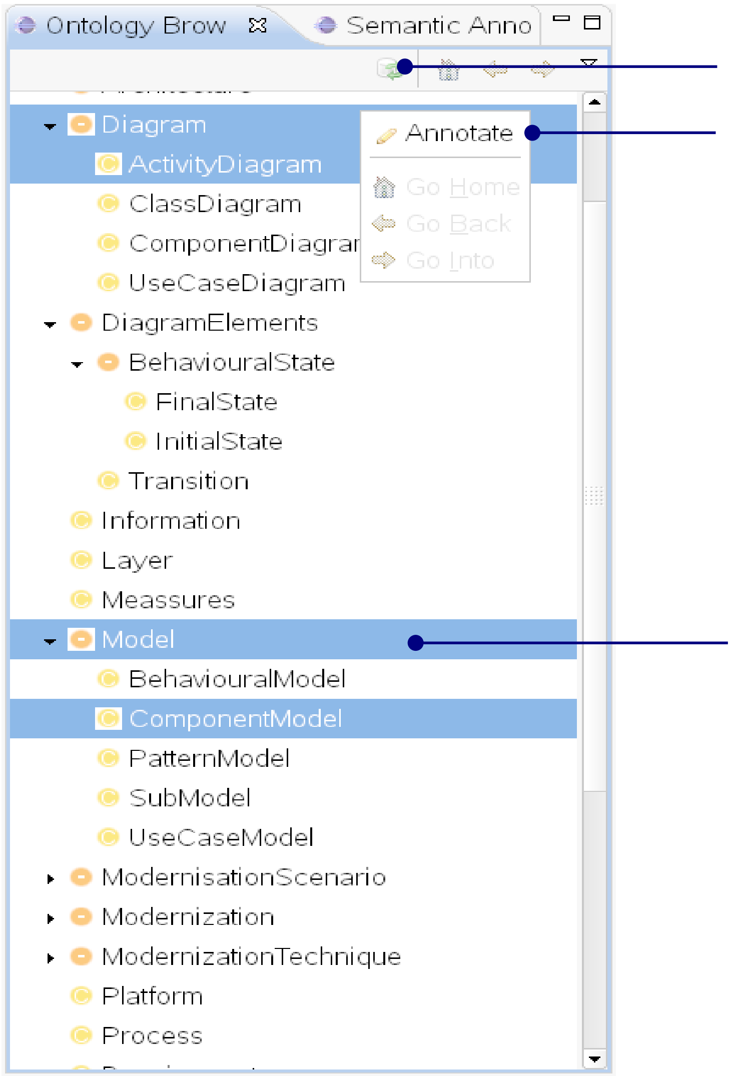

Resource nam Interface.kdm.v3.xmi

Switch to normal / categorized view:

The properties can be displayed divided into categories or ordered alphabetically. To switch between these different modes click the button located at the properties view toolbar.

4.1.3 KB Repository Ontology Browser

Ontology Browser view is a graphical browser that shows a selected ontology to be used to annotated the KB Repository artefacts selected in the KB Explorer view with semantic concepts. The reason to annotate repository artefacts with semantic metadata is to support intelligent search (see section 4.4.2). Those KB Repository artefacts can be repositories, folders, models, transformations.



Change ontology

Contextual menu for annotation

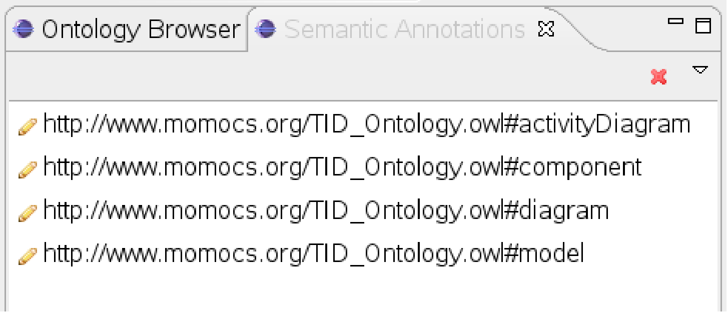
Multiple selected ontology concepts for annotation

Figure 20 KB Ontology browser view

The user can navigate the ontology, browsing concepts and selected those that better describe the artefact selected within the repository in KB Explorer View. Then, right clicking a contextual menu appears with an option to annotate the artefact with the semantic concepts selected. The annotations of each repository artefact are shown in the semantic annotation view (section 4.1.4). The ontology shown in this view can be changed by clicking in this view toolbar button [2].

4.1.4 KB Repository Semantic Annotation View

Semantic annotation view shows the semantic annotations attached to the repository artefacts. The annotations shown in this view are refreshed every time another artefact is selected within the KB Explorer view or new annotations are attached by the user for that artefact. An annotation entry is a qualified ontology concept, that is, a ontology namespace prefix plus the concept name: <ontology Namespace>#<concept name>



http://www.momocs.org/TID_Ontology.owl#activityDiagram
http://www.momocs.org/TID_Ontology.owl#component
http://www.momocs.org/TID_Ontology.owl#diagram
http://www.momocs.org/TID_Ontology.owl#model

Figure 21 KB Semantic annotation view

4.1.5 Informative / auxiliary views

KB Repository perspective owns a set of auxiliary views reflecting additional information depending on the current selected artefact.

[2] This feature is not available in current version, but it will be implemented in next ones.

4.1.5.1 Historic view

This the most important view for the XIRUP modernisation process and model transformation context. The “Historic view” shows graphically the model evolution relationships, transformation paths and branches, offering an easy way to visualize the particular evolution steps of the selected model artefact.

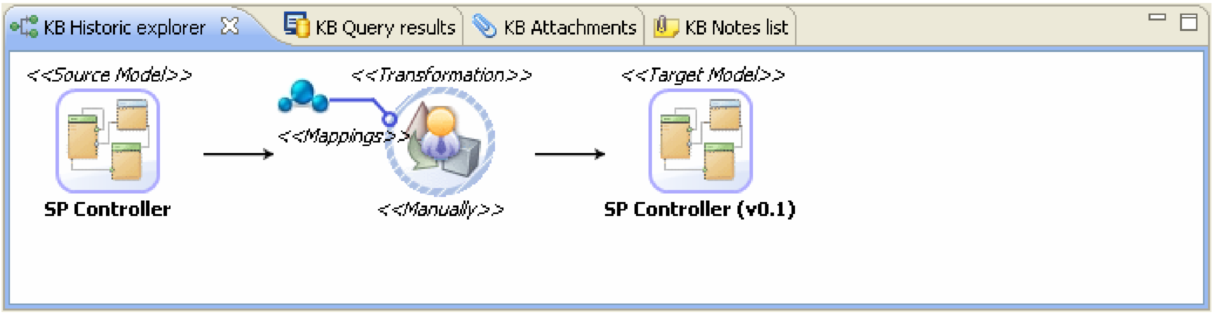


Figure 22 KB Historic view

To open a model or transformation artefact

Right click over the model icon and click the Open option at the contextual menu.

To open the transformation artefact associated to the evolution process, right click on its image icon and select the “Open” option.

The user must take into account that model transformations may occur without using a particular transformation artefact, that is: a manual transformation. In this case, the icon image of the transformation is decorated with different motives.

A special greyed effect applies to the artefacts image icon when they are deleted from the KB Repository tool. In other words, the model transformation information persists in spite of the state (deleted or not) of the rest of the components conforming the evolution process.

Navigating through the model evolution process

In a transformation process from a To Be Modernised System (TBMS) to a Modernised System (MS) a vast number of models may result. The system modernisation may make use of a large number of iterations to finally obtain a successfully modernised models package.

Tracking each model evolution is a key functionality in order to keep track of the changes that took place during each step of the process.

Using the graphical capabilities provided by this view it is easy for the Analyst to locate the model transformation history path. To illustrate this useful functionality let's imagine that we have a To Be Modernised model named "TBMS (1)", after applying some kind of transformation results a new model (as an evolution from the previous one) named "TBMS (2)".

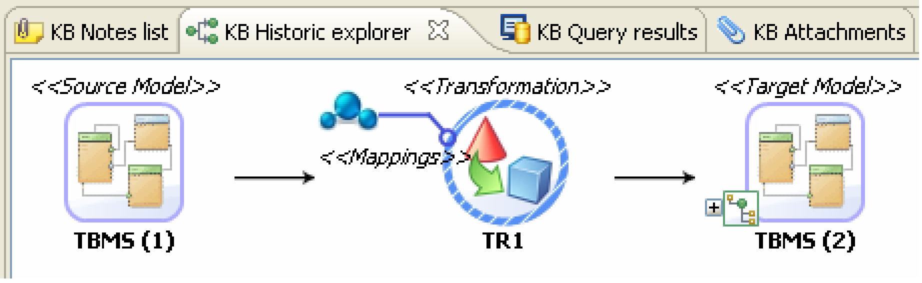


Figure 23 KB Historic view (II)

After inspecting its features, model "TBMS (2)" is considered a well-formed model and approved to be further used along the modernisation process. After that the XIRUP Analyst proceeds with the transformation of "TBMS (2)" model into a new "MS (1)" XSM modernised model.

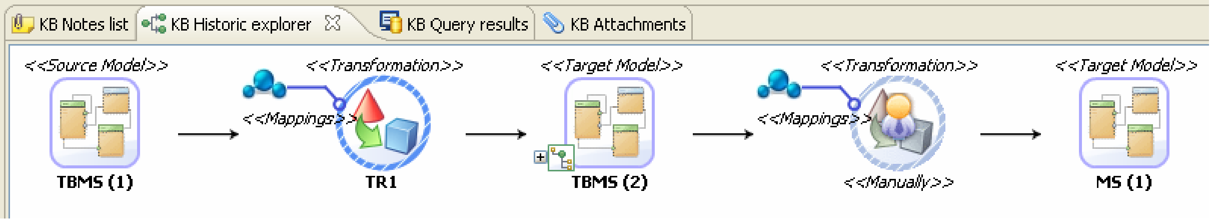


Figure 24 KB Historic View (III)

After performing some testing over the new model it is determined that is not a valid modernised one due to some constraint violations. The Analyst, using the KB Historic view, is able to easy locate graphically the "TBMS (2)" and restart, at the desired point, the modernisation process again by creating a new branch.

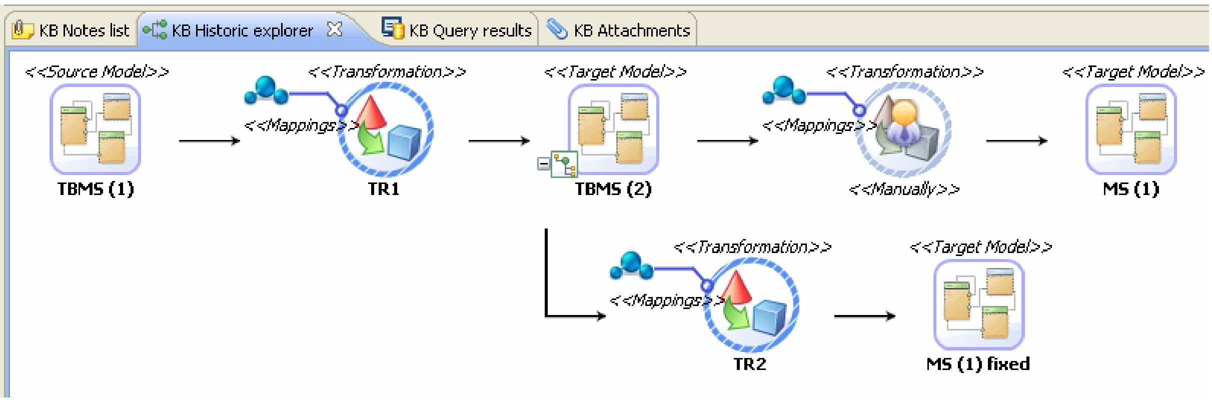
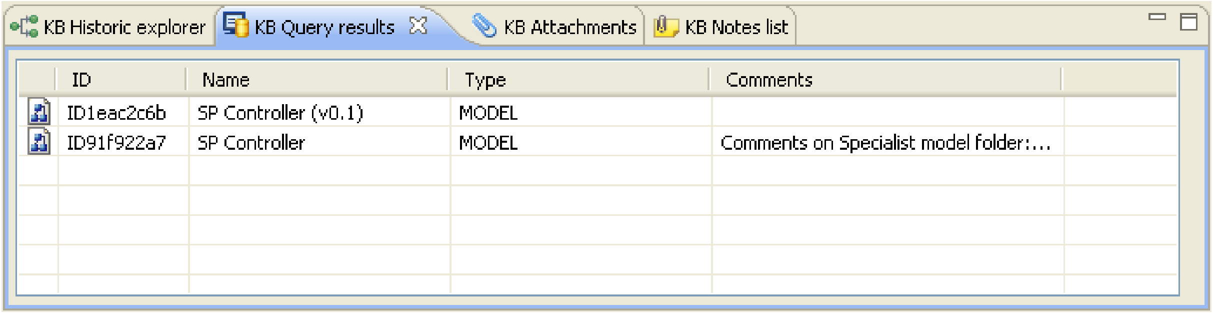


Figure 25 KB Historic View (IV)

4.1.5.2 Query results view

As a result of a searching action via the KB Repository search engine the Query result view shows the returned items satisfying the desired criteria ordered by matching score.

The results appear in the form of table where the most common properties of the KB artefacts are displayed. See the figure below.

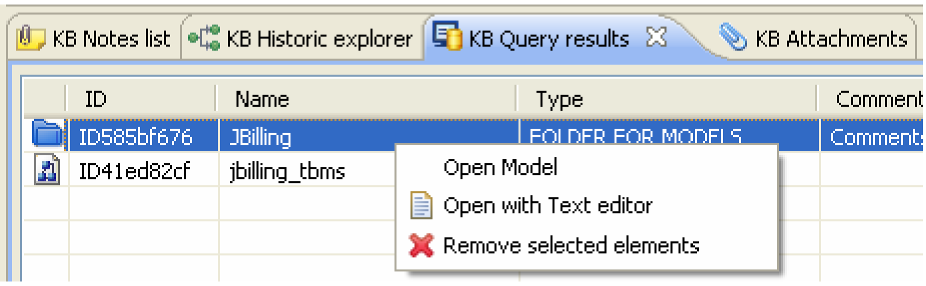


ID	Name	Type	Comments
ID1eac2c6b	SP Controller (v0.1)	MODEL	
ID91f922a7	SP Controller	MODEL	Comments on Specialist model folder:...

Figure 26 KB Query results

Selecting an element from the list causes the workbench to focus the selected object and reflect its properties at the Properties view.

Right-clicking on one of the items at the results list, it is displayed the contextual menu giving the user access to the common actions associated to each KB artefact.



ID	Name	Type	Comment
ID585bf676	JBilling	FOLDER FOR MODELS	Comment:
ID41ed82cf	jbilling_tbms		

Figure 27 KB Query results view contextual menu

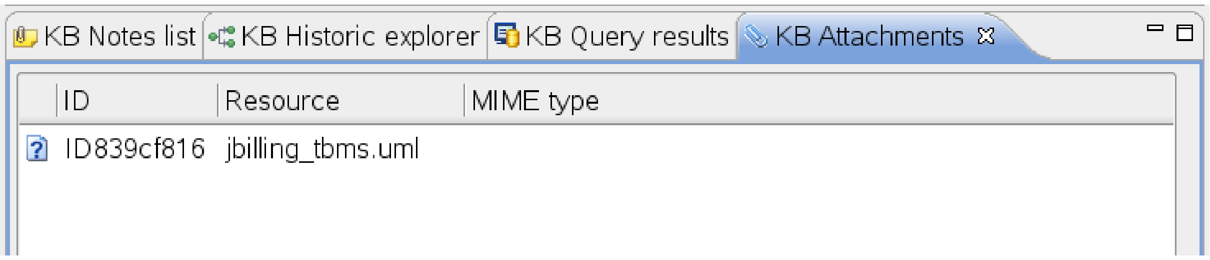
To order the result list:

The results can be ordered by Name or by Type by clicking the respective column header.

By default, the result are presented to the user ordered by matching score, the most accurate result is displayed first.

4.1.5.3 KB Attachments view

Attachments are automatically added to models or transformations when they are added to the repository tree view. These attachments are those additional files found within the same artefact source file directory sharing the same name but a different extension. For instance, this is a requirement of some UML editors which require one file for the model and another for the layout, therefore the repository should add both. Future releases of KB Repository tool will also allow attaching any file to those models or transformations added into the repository, which may complement them. Attachments for a particular repository artefact are show within the KB Attachments view.

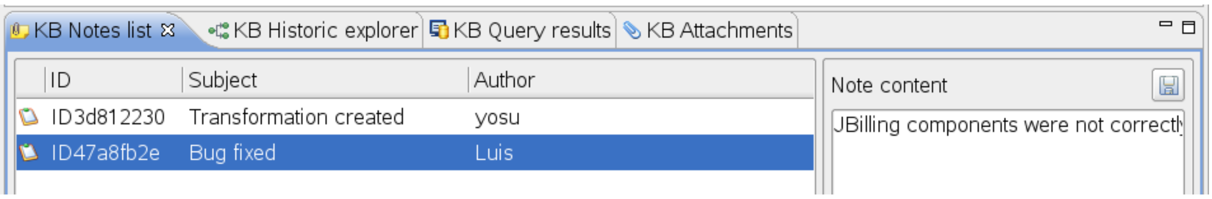


ID	Resource	MIME type
ID839cf816	jbilling_tbms.uml	

Figure 28 KB Notes list view

4.1.5.4 KB Notes view

Notes permit to annotate any repository artefact with additional information described using natural language, however this metadata are not used for searching purposes, but to complement the description of each artefact or to provide additional information. Notes can also been used to exchange messages between users. KB Notes list view in the auxiliary views panel on the bottom left lists all the notes attached to a particular repository artefact (Figure 29). By clicking on a note, the right panel show its content.



ID	Subject	Author
ID3d812230	Transformation created	yosu
ID47a8fb2e	Bug fixed	Luis

JBilling components were not correctl

Figure 29 KB Notes list content viewer

4.1.6 KB Repository Menu and button toolbar

Most of the KB Repository features are accessible from the KB Repository views main and contextual menus. Besides, most important features are also accessible through the KB Repository button tool bar (Figure 34) and main menu (Figure 30).

KB Repository Main menu:

The KB Repository main menu is divided into functionally differentiated submenu groups.

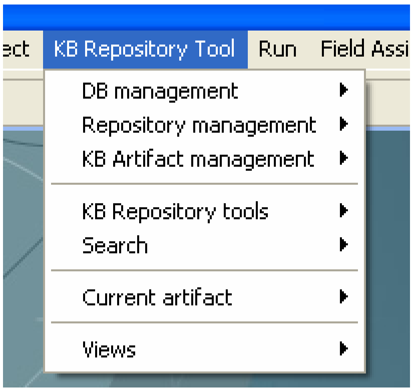


Figure 30 KB Repository main menu

“DB Management” submenu: Access the eXist server-side control commands (see detailed section 4.3.1)

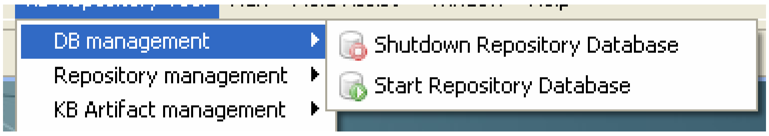


Figure 31 DB Management menu

"Repository management" submenu: this menu provides a command to create KB Repository elements of type "Repository", note that "Repository" type elements are "top level" element within the KB element hierarchy (see 4.2.3).

"Repository management" submenu: provides access for the instantiation of new KB artefacts using wizards (see 4.2.3).

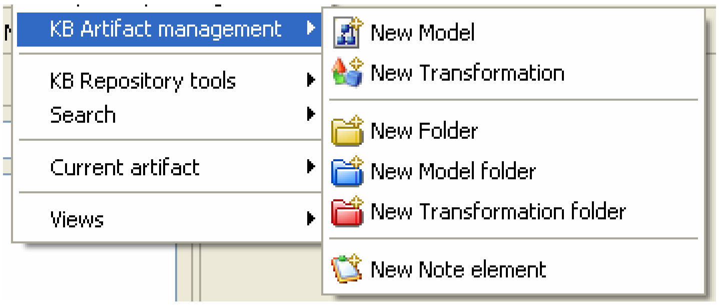


Figure 32 KB Artefact management

"KB Repository tools" submenu: Export / Import tools commands (See 4.3.2).

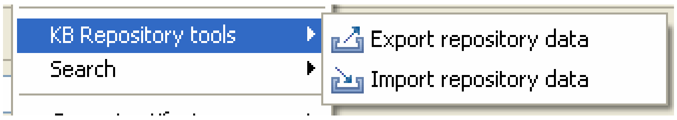


Figure 33 KB Repository tools menu

"Search" submenu: the search command will open the Search dialog.

"Current artefact" submenu: offers commands to rename or remove the currently selected KB artefact.

"Views" submenu: provides the user commands for opening each view individually. Since the KB perspective if fully customizable, views can be closed, moved, etc. To display a hidden view or bring it to front just select the desired one choosing the respective command at this submenu.

KB Repository button toolbar

The KB Repository button toolbar is shown in next pictures. It provides four buttons to perform some actions:

Repository artefacts creation button: opens a drop down menu supporting the creation of the commonest artefacts (i.e. any sort of folder, models, transformations, etc). See section 4.2 for more details.

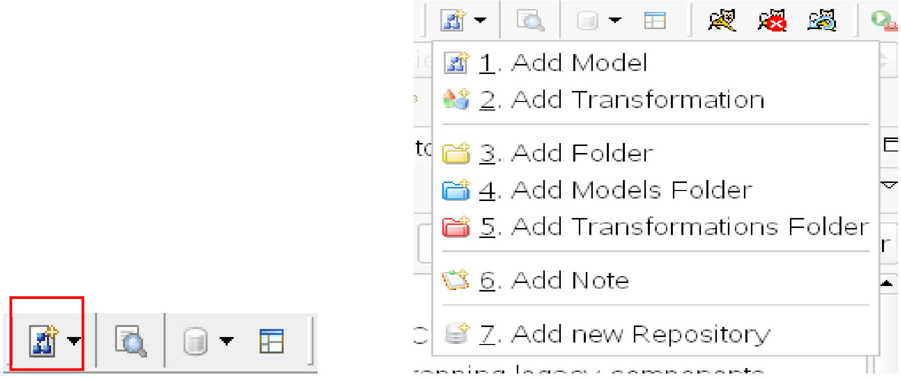
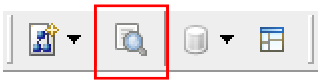
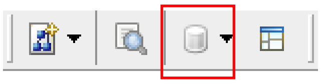


Figure 34 KB Repository button toolbar

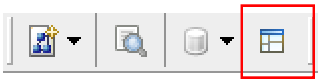
Repository search button: opens a wizard to perform a search under the folder selected in the KB Explorer view. See section 4.4 for more details.



Repository database management button: opens a drop down menu supporting the start up and shutdown of the local repository database. See section 4.3.1 for more details.



KB Repository perspective button: opens the KB Repository perspective.



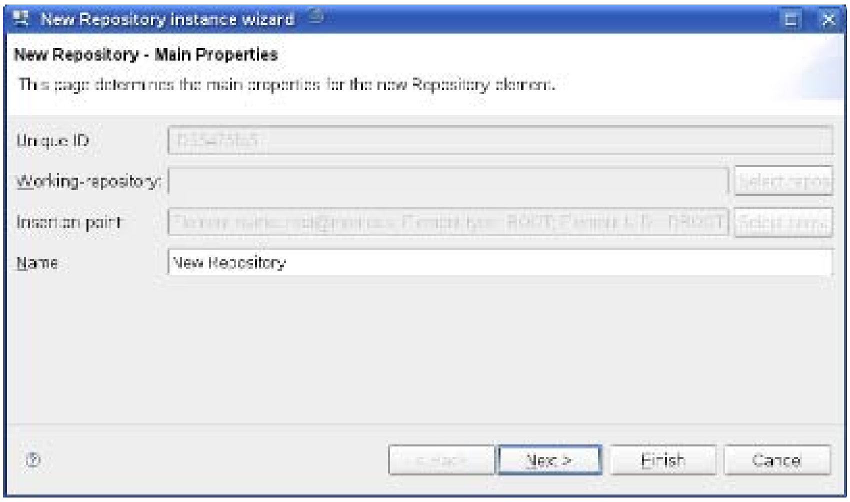
4.2 KB Repository administration guide

4.2.1 KB Repository domain model elements

KB Repository is a centralised container of main artefacts created and used during the XIRUP Modernisation process [2007c]. KB Repository can manage several thematic repositories. Each repository can be structured in a tree based folder structure, using normal folders (for creating the repository structure), model folders that may contain models and other model folders, transformation folders that may contain transformation rules and other transformation folders. Those elements constitute the basic artefacts contained within the KB Repository tool, which are relevant in the XIRUP modernisation process. Additional artefacts supported by the tool are notes, linked to each of aforementioned artefacts to complement them with additional information and attachments, which can also be linked with models and transformations, to complement them with additional artefacts (for instance, layout descriptions of models, matching logs on applied transformations, etc).

4.2.2 Repository creation / removal

KB Repository can manage different repositories stored within the database. Those repositories are shown in the KB Repository Explorer. To create a new repository click on this icon in the KB Repository Explorer view toolbar. Then a wizard opens (Figure 35). In the first wizard page we can edit the name of the repository. Next wizard pages are common to most of the KB Repository Explorer view wizards for creating artefacts. Second wizard page (Figure 36) allows the user to annotate the just created artefact (a repository in this case) with metadata: a set of keywords. Keywords can be added by clicking onto the "Add to list" tab button. To remove keywords or to move them up or down in the list, click "Remove", "Move up", "Move down" tag buttons respectively. Third and last wizard page (Figure 37) allows users to add comments describing the artefact to be created.



New Repository instance wizard

New Repository - Main Properties

This page determines the main properties for the new Repository element.

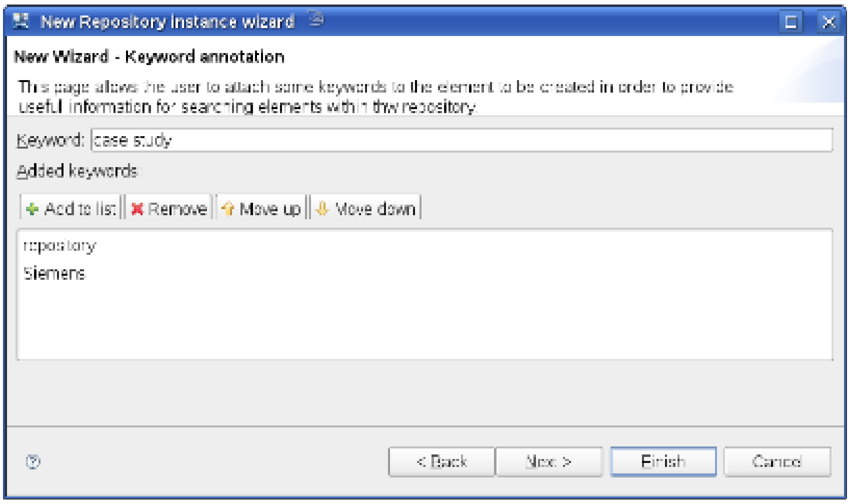
Unique ID: [illegible]

Working-repository:

Insertion point: [illegible]

Name: New Repository

Figure 35 New repository instace wizard


New Repository instance wizard

New Wizard - Keyword annotation

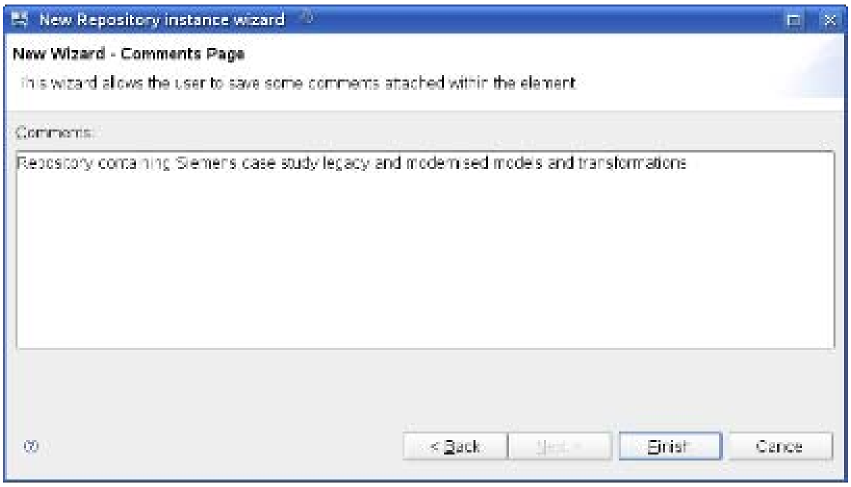
This page allows the user to attach some keywords to the element to be created in order to provide useful information for searching elements within the repository.

Keyword: case study

Added keywords

repository
Siemens

Figure 36 New wizard keyword annotation page


New Repository instance wizard

New Wizard - Comments Page

This wizard allows the user to save some comments attached within the element

Comments:

Repository containing Siemens case study legacy and modernised models and transformations

Figure 37 New wizard comments edition page

4.2.3 Artefact creation

KB Repository organised in a tree based structure those artefacts produced and consumed by the XIRUP modernisation process. This section explains how to create and organised those artefacts within the repositories.

4.2.3.1 Folders

Each repository managed by the KB Repository explorer view can be organized within a structure of folders and subfolders up to any depth. KB Repository manages three types of folders:

- Normal Folder: used to organise the repository structure. It may contain any other kind of folder.
- Model Folder: used to contain only model artefacts. It can contain only other model folders or models.

Transformation Folder: used to contain only transformation artefacts. It can contain only other transformation folders or transformations.

To create a folder of any type select the appropriate parent folder (or a repository) entry in the KB Repository explorer tree view and then right click to open the contextual menu (Figure 38). Select the appropriate menu entry, depending on the type of folder the user wants to create. The contextual menu entries change depending on the tree view item selected.

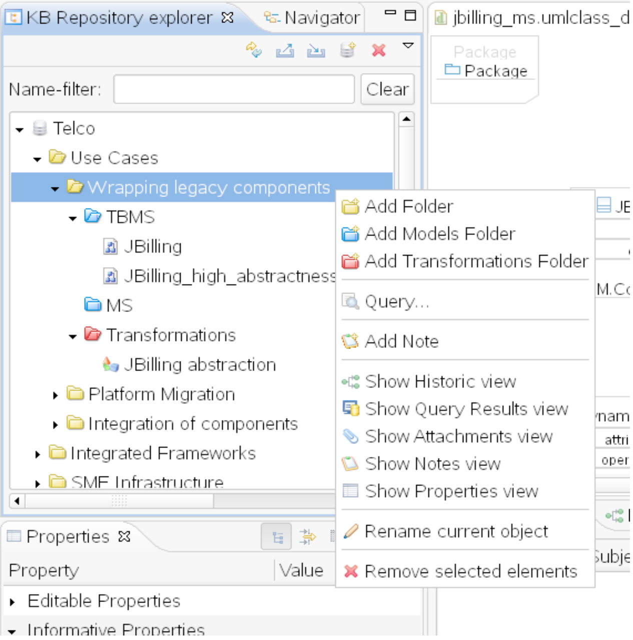
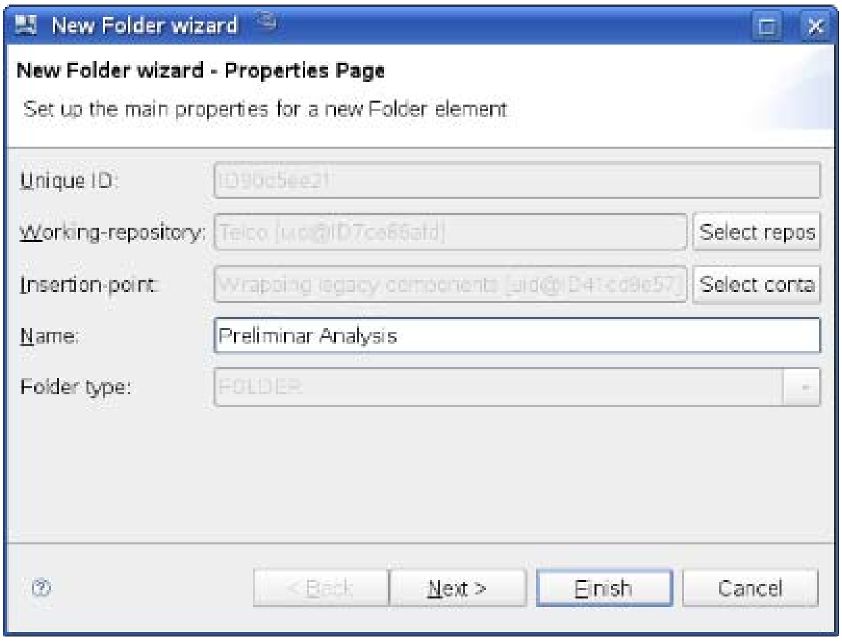


Figure 38 KB Explorer view contextual menu

After selecting the contextual menu item, a New Folder wizard opens. The first page allows the user to edit the name of the new folder (Figure 39). Next pages allows to annotate the new folder with metadata and comments as explained above (see section 4.2.2)



New Folder wizard

New Folder wizard - Properties Page

Set up the main properties for a new Folder element

Unique ID: ID90c5ee21

Working-repository: Telco [uid@ID7ca66afd]

Insertion-point: Wrapping legacy components [uid@ID41cd9e57]

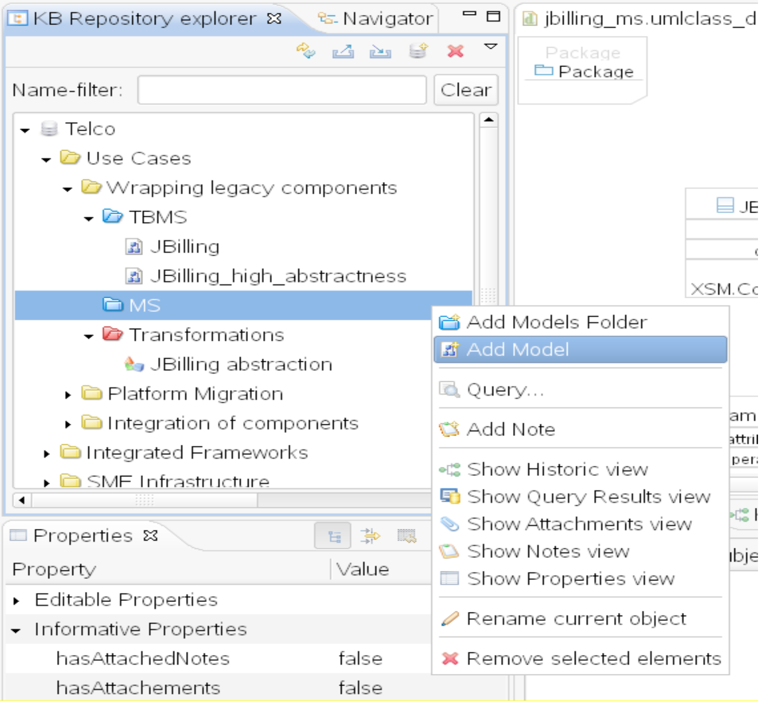
Name: Preliminar Analysis

Folder type: FOLDER

Figure 39 New folder wizard

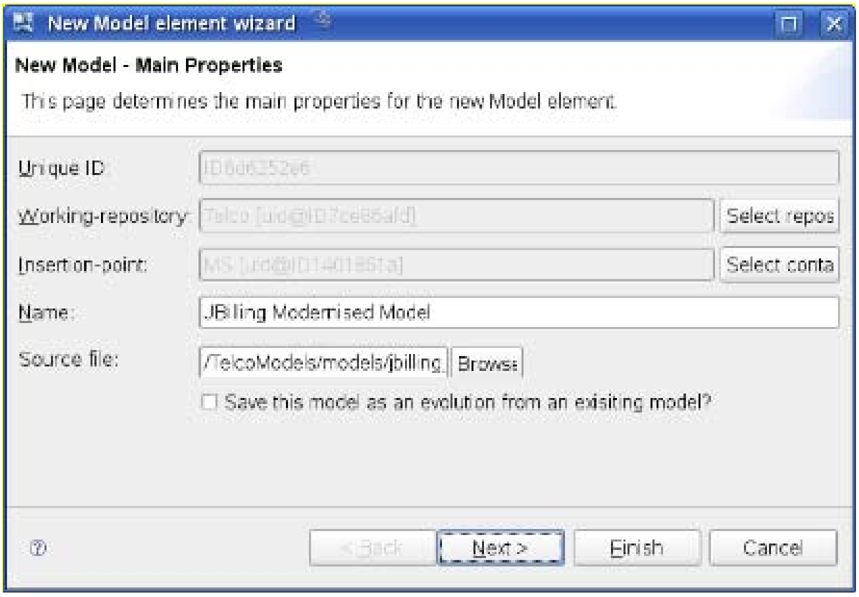
4.2.3.2 Model and transformation artefacts

Within models or transformations folders the user can add models or transformations respectively. Just by right clicking onto a models or transformation folder the opened contextual menu shows the appropriate menu item to create a new model or transformation (Figure 40). Selecting "New Model" or "New Transformation" opens a wizard. The first page (Figure 41) permits the user to give a name for the model/transformation and selects the source file from the workspace (Figure 42). Rest of wizard pages are the same than in case of other artefact creation wizards aforementioned.



Property	Value
hasAttachedNotes	false
hasAttachements	false

Figure 40 Add Model contextual menu



New Model - Main Properties

This page determines the main properties for the new Model element.

Unique ID: ID6d6252e6

Working-repository: Telco [uid@ID7ce86afd]

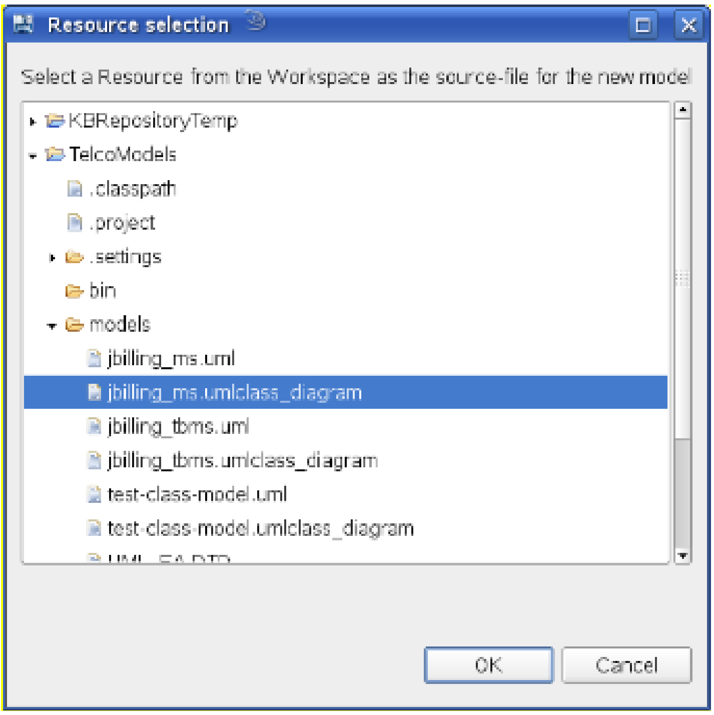
Insertion-point: MS [uid@ID1401861a]

Name: JBilling Modernised Model

Source file: /TelcoModels/models/jbilling

□ Save this model as an evolution from an exisiting model?

Figure 41 New model wizard



Resource selection

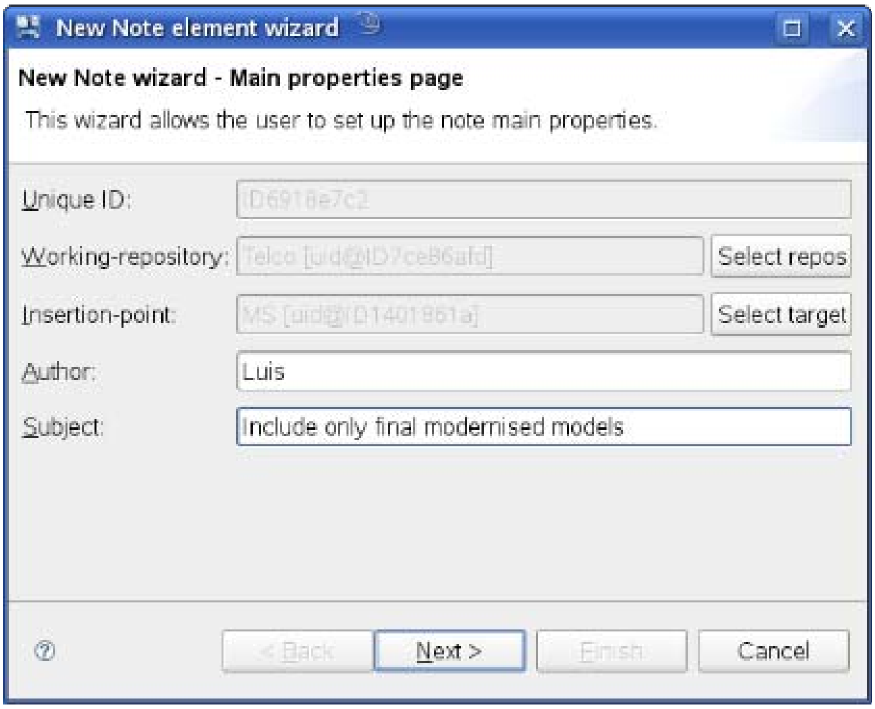
Select a Resource from the Workspace as the source-file for the new model

- KBRepositoryTemp
- TelcoModels
  - .classpath
  - .project
  - .settings
  - bin
  - models
    - jbilling_ms.uml
    - jbilling_ms.umlclass_diagram
    - jbilling_tbms.uml
    - jbilling_tbms.umlclass_diagram
    - test-class-model.uml
    - test-class-model.umlclass_diagram

Figure 42 Resource selection wizard

4.2.3.3 Notes and attachments

Notes can be created by selecting the appropriate repository artefact within the KB Repository explorer tree view, right clicking and selecting “Add Note”. The first “New Note” wizard page allows editing the note author and the subject (Figure 43). Second page is used to create the note content (Figure 44). Once created, the note is added to the KB Note list view in the auxiliary views panel (see section 4.2.3.3).



New Note element wizard

New Note wizard - Main properties page

This wizard allows the user to set up the note main properties.

Unique ID: ID6918e7c2

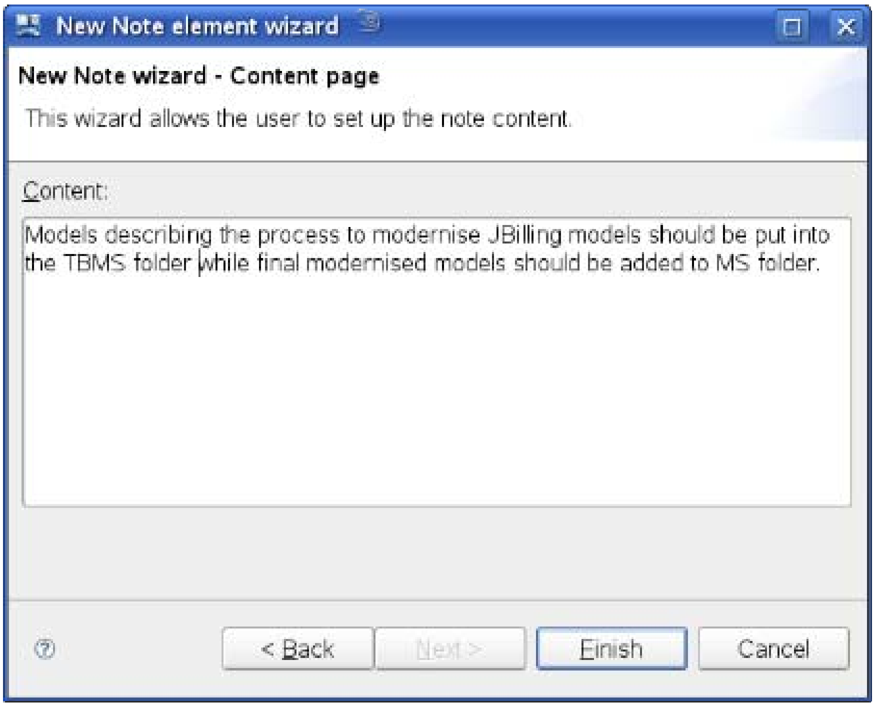
Working-repository: Telco [uid@ID7ce86afd]

Insertion-point: MS [uid@ID1401861a]

Author: Luis

Subject: Include only final modernised models

Figure 43 New note wizard



New Note element wizard

New Note wizard - Content page

This wizard allows the user to set up the note content.

Content:

Models describing the process to modernise JBilling models should be put into the TBMS folder while final modernised models should be added to MS folder.

Figure 44 New note wizard content page

Regarding attachments, this tool release does not support to add user-selected attachment, but those described in section 4.2.3.3. However, future releases of KB Repository tool will also allow attaching any file to any model or transformation added into the repository, to complement them. Attachments for a particular repository artefact are show within the KB Attachments view (see section 4.1.5.3).

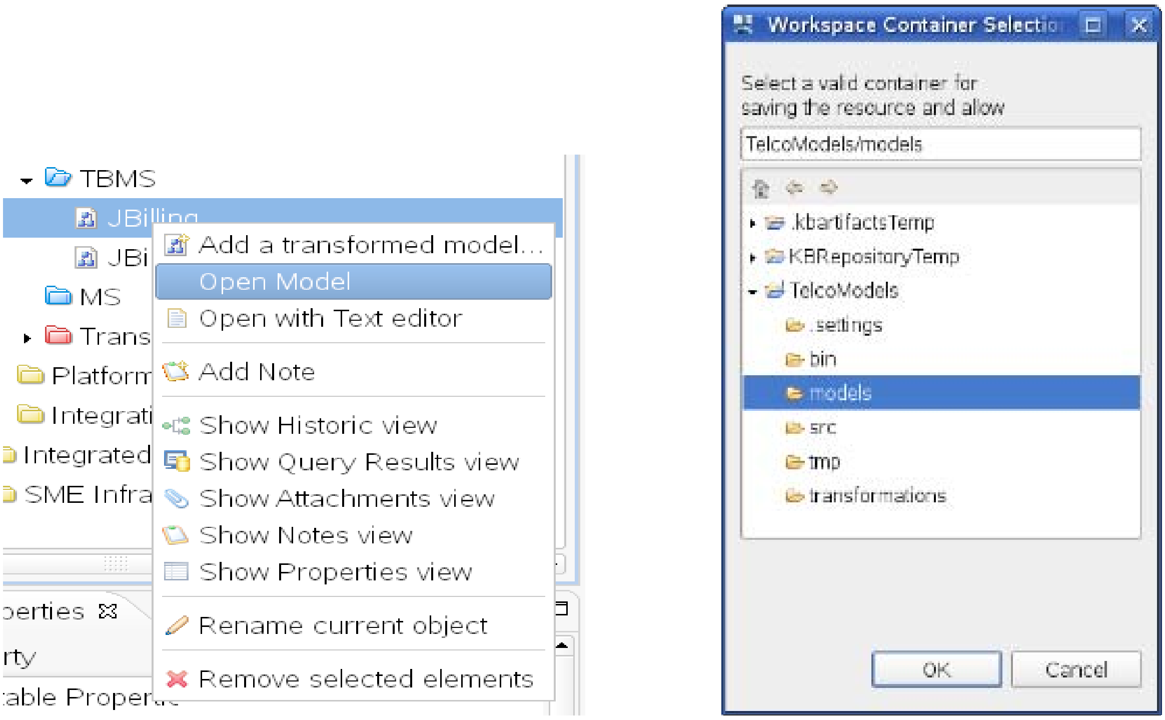
4.2.4 Artefact edition

Repository artefacts like folders, model folders, transformation folders, models, transformation, notes can be edited whenever is required to change some properties as name, keyword based metadata annotation, comments and so on, by selecting the artefact within the KB Repository explorer tree view and accessing the Eclipse properties view (usually located on the left bottom panel, see section 4.1.2). In the properties view, the user can modified all the editable properties.

4.2.5 Artefacts visualisation. Exporting Models and transformations

Some artefacts like models and transformations can be visualised by exporting them within the appropriate MOMOCS Suite tool: to visualise models, they have to be exported into the XSM Model Editor, and, for transformation into the XSM Transformation editor. Exporting models and transformations for visualisation means to export them into the current workspace, afterwards the appropriate tools is invoked passing to it the artefact path relative to the workspace, in order to be opened.

For example, to export a model within the workspace and to open it within the appropriate editor (similar procedure applies to open a transformation) right click on the model, select "Open Model" in the contextual menu.



Select a valid container for saving the resource and allow

TelcoModels/models

Figure 45 Open model contextual menu and workspace container selection wizard

In the Workspace Container Selection wizard select the folder within some project of the workspace where to export the model (and all its attached files). Then, those files will be exported into the selected workspace folder and the appropriate editor invoked. The model appears in the KB Repository perspective central view, opened by the editor (Figure 46).

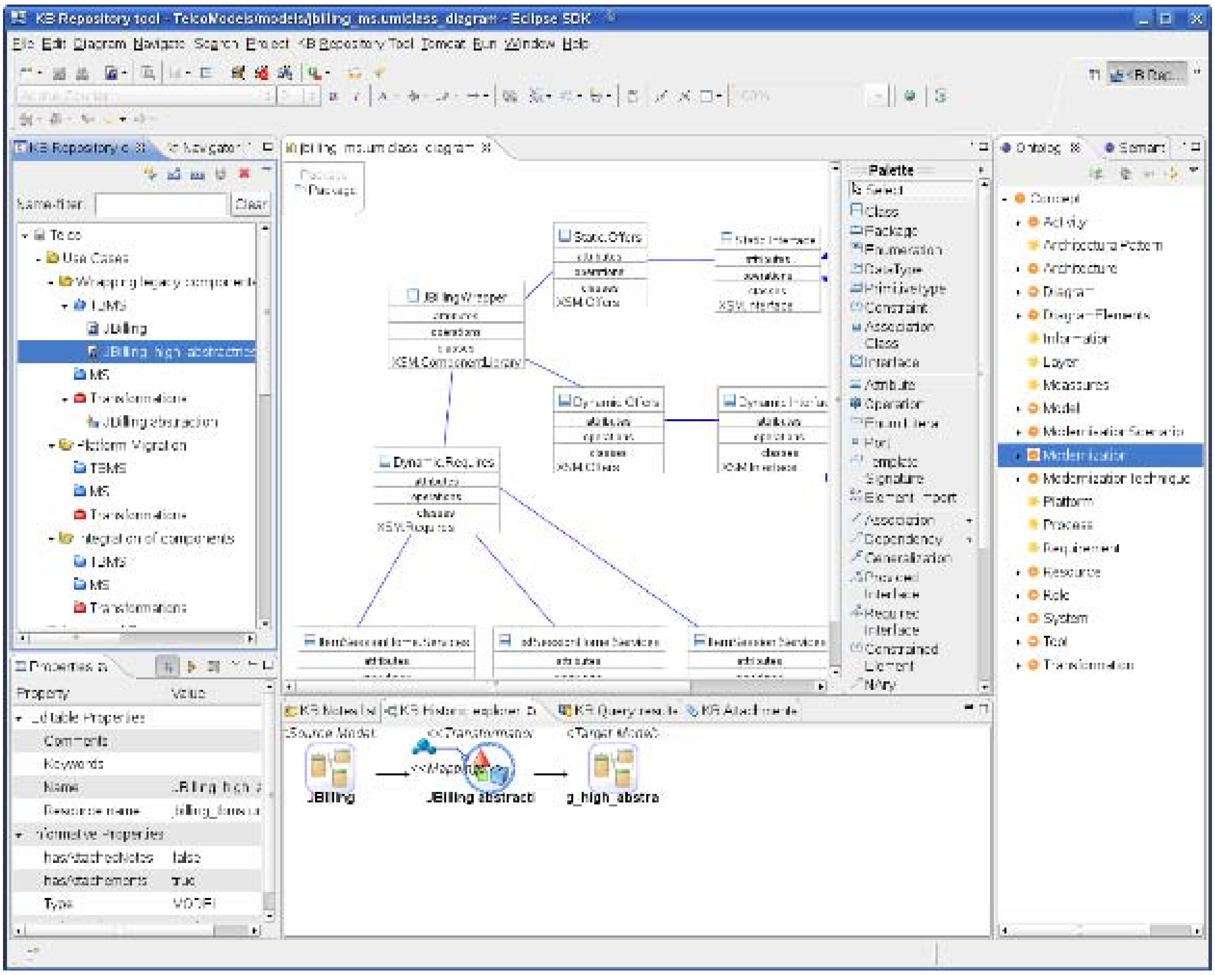


Figure 46 Model opened within the Momocs Suite workbench

4.2.6 Artefact removal

In order to remove any KB Repository artefact: repositories, folders, model folders, transformation folders, models, transformation, notes, etc. just right click on the artefact and select the contextual menu entry “Remove selected elements”. Confirm in the dialog. Note that this procedure can not be undone, so once deleted the artefact can not be restored.

4.2.7 Artefact annotation

Repository artefacts can be annotated with keyword based metadata when they are added into the repository, using the wizard (see section 4.2) or anytime after by using the Properties Editor view (see section 4.2.4). Additionally artefacts can be annotated anytime with semantic metadata using the Ontology Browser view (see section 4.1.3).

4.3 KB Repository auxiliary tools

4.3.1 Repository database management

KB Repository tool offers a basic management support for the local repository database:

To start the local database, choose from the eclipse menu: KB Repository Tool>DB Management>Start Repository database or the same option from the KB Repository tool bar (see picture below)

To shutdown the local database choose from the eclipse menu: KB Repository Tool>DB Management>Shutdown Repository database or the same option from the KB Repository tool bar (see picture below)

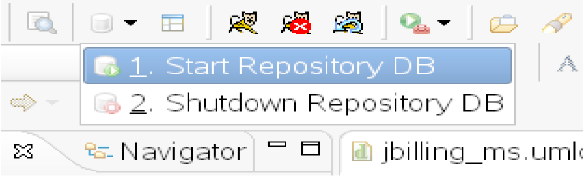
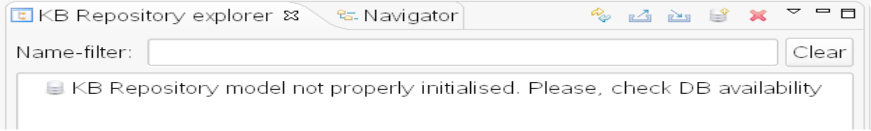


Figure 47 KB Repository database management toolbar menu

After shutting down the local repository database or in case that the local or remote database is not accessible, the KB Repository Explorer shows an error message.



KB Repository model not properly initialised. Please, check DB availability

Figure 48 KB Repository explorer showing database access error message

After starting the local repository database up or after refreshing the KB Explorer view once the remote makes available, the KB Explorer view removes this error message and shows the appropriate KB Repository content.

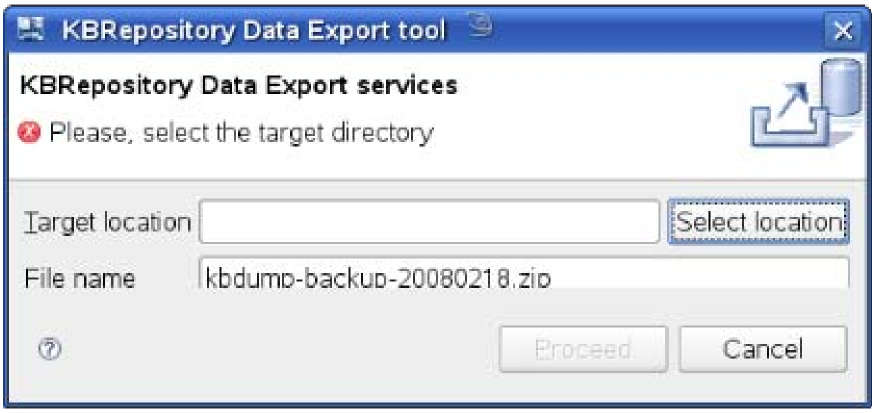
4.3.2 Repository export / import tool

KB Repository provides a facility to export the local database, creating a dump which can be later used to import its contents. This can be useful to migrate from one local database to another or to make a backup.

The procedure to export the local database[3] is as follows:

From "KB Repository Tool" menu select "KB Repository Tools/Export repository data" or click on the button of the KB Repository explorer view. In the opened wizard select a target location for the database dump backup (click "Select location" button to navigate through the file system) and "proceed" button. A zip file containing the database dump backup will be place into the selected directory.

[3] The local database must be started both during the export and import processes.



KBRepository Data Export tool

KBRepository Data Export services

Please, select the target directory

Target location:

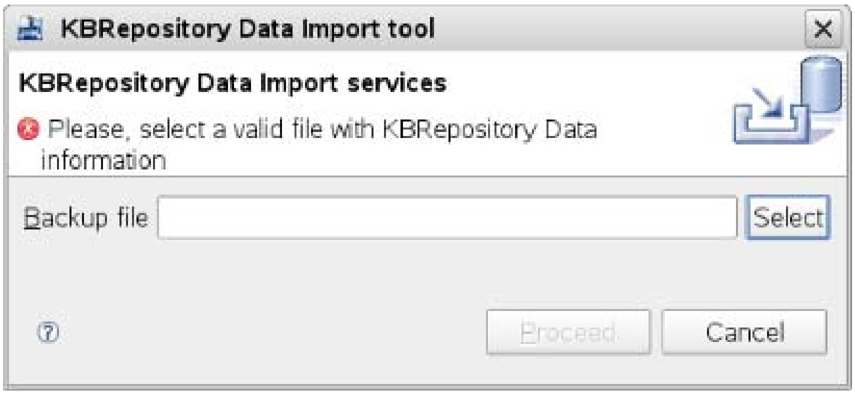
File name: kbdump-backup-20080218.zip

Figure 49 KB Repository export wizard

The procedure to import the local database is as follows:

From “KB Repository Tool” menu select “KB Repository Tools/Import repository data” or click on the button of the KB Repository explorer view.

In the opened wizard, select the database dump backup file by clicking onto the “Select” button and navigating through the file system looking for the appropriate backup. Then click on “Proceed” button.



KBRepository Data Import tool

KBRepository Data Import services

Please, select a valid file with KBRepository Data information

Backup file:

Figure 50 KB Repository import wizard

4.4 KB Repository Search Facilities

One of the most important aspects covered by this tool is information retrieval.

As said before a medium transformation process could result in a vast number of KB artefacts like models or transformations. Search facilities are given to users to easily locate an artefact within the repository structure and take advantage of the tool.

Both semantic (still in development) and keyword search engine are provided. Taking expertise in both of them engines will leverage the modernisation process by easily store, track, and retrieve the different KB artefact during its life-cycle and furthermore. Keyword search uses advanced information retrieval e algorithms based on keyword matching.

In order to recover a KB artefact effectively the user must annotate either with keyword metadata the artefact (see section 4.2.3) or using semantic metadata annotation (see section 4.1.3)

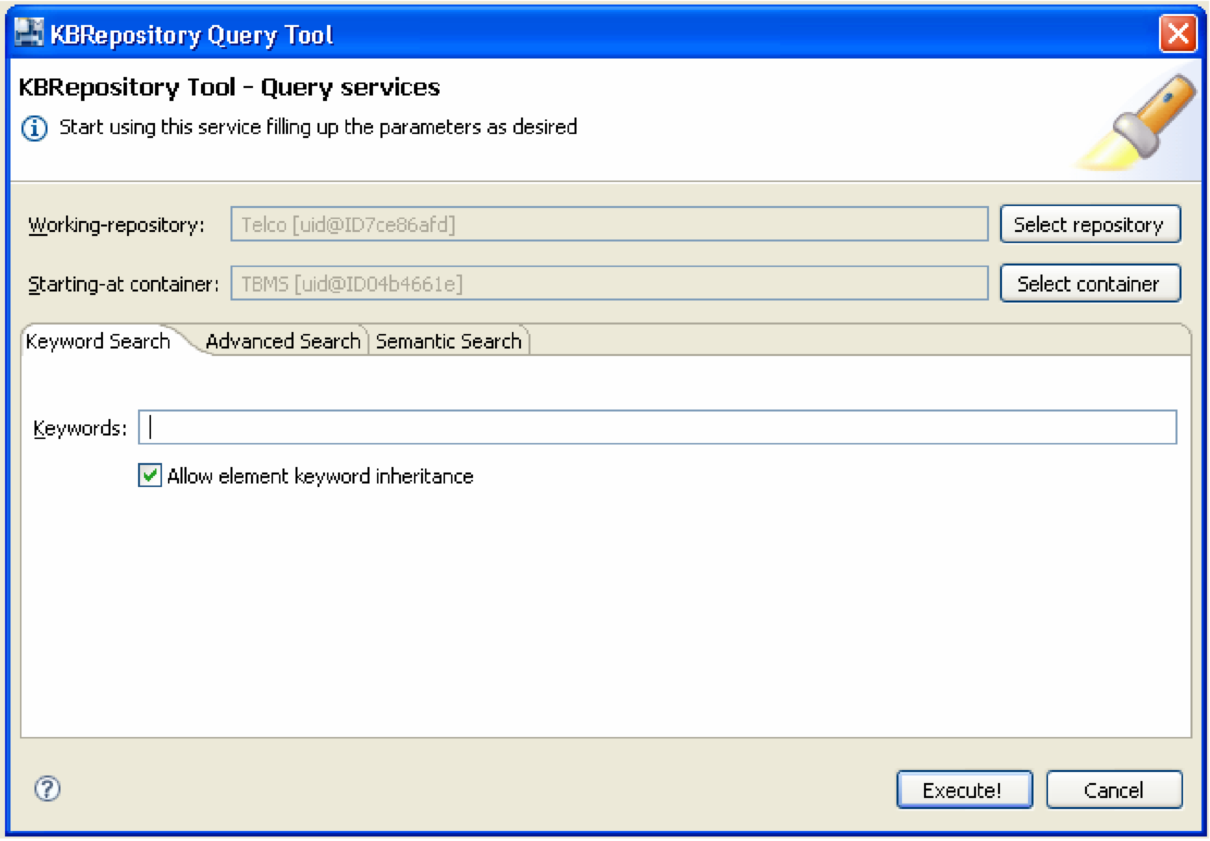
4.4.1 Keyword matching search

KB Repository uses the well-known and full-featured Open Source search engine named "Lucene" (lucene.apache.org), from the Apache group, to perform the keyword matching and KB artefacts indexing. In complex repository structures efficient information (object) retrieval workflows are vital to reduce time and mandatory to simplify the object location and selection. For this matter it is very important to take into account that the information (in the form of keywords in this case) the user attaches to a KB artefact is of vital importance for an optimal object recovery.

To start performing a keyword based search:

1. **Open the Search dialog**:

Select the "Keyword Search" tab, if not selected, and the keyword dialog will be ready to use:



KBRepository Query Tool

KBRepository Tool - Query services

ⓘ Start using this service filling up the parameters as desired

Working-repository: Telco [uid@ID7ce86afd]

Starting-at container: TBMS [uid@ID04b4661e]

Keywords:

☑ Allow element keyword inheritance

Figure 51 KB Repository query tool wizard

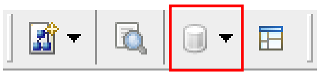
2. **Select the container to search within**:

Before searching, select a container where to start. Most of the times selecting an optimal starting point is the best option rather than querying the whole repository structure, by doing this the potential results and query time are significantly.

To proceed, first select the target repository by pressing the "Select repository" button and choose the desired one; second, select a KB container from within the repository.

This procedure applies to the all the searching methods offered by the KB Repository tool.

3. **Continue** by pressing the button "Proceed" at the Search dialog. The results will be automatically shown at the KB Query results view.

Tip: It is easier to select the "Starting-at" target container while browsing the repository using the KB Repository Explorer view. Select a container (where it is supposed to be located the requested element/s) and press the "Search" button from the KB Repository main toolbar  or right-click the target container and select the option "Query…".

4.4.1.1 Search syntax

KB Repository tool uses Apache's Open Source Lucene search engine (lucene.apache.org), one of the most widely used OSS search engine that offers a lot of indexing and searching capabilities. Thus, the Lucene special syntax applies for querying the KB Artifacts.

Here below some examples to illustrate how to query and find inside the KB Repository.

A Simple query:

- To match all KB artifacts having the keyword "DB", just insert the text "DB" at the "Keyword" text field and press "Proceed".

- To match exactly a phrase in a keyword use the quotation marks. Executing the query: “Only Modernised models” will search those elements having a keyword with the exactly given text.

Important note: Space characters inside a query string are used as a Boolean “OR” clause connector (see usage below), e.g. the query string: DB CORE, will match those elements owning a keyword with the value “DB” or the value of “CORE”.

Wildcards:

KB Repository tool keyword search facility supports single and multiple character wildcard searches.

- To perform a single character wildcard search use the "?" symbol.

- To perform a multiple character wildcard search use the "*" symbol.

The single character wildcard search looks for keyword that match that with the single character replaced. For example, to search for "text" or "test" you can use the search: `te?t`

Multiple character wildcard searches looks for 0 or more characters. For example, to search for test, tests or tester, you can use the search: `test*`

You can also use the wildcard searches in the middle of the keyword. `te*t`

Important note: You cannot use a * or ? symbol as the first character of a search.

AND, OR clauses: Boolean clause connectors are available

- AND: The AND operator matches documents where both keywords exist anywhere at the KB artifact keyword list. The symbol && can be used in place of the word AND.

To search for elements owning both keywords "modernised" and "j2ee" use: "modernised" AND "j2ee" query string.

- OR: this is the default conjunction operator. The symbol || can be used in place of the word OR.

To search for documents that contain either "fixed model" or just "modernised" use the query: "fixed model" modernised.

Complex queries:

- Just play with mixing all of the explained query methods above. Supposing we have divided a modernisation process into different step and annotated each element with “Step XX” keyword and also with “Modernised” or “To Be Modernised” keyword, for recovering all modernised KB artifacts involved at modernisation step 10 to 20 we could use the query string: Step 1? AND Moder*.

4.4.1.2 Query creation

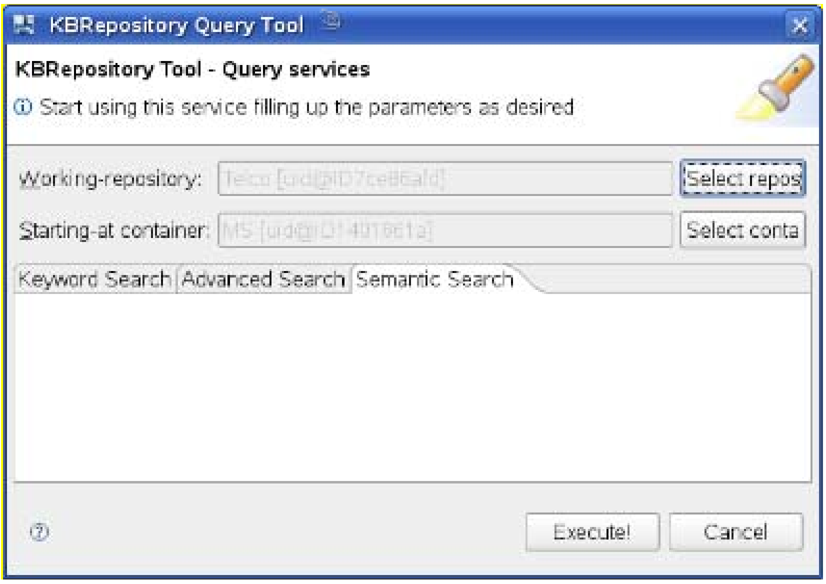
This feature is under development and will be integrated into the “Advanced Query” tab at Search dialog.

4.4.1.3 Browsing results

Results of querying into the repository can be browsed within the KB query results view (see section 4.1.5.2).

4.4.2 Semantic Search

This feature is under development and will be supported in next releases of the KB Repository Tool. Once available this section will describe how to define intelligent search criteria, how to perform an intelligent search and how the results are scored and shown to the user.



KBRepository Query Tool

KBRepository Tool - Query services

Start using this service filling up the parameters as desired

Working-repository: Telco [uid@ID7ce86ald]

Starting-at container: MS [uid@ID14018б1a]

Figure 52 KB Repository query tool. Semantic search wizard

5 Troubleshooting

5.1 Known issues

Under certain circumstances on Windows platforms, it has been observed sometimes that the internal KB Repository database (an instance of eXist xml database) hangs up, when it is managed from the KB Repository. Attempts to shut it down from KB Repository tool fail, therefore it is required to restart the KB Repository itself. This phenomenon has not been observed for KB Repository tool running on Linux operating systems.

5.2 Bug reporting

This first version of KB Repository Tool has not been yet assessed by final users, hence, even if a bug tracking facility has been supplied for such reporting, there have not been reported bugs. Incoming versions of this document will summarise the bugs reported and additional information about fixing.

5.3 FAQ

Question: Current KB Repository tool release works well on both Windows and Linux operating system (or whatever other UNIX systems, or systems compatible with JVM, including Mac OSX, etc). However, only a Windows install facility has been provided. How to support other OS?

Answer: It will be soon made available an Eclipse update site that will support KB Repository tool Eclipse installations. This facility will require an additional standalone installation of the eXist database (included in the Windows install facility).

6 Appendixes

6.1 Ongoing Features

This section summarises main incoming features not yet implemented into the current release of KB Repository Tool:

- KB Repository Advanced Keyword search will provide the user the capability for building complex queries supporting set of clauses.
- Semantic Search: incoming KB Repository tool will provide intelligent search facilities based on semantic queries performed upon the semantic knowledge base created by manual annotation of the KB Repository artefacts. This semantic search support is described in 2007b.
- Concurrent access to local or remote KB Repositories database instance: current KB Repository implementation support both local and remote eXist database instances but only with one user access. Incoming releases will support concurrent access to the database instance, managing user's account, user's concurrent changes, commits, updates and so on.
- Query storage: at Search dialog, a new command will allow the user to execute and store, using a familiar name, the query being processed. Later the user would be able to easy locate and run his stored queries.
- Keyword inheritance: in a top-down inheritance sequence a KB artefact could inherit the parent's keywords. This could help the user to not repeat the keywords introduced before and to give an element more information in order to evaluate de relevancy level in queries.

- Attachment enhancement: at this moment attaching “extra” resources to a KB artefact is made automatically by the tool. In next releases, we will provide functionality to allow the user insert manually any kind of attachments.

7 Annexes

7.1 Acronyms and Glossary

Acronym	Meaning
XSM	XIRUP system model
TBMS	To be modernised system
MS	Modernised system
UC	Use case
TID	Telefonica I+D
KB	Knowledge base
RCP	Rich client platform
SWT	Standard widget toolkit
PIM	Platform independent model
CIM	Computer independent model
PSM	Platform specific model
JVM	Java virtual machine
XIRUP	eXtreme end-User dRiven Process
FAQ	Frecuent answered questions
JRE	Java Runtime Environment
URI	Uniform Resource Identifier
DB	Database
UML	Unified Modelling Language
OS	Operating System
OSX	Macintosh Operating System

Table 1 Acronyms and Glossary

7.2 Reference Documents

[2007a], F. Garijo, J. Pavón, C. Rodriguez: Momocs Telco use cases.

[2007b], Momocs team, D4.1 XIRUP supporting tools specification.

[2007c], Momocs team, D31 XIRUP Methodology

[eXist] Open Source Native XML Database. http://exist.sourceforge.net/