Objectiver User Manual

Objectiver 3.0

1.0 Introduction

Objectiver is a Requirements Engineering tool supporting the GORE/KAOS methodology designed to help analysts identify, formalize requirements and write requirements documents.

1.1 Prerequisites

The reader should have a basic knowledge of the GORE/KAOS approach. The intent of this guide is not to introduce the methodology but to show how to build a GORE/KAOS specification with the tool.

1.2 Conventions used in this book

The following icons are used throughout this manual to highlight new features and changes with respect to previous versions of the tool:

NeW is used to highlight new features of the tool.

Updated is used to highlight features, whose description has been modified wit respect to the previous version of this manual.

🍷 is used to highlight limitations of the current release.

💡 is used for hints and advices.

1.3 Table of contents

2.0 Overview

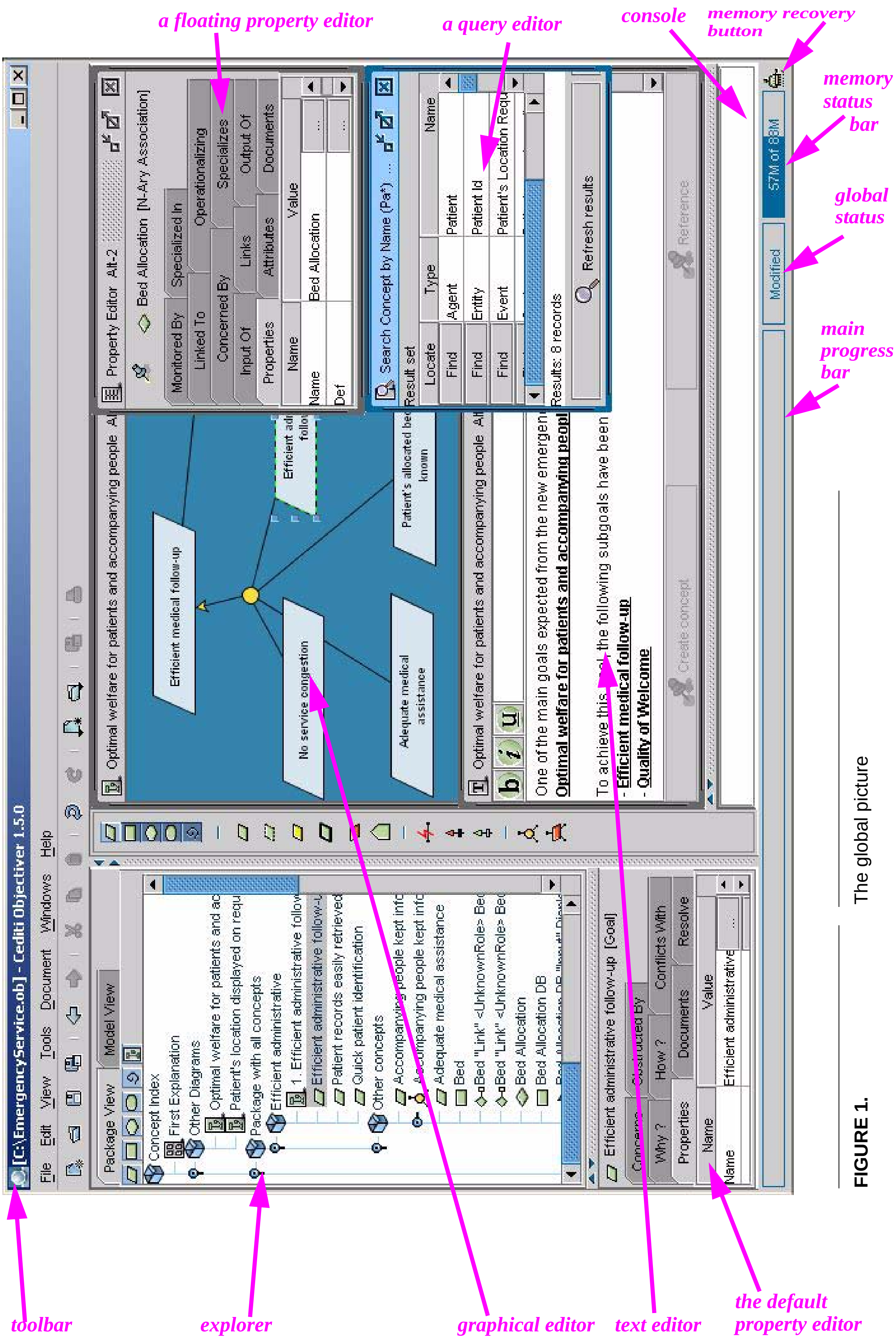
2.1 Objectiver components Composants Objectiver

Objectiver version 3.0 is made up of a number of components; including:

- a **diagram editor** to represent the concepts and their relationships (see Section 6.0).
- an **explorer** to browse through diagrams, text documents, and concepts (see Section 7.0).
- a **simple text editor** to write simple textual descriptions, for instance to write down interview summaries (see Section 8.0).
- a **rich text editor** to write more sophisticated textual descriptions than with the simple text editor, for instance, complex diagram explanations including tables, figures, and images (see Section 11.0).
- a **property editor** to assign values to the predefined attributes of a concept. You can create your own attributes. (see Section 9.0).
- a **query tool** to run queries against the current model (see Section 10.1).
- a **reporting tool suite** including:
  - a **web publisher** to create a distributable web version of your model. (see Section 12.0).
  - a **report editor** to create composite documents (see Section 13.0).
  - a **report publisher** to export reports in different formats (see Section 13.7).
- a **document generator** to generate diagrams, texts or reports (see Section 15.0).
- a **requirements document generation wizard** to create an editable version of the requirements document (see Section 16.0).
- a **console** used for reporting important messages to you.

2.2 File format

Objectiver projects are saved as OB files, which are compressed XML files.



Optimal welfare for patients and accompanying people Alt

One of the main goals expected from the new emergen
Optimal welfare for patients and accompanying peopl

To achieve this goal the following subgoals have been
- Efficient medical follow-up
- Quality of Welcome

FIGURE 1. The global picture

PART A Working with Objectiver

3.0 Project-level Operations

3.1 About projects

Objectiver is used to build requirements models. Each Objectiver project contains exactly one requirements model.

The whole project is stored in one file, the name of which ends with the .ob extension.

Only one project can be opened at the same time for editing. Modifications to the project are written back to file when the project is saved.

3.2 Creating a new project

Choose **File>New Project** to start a new project.

If a project is open and has been modified, Objectiver will ask you to save the current project before resetting the tool. You cannot indeed have two projects open at the same time.

Note: a new project is automatically created when you start Objectiver.

3.3 Opening an existing project

1. Choose **File>Open Project...**
If a project is currently being open and modified, Objectiver will ask you to save it before resetting the tool.
2. Select the directory where your project file resides.
3. Double-click the file corresponding to your project.

When you open a project, a lock is set to prevent other users from editing the same file at the same time. If someone attempts to open a locked project file, Objectiver opens it in a read-only mode and the icon beside *Concept Index* is orange.

Note: If the project contains at least one control unit (see Section 19.0 “Control units), Objectiver doesn’t lock the project and opens it in a read-only mode. To edit it, right-click *Concept Index* and select **Control Unit>Edit** in the context menu.

To edit a locked project file:

1. Right-click *Concept Index* and select **Control Unit>Edit** in the context menu.
2. You then have the possibility to either retry, or cancel the file opening operation, or reset the lock if nobody is still editing the project. So click **Reset the lock** if this one results of a crash of Objectiver.
3. Click **Yes** to confirm your choice.
If a user was editing this file, Objectiver warns him/her that lock is being seized by somebody else.

To edit a copy of a locked project file:

1. Choose **File>Save Project As...**
A file dialog window pops up and prompts you to enter a filename.
2. Right-click *Concept Index* and select **Control Unit>Edit**

You can also reopen a project you recently opened as follows:

1. Choose **File>Reopen**.
2. Select the project file name you want to open from the list that appears.

Note: With the MS-Windows Environment, the Objectiver installer associates the .ob file extension with Objectiver. When double-clicking an Ob file, Objectiver starts up and loads the project file.

3.4 Saving a project

Choose **File>Save Project**

If the project already exists on file, Objectiver converts the model to XML and saves it to the current project file.

Otherwise, a file dialog window pops up and prompts you to enter a filename.

Backup files. When saving your document, Objectiver keeps a backup of several previously saved states of the file. Backup file names consist of the project name followed by a backup identification number, followed by the "ob_bak" file extension. This allows you to retrieve an older version of your project file.

Objectiver manages backup files as follows: the last ten saved versions (identified by a number: file--XX--.ob_bak, e.g. file--45--.ob_bak, ...) are kept; older backup files are erased except one backup file for each day of the last week, and one backup file per preceding week (daily and weekly backup file names are identified by a date: file--2003.12.15--.ob_bak for instance).

3.5 Saving a project under a different file name

Choose **File>Save Project As...**

3.6 Working with shortcuts

You can create shortcuts to files or folders you frequently edit. Shortcuts can be created when you invoke the File manager dialogue from the **Open Project** or **Save Project As** menu. Proceed as follows:

1. Choose **File>Open Project...** or **File>Save Project As...**
2. Select the file or folder for which you want to create a shortcut.
3. Click **Add** in the **Shortcuts** panel.

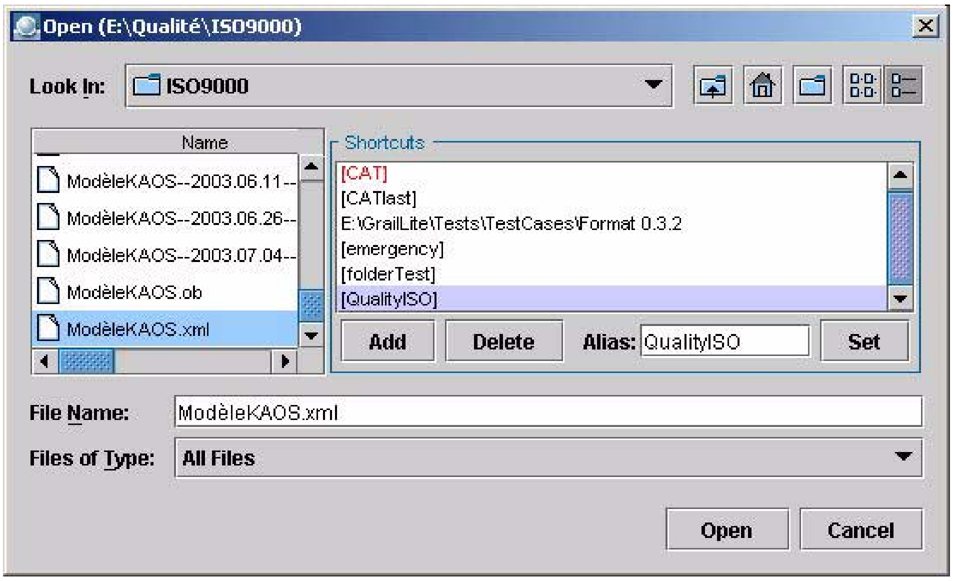
A shortcut is added to the shortcuts list. You can give alias names to long shortcuts by entering a string in the alias field and by clicking "Set". You can also assign a color to specific entries by preceding the alias name with the color name followed by the pound (#) character, e.g. "red#QualityISO".

3.7 Closing a project

Choose **File>Close Project**

3.8 Renaming a project

1. Make sure the project is not being edited with Objectiver.



Open (E:\Qualité\ISO9000)

Look In: ISO9000

Name

ModèleKAOS--2003.06.11--
ModèleKAOS--2003.06.26--
ModèleKAOS--2003.07.04--
ModèleKAOS.ob
ModèleKAOS.xml

Shortcuts

[CAT]
[CATlast]
E:\GrailLite\Tests\TestCases\Format 0.3.2
[emergency]
[folderTest]
[QualityISO]

Alias: QualityISO

File Name: ModèleKAOS.xml

Files of Type: All Files

FIGURE 2. Creating a shortcut to a file

2. Rename the file in which your project resides (the one with the .ob extension) by using the appropriate operating system command.

3.9 Exiting Objectiver

Choose **File>Quit**.

If the current project has not been saved since your last modifications, Objectiver asks you whether you want to save the project before terminating the application.

4.0 Document-level Operations

4.1 About documents

An Objectiver project consists of a set of interrelated concepts and documents. There are three kinds of documents: diagrams, texts and reports. *Diagrams* represent different concepts and the relationships among them. Text documents contain textual information with references to the concepts in the model. Styles can be used like in most general-purpose text processors. Reports are composite documents combining texts, diagrams and reports. They are either created manually or automatically, based on predefined templates.

All documents are stored in the project file. They can be manipulated using the Document menu and the Document toolbar.

This section details the common operations provided to manipulate documents. Specific operations for Diagrams are detailed at Section 6.0, operations for Texts at Section 8.0, and operations for Reports at Section 13.0.

4.2 Creating a document

1. Click the **New Document** icon in the toolbar.

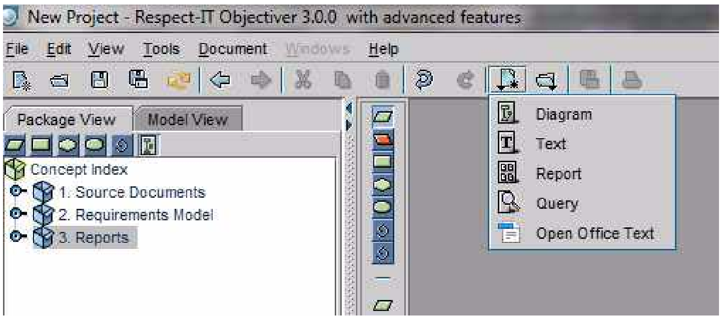
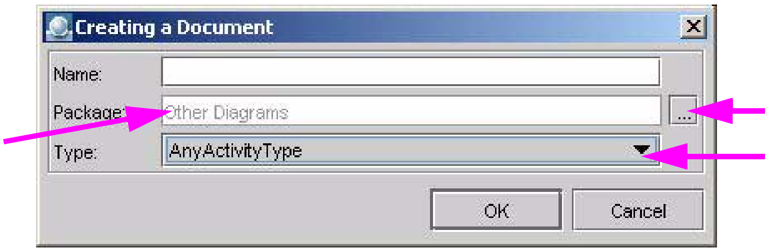


FIGURE 3. Create a new document

2. Choose whether you want to create a diagram, a text or a report.
3. Objectiver prompts you to specify the document name, package and type (Figure 4).



Name:

Package: Other Diagrams

Type: AnyActivityType

Name of the package in which the document will be created

Choose package button

Diagram Type

FIGURE 4. Enter the document name

4. To select the package in which the document will be created, click the “**...**” button, select a package and click **Select** (Figure 5). The package selected by default is the one currently selected in the explorer.

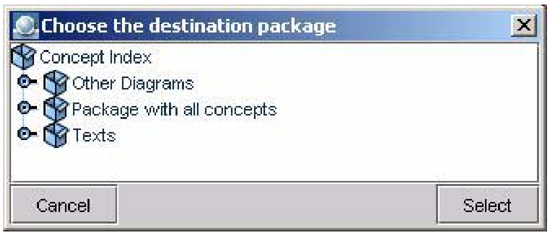


FIGURE 5. Choose the destination package

5. When you create a diagram, you can associate an activity type to this one. You will learn more about activity type in Section 6.4 and Section 6.5.
6. Click **OK**. An editor window opens up.

4.3 Opening a document

1. Click the **Open Document** icon in the toolbar.

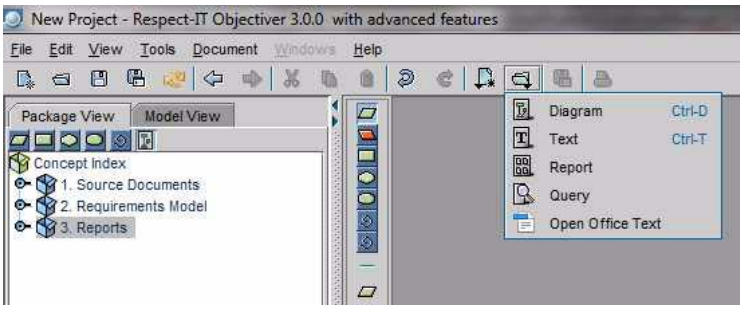


FIGURE 6. Open an existing document

2. Select the document type. A dialog box containing the list of all documents of this type in the project appears.
3. Select the document you want to open by clicking on its name in the list. You can also use the Auto-completion mode to select the document name (see Section 6.7.1).
4. Click **OK** or double-click the document name. Objectiver then loads the document.

A document can also be opened by double-clicking on its name in the explorer (see Section 7.4).

4.4 Saving a document

Documents are project components, hence they are saved on disk when you save your project.

4.5 Cloning documents

1. In the explorer, Right-click the document you want to clone.
2. Select **Clone** in the context menu. Objectiver prompts you to specify the document name (Figure 4).
3. Enter the document name and click **OK**

You can also click **Clone Document** in the toolbar to duplicate the currently selected document.

If you only want to copy some concepts from a diagram to another, refer you to Section 6.7.3.

4.6 Deleting a document

Deleting a document from a specification can be done from the explorer (see Section 7.6.2).

4.7 Renaming a document

Use the property editor or the explorer to rename a document (refer to Section 7.7 to perform this operation).

5.0 Help

5.1 Displaying the Help window

Choose **Help>Help...** (or press F1). Objectiver opens the Help window. If you press F1 while an editor has the focus, the help window displays the documentation corresponding to this editor.

5.2 Using the context help

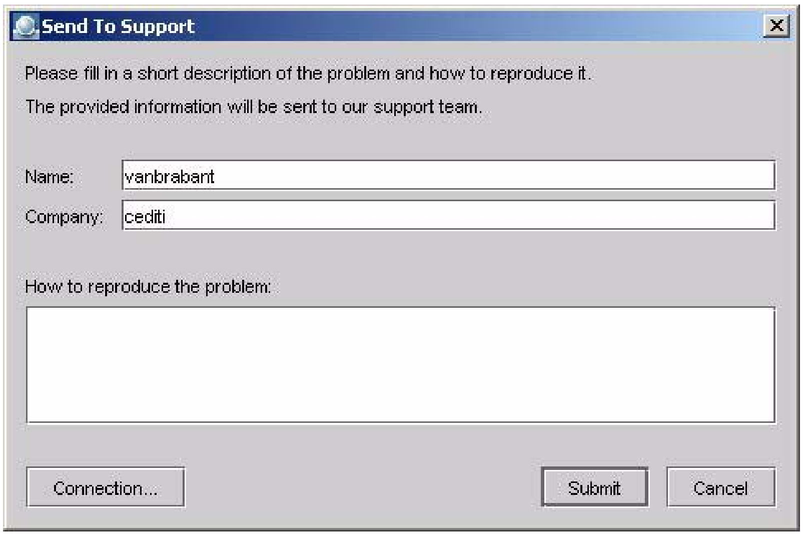
To get help for a menu item, you need to first click **Help on item** in the help menu, then open the menu that contains the item of interest using the keyboard accelerator sequence (e.g. Alt-f for the “File” menu). Finally use your mouse and click the item to get the help for it.

Note: if you attempt to open the menu using the mouse, the help window shall be displayed for it instead.

5.3 Send a problem description or a suggestion to support

Choose **Help>Problem or Suggestion**. Figure 7 shows a screenshot of the “Send To Support” frame.

If you use a Proxy server, you will have to fill in the connection preferences (see Figure 23).



Send To Support

Please fill in a short description of the problem and how to reproduce it.
The provided information will be sent to our support team.

Name: vanbrabant

Company: cediti

How to reproduce the problem:

FIGURE 7. Send to support

PART B Basic Modelling functions

6.0 Diagram Edition

6.1 About diagrams

A diagram is a type of Objectiver document. Diagrams are used to represent graphically the concepts and the relationships existing between them. Diagrams are **views** on the model composed by the analysts: the concept (and relationship) representations appearing in a diagram are indeed **references** to the concepts (and relationships) existing in the model. Some diagram operations concern the view only: for instance, moving a concept from one position to another in a diagram; other operations impact both the views and the underlying model: for instance renaming a concept.

This also explains why there are 2 delete operations: a first one for removing a concept reference from a diagram (without removing it from the underlying model) and another one for removing a concept from the model and consequently from all the diagrams in which it appears (see Section 6.10).

New concepts can be created in the model and a reference inserted in a diagram using the concept palette. References to existing concepts can also be added to any diagram.

This section details specific operations related to diagram handling.

6.2 Basic Operations

Refer to Section 4.0 to learn how to create, open, clone, delete or rename a diagram.

6.3 Diagram Editor toolbar

Figure 8 shows the diagram editor toolbar. The toolbar provides access to the following functions:

- adding a reference to an existing concept
- removing the selected concept references from this diagram
- applying an automatic layout algorithm to your diagram.

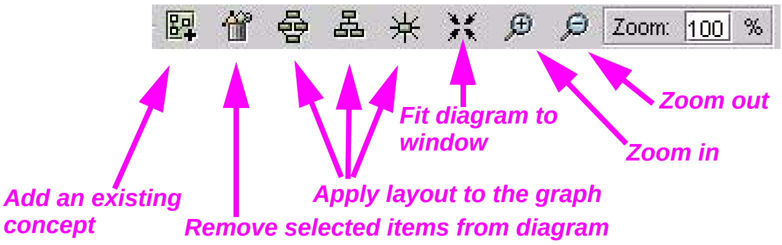


FIGURE 8. Diagram Editor toolbar

6.4 About the concept palette

There are five main types of modelling activities in Objectiver: the goal modelling, the obstacle analysis, the object modelling, the agent modelling and the operation modelling. Concept icons are grouped by activity in the *Concept palette.* On top of the concept palette, there are seven toggle buttons to choose the modelling activity. The sixth one provides access to additional notations that are not specifically related to the methodology (for instance notes) and the last one enables you to attach review notes to some concepts. Figure 9 shows the obstacle analysis palette.

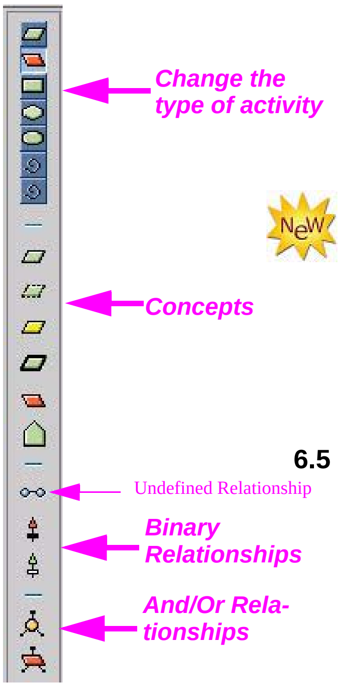


FIGURE 9. Concept Palette

Note about the palette orientation . You can control the palette orientation (horizontal or vertical) from the Preferences menu. To change the palette orientation, proceed as follows:

1. Select *Tools/Preferences*
2. Select the wished orientation for the palette by unfolding the combo box named *Palette orientation* in the *General* tab.

6.5 Diagram activity type

When you create a diagram, you can choose an activity type. The activity name shall later be shown near the diagram name in the tool. If the *lockedType* property is checked you may only create and add in this diagram concepts and relationships belonging to the associated activity. Section 6.4, provides the list of the diagram activity types.

If the lockedType property is unchecked, you may introduce any type of concept or relationship in the diagram. Each time you modify a diagram, Objectiver changes the activity type according to the diagram content.

6.6 Creating Items

6.6.1 Creating concepts

1. In the *Concept palette*, select the type of concept.
2. Click inside the diagram window where you want to create a concept.
3. Give the concept a meaningful name.
4. Press **Return**.

Notes:

- You can lock the editor in a mode allowing you to create several concepts of the same type in a row. Shift-click (or click twice) on an item of the *Concept palette* to enter the multi-concept creation mode.
- You can exit this mode by clicking on the palette button once again.
- Please refer to the section concerning the notation in the *Guide to using the web-generated doc* manual for a description of the different types of concepts available.

An alternative way of creating new concepts makes use of the context menu as follows:

1. Right-click in the diagram.
2. Select **New** then the concept type you want to create.

6.6.2 Creating binary relationships

Prior to creating a binary relationship between two concepts, make sure that both concepts are part of the current diagram.

1. In the *Concept palette*, select the type of binary relationship you want to create. Refer to the following list:

TABLE 1. Types of binary relationships

Type	From	To
Binary Association	Object[a]	Object
Cause	Event	Operation
Concerns	Objective[b]	Object
Conflict	Objective, DomProp	Objective, DomProp
Control	Agent	Object
Input	Object	Operation
IsA	Object	Object
Link	N-Ary Association	Object
Monitoring	Agent	Object
Obstruction	Obstacle	Objective
Output	Operation	Object
Performance	Agent	Operation
Resolution	Objective	Obstacle
Restoration	Operation	Requirement

[a] Object is a generic name for Agents, Entities, Events and N-Ary Associations.

[b] Objective is a generic name for Goals, Requirements, Expectations, SoftGoals.

2. In the diagram, drag the mouse from the origin concept to the destination concept in accordance with the previous table.

Note: If the destination concept you want to connect to is present in the diagram but not currently visible in the editor window, you can drag the mouse in its direction. The window will automatically scroll until you find the destination concept and release the mouse button on it.

If you select the **undefined relationship** in the *Concept palette* and drag the mouse from a concept to another one, the model will be queried to determine which relationship(s) can be created between the concepts. If more than one relationship can be selected, a pop-up window prompts the user to select one of them. If there is no relationship available or if the user cancels the selection in the pop-up window, the tool proposes him/her to create a relationship in the other direction, if it is possible. If only one relationship is possible, this one is automatically created.

6.6.3 Changing relationship arguments

To change a relationship from concept A to concept B to a relationship from concept A to concept C, proceed as follows:

1. Select the relationship.

2. Drag the point connecting the relationship line to B and drop it on the center of C (you must see a hollow rectangle around C while you drop the connection point)

6.6.4 Creating AndOr relationships

You create AndOr relationships in the same way you create binary ones.

1. Select the type of relationship to create from the *Concept Palette*.
   Refer to the following list:

TABLE 2. Types of AndOr relationships

Type	Father	Son(s)	Color
Assignment	Objective[a]	Agent (only one son)	pink
Operationalization	Requirement, Expectation	Operation	blue
Refinement	Objective	Objective, DomProp	yellow
O_Refinement	Obstacle	Obstacle	orange
Responsibility	Requirement	Agent (only one son)	red

[a] Objective is a generic name for Goals, Requirements, Expectations, SoftGoals.

2. Drag the mouse starting from the *Son* concept to the *Father* concept.

If you want to add a concept to an existing AndOr relationship, drag the mouse from the *Son* concept to the circle representing the relationship.

Recommendation. The *Father* concept is typically drawn above the *Son* concepts.

An alternative and quick way of creating refinements with many sons makes use of the context menu as follows:

1. Select the *Son* concepts (click and drag to select several concepts).
2. Right-Shift-click on the *Father* concept to get the context menu.
3. Select **Refine with selection**.

Notes:

1. Goal refinement alternatives can be named:

Or-refinements can be named by giving a value to the Alt-Name attribute of the refinement in the property editor; the name is displayed on top of the refinement symbol.

2. Goal refinements can be marked as complete:

And-refinements can be marked as being complete by setting the *Complete* checkbox in the property editor for the And-refinement. A check mark (✓) is displayed in the middle of the refinement symbol.

6.6.5 Creating HyperText, Picture and Documentation Links

Objectiver allows you to associate any concept of a diagram to a picture, an hypertext or a reference to any Objectiver document. This can be made in a two-step process: first you have to create a concept of type **HyperTextLink**, **PictureLink** or a **DocumentReference** in the diagram. Then you have to connect the hypertext, picture or document to the concept of the diagram, you want to associate to, by creating a **DocumentationLink** relationship.

Step 1. Creating concepts of type HyperTextLink or PictureLink in diagrams is done in the same way as concept creation (refer to Section 6.6.1). In the concept palette, select the “HypertextLink” or “PictureLink” concepts from under the “Other Concepts” Palette. Section 7.5.1 explains how to add references to an existing Objectiver document into a diagram.

After creation of the HyperTextLink or PictureLink concept, click it in order to fill in the address property in the property editor. See Section 8.6 to have an example of the different possibilities. In the generated web documentation (see Section 12.0), these concepts are hyperlinks to the referenced documents. The concepts of type HyperTextLink or PictureLink can also be opened from Objectiver by right-clicking the concept and selecting **Open in external browser** in the context menu.

Step 2. Creating relationships of type DocumentationLink in diagrams is done in the same way as relationship creation (refer to Section 6.6.2). In the concept palette, select the “DocumentationLink” relationship from under the “Other Concepts” Palette.

Concepts having HyperTextLink, PictureLink or DocumentReference concepts associated with them through documentation links are displayed in bold in the Objectiver explorer (see Section 7.0). The hyperlinks and Objectiver documents linked to those concepts can be opened directly from the explorer by right-clicking the concept and selecting **Open document** in the context menu.

6.6.6 Refine a goal in a new diagram

1. In a diagram, right-click a goal and select **Refine in new diagram** in the context menu.
2. Objectiver prompts you to specify the name and package of the new diagram.

The operation

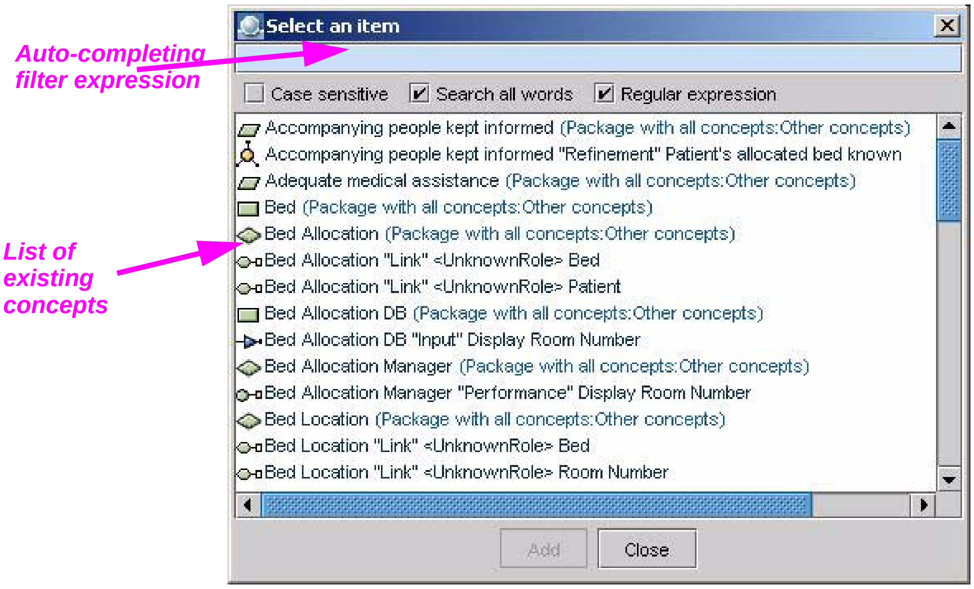
- removes the refinement subtree of the selected goal from the diagram including all other concepts and relationships (such as operationalization, responsibility and operationalisation) linked to it in the diagram
- recreate it in a new diagram
- introduces cross-references between the created diagram and the source diagram by means of two DocumentReference concepts linked to the selected goal (one document reference per diagram pointing to the other).

Model elements from the subtree connected to other elements in the source diagram including the selected goal remains in the source diagram. This operation available on any goal offers a powerful way to restructure quickly a complex goal hierarchy among different diagram in order to produce clear requirement documents.

6.7 Adding items

6.7.1 Adding references to existing concepts into a diagram

1. Click **Add existing** from the diagram editor group of buttons in the toolbar (Figure 8). A popup window appears with a list of all the concepts that can be inserted in this diagram (see for instance Figure 10).



Auto-completing filter expression

List of existing concepts

□ Case sensitive ☑ Search all words ☑ Regular expression

Accompanying people kept informed (Package with all concepts:Other concepts)
Accompanying people kept informed "Refinement" Patient's allocated bed known
Adequate medical assistance (Package with all concepts:Other concepts)
Bed (Package with all concepts:Other concepts)
Bed Allocation (Package with all concepts:Other concepts)
Bed Allocation "Link" <UnknownRole> Bed
Bed Allocation "Link" <UnknownRole> Patient
Bed Allocation DB (Package with all concepts:Other concepts)
Bed Allocation DB "Input" Display Room Number
Bed Allocation Manager (Package with all concepts:Other concepts)
Bed Allocation Manager "Performance" Display Room Number
Bed Location (Package with all concepts:Other concepts)
Bed Location "Link" <UnknownRole> Bed
Bed Location "Link" <UnknownRole> Room Number

FIGURE 10. List of existing concepts

2. From the list, select the concept you want to insert.
3. Click **Add** or double-click the selected concept.

6.7.2 Auto-completion mode

Another way to select entries from the list is to type the first few characters of what you are looking for. Then when you hit the space bar, the items displayed in the list are reduced to those that match your entry. More precisely, you can type any regular expression. With regular expressions, you can type wildcard characters providing a powerful pattern-matching mechanism. The most frequently used wildcards are the following ones:

. matches any character. Example: “ex..nsion” matches “extension”,”expansion”.

* matches zero or more occurrences of the previous character expression. Example: “.* ” matches anything.

+ matches zero or more occurrences of the previous character expression.

$ matches the end of the concept name. Example: “filet$” does not match “filet mignon”.

You can deactivate the special meaning of these wildcards by **deselecting** the check box “Regular expression”. You can also deselect the check box “Search all words” in order to only retrieve those entries whose **first** characters match those typed in the auto-completion line.

For instance, to retrieve a concept for which you know the first characters, proceed as follows:

1. Deselect the check box “Search all words”.
2. Enter the first characters of the concept name. If input is locked while typing (no characters echoed), it means that there is no matching concept.
3. Press the space bar or the Enter key to restrict the list only to those concepts, the name of which begins with the string you have typed.
4. Select one of the remaining concepts from the list or, if needed, keep typing characters and pressing the space bar to restrict the list further.
5. Click **OK** once you have selected a concept or press **Enter** if a single concept is found.

6.7.3 Notes:

- A concept may not appear twice in the same diagram.
- Inserting a relationship triggers the insertion of the concepts involved in the relation.
- You can also add concepts to a diagram by using the explorer (see Section 7.5.1).
- You can also add concepts to a diagram by using the Copy/Paste function.
  To copy some concepts from a diagram to another, proceed as follows:
  1. Select in the diagram the concepts to be copied.
     You can select many concepts by clicking and dragging the mouse, or by clicking on individual concepts while the Control key is depressed.
  2. Copy them (Ctrl-C).
  3. Paste them (Ctrl-V) in the other diagram.
- You can insert all connected concepts directly to some concept by right-clicking on it in the diagram and selecting **Add neighborhood** in the context menu.

6.8 Layout

The **Layout graph** buttons enable you to apply a new layout to the current diagram.

Recommendation.

- Apply **Tree** or **Clan** layout strategy to *Goal diagrams*
- Apply **Spring** layout strategy to *Operation and Responsibility diagrams*

Notes:

- You can change the layout spacing in Objectiver’s **Preferences** (see Section 6.11)
- Like all operations that make changes to the project, applying a layout can be undone.

6.9 Copy a diagram to the clipboard

To insert a diagram bitmap in the clipboard, right-click in the diagram background and select **Copy to clipboard**. You can then paste it in another program.

6.10 Deleting and removing items

6.10.1 Deleting a diagram

Deleting a diagram from a specification can be done from the explorer. See Section 7.6.2.

6.10.2 Removing concepts from a diagram

1. Select the concept to remove.
2. Click **Remove from diagram** from the diagram editor toolbar (Figure 8) or Press **Delete**. The concept and all attached relationships disappear from the diagram.
3. If the concept does not appear on any other diagram or text, Objectiver suggests you deleting it from the project (see Figure 15).

Notes:

1. This operation only removes concepts from a diagram, not from the model. Deleting a concept from the model is explained in the next section.
2. Relationships removed without their linked concept(s) are not really removed but simply hidden. To display them again, use the context menu: **Add hidden relations**.

6.10.3 Deleting concepts and relationships from the model

1. Select the concept(s) or relationship(s) to delete.
2. From the context menu choose **Delete from model** or Press Ctrl+Shift-Delete and then confirm the deletion.

You can also use the Explorer to delete elements (see Section 7.6.1).

Important notes:

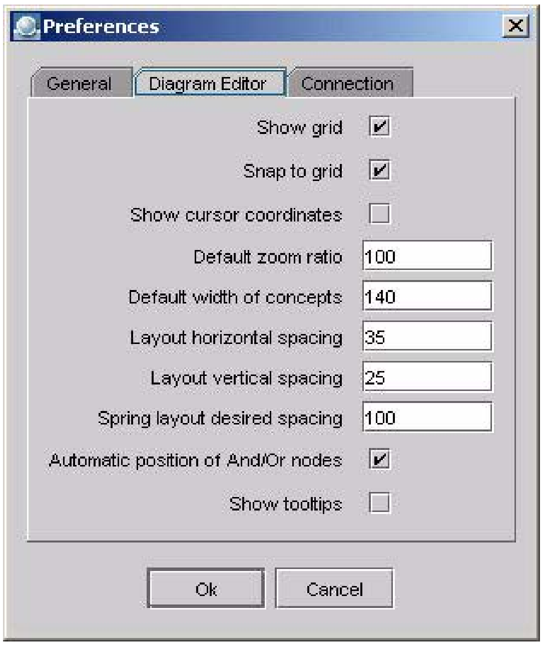
- After deletion from the model, all occurrences of the concept are removed from all diagrams and all attached binary relationships are deleted.
- All references to the concept in text documents (see Section 8.0) are replaced by a string: [Deleted: <Concept-name>].

6.11 Set Preferences for the Diagram Editor

1. Choose **Tools>Preferences**.
2. Select the Diagram Editor tab. (see Figure 11)

Different options are available on this tab to adjust the diagram editor to your needs.

Tooltips are defined on each field to help you understand their meaning.



Show grid ☑

Snap to grid ☑

Show cursor coordinates □

Default zoom ratio 100

Default width of concepts 140

Layout horizontal spacing 35

Layout vertical spacing 25

Spring layout desired spacing 100

Automatic position of And/Or nodes ☑

Show tooltips □

FIGURE 11. Preferences concerning the diagram editor

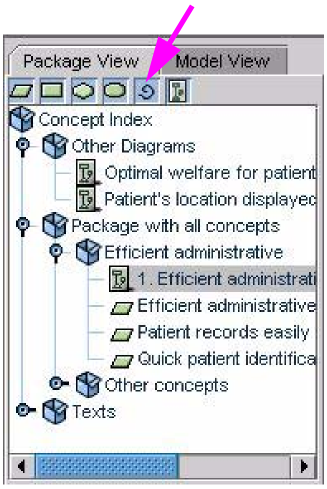
7.0 Exploring the model

7.1 About the explorer

An explorer allows you to browse the model. You can browse the model through the package structure package view or you can browse the model according to the concept types (model view).

You can locate a concept in the package structure from the diagram or text editor or by providing its name.

7.2 The explorer Contents


Show/Hide concepts of this model

Package View | Model View

Concept Index
- Other Diagrams
  - Optimal welfare for patient
  - Patient's location displayec
- Package with all concepts
  - Efficient administrative
    - 1. Efficient administrati
    - Efficient administrative
    - Patient records easily
    - Quick patient identifica
  - Other concepts
- Texts

FIGURE 12. Package View

The explorer contains a tree view of all the elements a project is made of.

There are two different views:

- **The Model View** is a hierarchical concept index providing:
  - on the 1st level, the following 5 model categories:

    1. Goal model

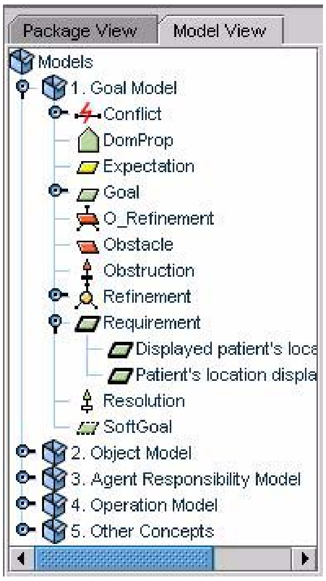
    2.Object model

    3.Agent Responsibility model

    4.Operation model

    5.Other concepts
  - on the 2nd level, the list of concept types present in the category. For instance, Figure 13 shows that the concept types relevant to the Goal model category are: Conflict, DomProp, Expectation, Goal, O-Refinement, etc.
  - on the 3rd level, the list of concept instances present in the model. For instance, Figure 13 shows that the current model contains only two requirements: *Displayed patient's location read* and *Patient's location displayed on request*.
  - Levels 1 and 2 are predefined, which means they are always present whatever the model. Level 3 lists the actual concepts in your model.
- **The package View**. Packages are concept containers. They are hierarchically structured. Section 7.3 details the notion of packages and how to defined them. The explorer (see Figure 12) displays the package hierarchy and the concept instances present in each package. It is similar to the file system of your operating system where packages stand for directories. Several filter buttons located in the upper left part of the explorer enable you to show or hide the concepts according to the model category you are interested in. (Note: if a concept is relevant to several model categories, it will be hidden only if all the corresponding filters are activated.)

You can switch between views by clicking the tabs in the upper part of the explorer.


Package View | Model View

Models
- 1. Goal Model
  - Conflict
  - DomProp
  - Expectation
  - Goal
  - O_Refinement
  - Obstacle
  - Obstruction
  - Refinement
  - Requirement
    - Displayed patient's loca
    - Patient's location displa
  - Resolution
  - SoftGoal
- 2. Object Model
- 3. Agent Responsibility Model
- 4. Operation Model
- 5. Other Concepts

FIGURE 13. Model View

7.3 Packages

7.3.1 About packages

Kaos includes the notion of package that gives an extra level of flexibility in structuring the content of an Objectiver project. Packages are containers for any type of concepts (including documents). Sub-packages can also be created offering a tree structure of containers which the analyst can use to best suit his needs.

Packages are created and visible in the **package view**. Inside a package, all the elements must have a different name. "Concept Index" stands for the root text package and cannot be renamed.

- To create a new package, use the context menu: Right-click the package in which you want to create a sub-package.
- You can move the documents and the concepts between packages using drag and drop.
- When you create a document, it is added in the package currently selected in the explorer (if you don't change the destination package see Figure 4).
- When you create a concept in a document, the concept is created in the same package as the document.
- When you add concepts already existing in the model to a document, those concepts remain in their current packages (they are never moved by the add operation). Therefore if the document belongs to another package than the one containing the added concepts, the document will refer to concepts that are outside the package which contains it. This property impacts the package import/ export as explained below (see Section 7.3.2)

Note: **Multiple selection** is possible in the explorer: press the Shift or Ctrl Key during your selection.

7.3.2 Import/Export package (Professional and Enterprise Edition only)

Import/export capabilities enables you to import and export packages between projects.

Exported packages must be self-contained. It means that all dependencies of the concepts in the package to be exported on concepts outside of it must be resolved. The tool will help you build self-contained packages by listing the documents and concepts for which there is a dependency relationship to solve.

To export a package:

1. Right-click a package and click **Export Package**. A control Wizard pops up. (see Figure 14).

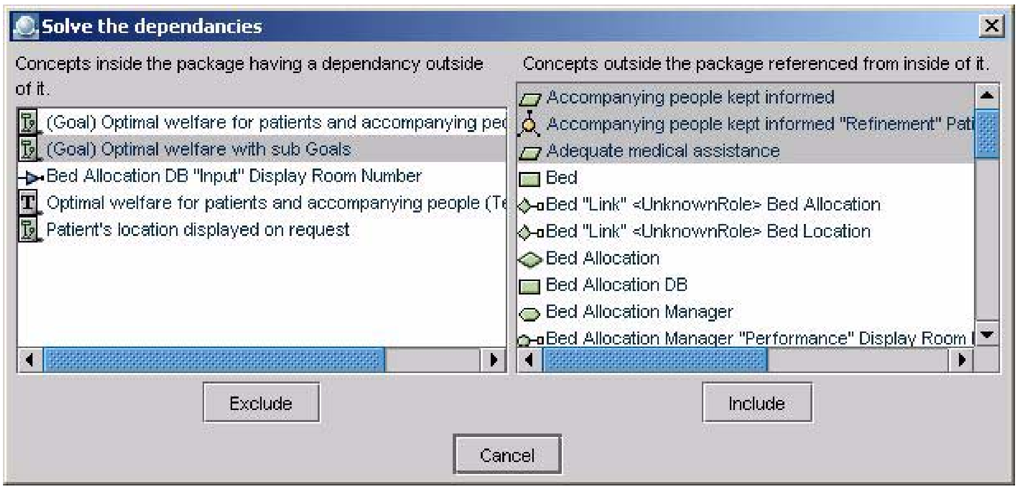
   The left-hand side column lists

   - all the relationships that are defined in the package to be exported and that link some concept defined outside the package. For instance (see Figure 14), it can be an input relationship for which the operation is defined in the package to be exported and the object is defined outside ("dangling" concept).
   - all the documents of the package to be exported that contain dangling concepts, that is, concepts defined in another package

   The right-hand side column lists

   - all the "dangling" concepts (including documents) that are defined outside the package to be exported and that are referred from inside the package. For instance, continuing the previous example, the object which is the input of the operation will appear in the right-hand side column.

   The left-hand side and the right-hand side columns are corresponding views: selecting one element in the left-hand side list will imply the selection of the corresponding concept(s) in the right-hand side list and conversely. For instance, in Figure 14, the selection of the diagram *optimal welfare with sub Goals* in the left-



Solve the dependancies

Concepts inside the package having a dependancy outside of it.

- (Goal) Optimal welfare for patients and accompanying pe
- (Goal) Optimal welfare with sub Goals
- Bed Allocation DB "Input" Display Room Number
- Optimal welfare for patients and accompanying people (Te
- Patient's location displayed on request

Concepts outside the package referenced from inside of it.

- Accompanying people kept informed
- Accompanying people kept informed "Refinement" Pati
- Adequate medical assistance
- Bed
- Bed "Link" <UnknownRole> Bed Allocation
- Bed "Link" <UnknownRole> Bed Location
- Bed Allocation
- Bed Allocation DB
- Bed Allocation Manager
- Bed Allocation Manager "Performance" Display Room I

FIGURE 14. Solve the dependencies

hand side column implies the selection of three concepts in the right-hand side column.

2. **Include** “dangling” concepts in the package to be exported.
   Or
   Exclude, from the package to be exported, relationships or documents that contain “dangling” concepts. In that case, choose the package where you wish to move out the concepts (relationships or documents).
3. When the lists are empty, a file dialog window pops up and prompts you to enter a filename. A file with the “opk” extension is created.

Note: When you Right-click a package and click **Bring Related Concepts**, the same Wizard pops up to enable you to include or exlude concepts.

To import a package:

1. Right-click a package and click **Import Package**.
2. Select the directory where your package file resides.
3. Double-click the file corresponding to your package.

7.4 Opening a document

Section 4.3 explains how to open an existing document by using a dialog box. A document can also be opened as follows:

1. In the explorer, find the document you want to open.
2. Double-click on it. Objectiver opens the document in a new window.

7.5 Adding items

The following operations suppose that the document in which you want to insert a concept is currently open.

7.5.1 Adding a concept or document reference to a diagram

1. Select the diagram window you want to edit.
2. In the explorer, locate the concept you want to insert.
3. For this step, you can either:

- **Alt-click** the concept in the explorer. Reference to this concept is added in the diagram.

Or

- **Drag** the concept in the explorer and **Drop** it where you want to add a reference to it in the diagram.

Or

- **Copy** the concept in the explorer by using the context menu (or Ctrl-C) and **Paste** a reference to it (via the context menu or Ctrl-V). If you select several concepts in the explorer, they all will be copied and pasted at the same time in the diagram.

and finally,

4. Move the added concept reference to the position you want. If you want to move the reference outside the diagram area shown on screen, drag the reference toward the target area to cause the window to scroll in that direction.

Note: When you add a concept reference to a diagram, all relationships that connect this concept to other concepts already present in the diagram are shown.

7.5.2 Adding a concept or document reference to a text document

1. Place the cursor in the text document at the position where you want to add a concept reference.
2. In the explorer, locate the concept for which you want to insert a reference. Select one or several concepts in the explorer.
3. You now have the choice between the following options:

- **Alt-click** the concept in the explorer. Objectiver then adds a reference to the selected concept in the text document at the cursor position.

Or

- **Drag** the concept in the explorer and **Drop** it where you want to add the reference in the text.

Or

- **Copy** the concept in the explorer by using the context menu (or Ctrl-C) and **Paste** a reference to it (via the context menu or Ctrl-V) where you want to add the reference in the text.

7.6 Deleting items

7.6.1 Deleting a concept from the model

In order to delete one or more concept(s) from the model, proceed as follows:

1. In the explorer select the concept(s) you want to delete. Alternatively, **Alt-click** the concept in a document (text or diagram). The (last) selected concept is displayed in the property editor.
2. Choose **Edit>Delete** or Press Ctrl+Shift-Delete and then confirm the deletion.

OR

1. In the explorer, Right-click the concept(s) you want to delete to open the context menu.
2. In the popup window, select **Delete** and then confirm the deletion.

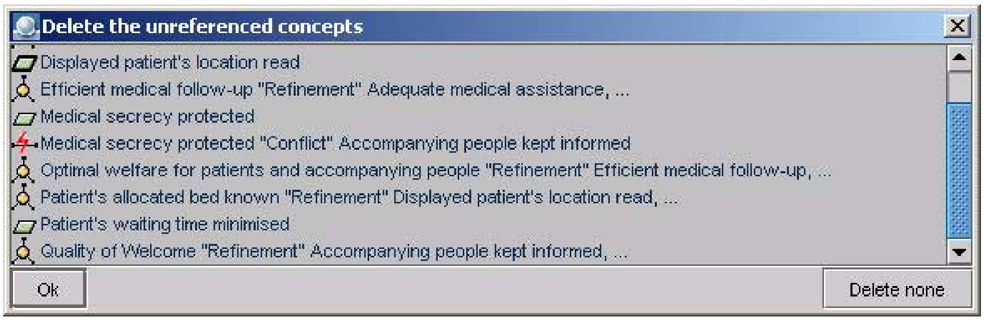
Important notes:

- All references to the concept in text documents are replaced by a string: [Deleted: <Concept-name>]. It allows you to rework the text easily.
- When a concept is deleted, all attached binary relationships are deleted.

7.6.2 Deleting a document

In order to delete a document from the model, proceed as follows:

1. In the explorer, select the document you want to delete.
2. Choose **Edit>Delete** or Press Ctrl+Shift-Delete and then confirm the deletion.
3. Objectiver asks whether you want to delete all the concepts that are no longer referenced, that is, the concepts which do no longer appear in any diagram or text. Figure 15 shows a screenshot of this dialog. By default all items are selected for deletion. You can deselect items by clicking their name while holding the Ctrl key down.


Delete the unreferenced concepts

Displayed patient's location read
Efficient medical follow-up "Refinement" Adequate medical assistance, ...
Medical secrecy protected
Medical secrecy protected "Conflict" Accompanying people kept informed
Optimal welfare for patients and accompanying people "Refinement" Efficient medical follow-up, ...
Patient's allocated bed known "Refinement" Displayed patient's location read, ...
Patient's waiting time minimised
Quality of Welcome "Refinement" Accompanying people kept informed, ...

FIGURE 15. Delete the unreferenced concepts

7.7 Renaming documents and concepts

In order to rename documents and concepts, proceed as follows:

1. Display the property editor (see Section 9.2).
2. Select the document or concept in the explorer (see Section 7.2); the attribute names and values for the document or concept are displayed in the property editor.
3. Modify the value of the *Name* in the Properties tab by typing the new document or concept name in the corresponding field.

Or more easily,

1. In the explorer, Right-click on the concept you want to rename to open the context menu.
2. In the popup window, select **Rename**.
3. Type the new document or concept name.

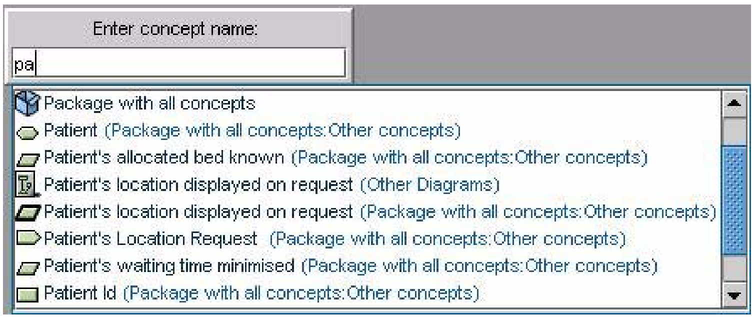
7.8 Modifying the type of a concept

In order to modify the type of a concept, proceed as follows:

1. Right-click a concept in the explorer or a document and choose **Modify type** from the context menu.
2. In the dialog that pops up, select the new concept type from the list.
3. If any attribute or relationship is about to be deleted while changing the concept type, Objectiver then prompts you to confirm your request before changing the concept type.

7.9 Locating a concept in the Concept Index

1. Choose **Edit>Find a concept**. A window pops up (see Figure 16).
2. Enter the first characters of the concept name. The list is restricted to the concepts beginning with the string you have typed.
3. Select a concept and press **Return**.



Enter concept name:

pa

- Package with all concepts
- Patient (Package with all concepts:Other concepts)
- Patient's allocated bed known (Package with all concepts:Other concepts)
- Patient's location displayed on request (Other Diagrams)
- Patient's location displayed on request (Package with all concepts:Other concepts)
- Patient's Location Request (Package with all concepts:Other concepts)
- Patient's waiting time minimised (Package with all concepts:Other concepts)
- Patient Id (Package with all concepts:Other concepts)

FIGURE 16. Locate a concept

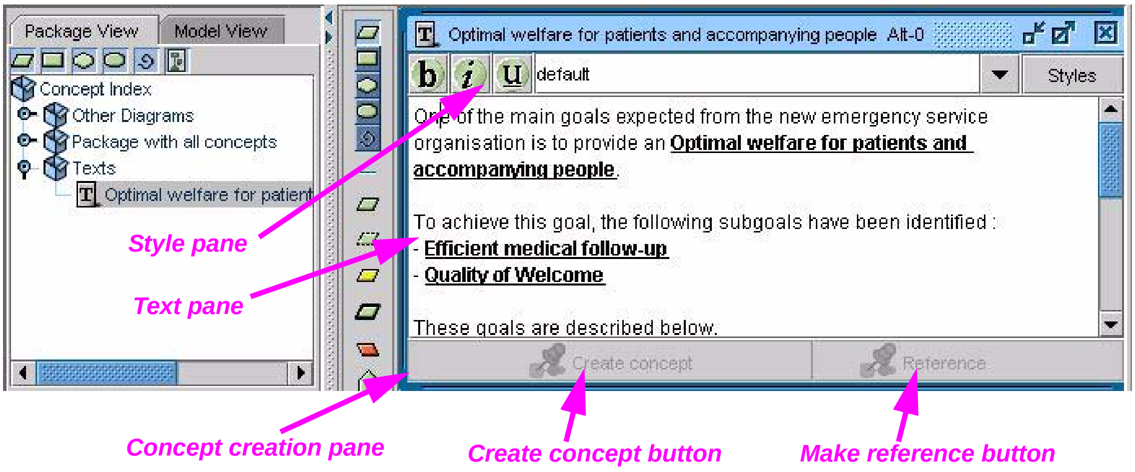
8.0 Text Edition

8.1 About Texts

Objectiver text documents (TXT documents) are typically used to record short notes, to explain simple diagrams, to keep track of interviews or to record other textual materials used during a requirements analysis project provided that those textual descriptions do not require a sophisticated layout. The Objectiver text editor (TXT editor) makes it possible to edit such documents.

8.2 Basic Operations

Refer to Section 4.0 to learn how to create, open, clone, delete or rename a diagram. Figure 17 shows you a new TXT document.



One of the main goals expected from the new emergency service organisation is to provide an **Optimal welfare for patients and accompanying people**.

To achieve this goal, the following subgoals have been identified :
- **Efficient medical follow-up**
- **Quality of Welcome**

These goals are described below.

FIGURE 17. A text document

A TXT editor consists of three panes:

- a **style pane** showing all standard and user-defined character styles that can be applied to sequences of characters.
- a **text pane** where text and references to concepts can be entered freely.
- a **concept creation pane** allowing you to create concepts and references from the text editor.

8.3 Working with character styles

There are three standard, predefined text styles: **bold**, *italic*, and underlined. You can also define your own styles and apply them to any sequence of characters, to paragraphs or to references to concepts or documents.

Figure 18 shows the character style pane. The following sections describe how to create a new character style and how to apply it to a paragraph in a text pane or to references to concepts or documents.

Note: In this release, character styles are local to your model You cannot reuse the character styles you define in another model.

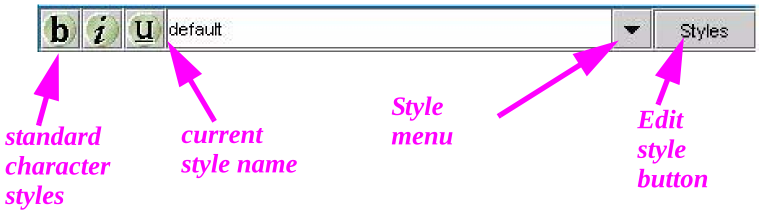
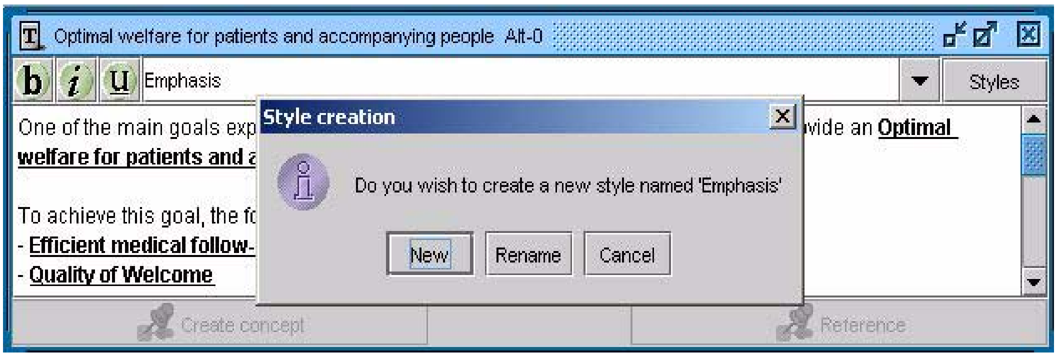


FIGURE 18. Character style pane

8.3.1 Defining a new character style

1. In the style name field, enter a new name for the character style you want to create, for instance `Emphasis`.
2. Press **Return**. Objectiver asks you whether you want to create a new style (see Figure 19). Click **New**.


Optimal welfare for patients and accompanying people Alt-0

Style creation

Do you wish to create a new style named 'Emphasis'

b
i
u
Emphasis
Styles
One of the main goals exp
welfare for patients and a
vide an Optimal
To achieve this goal, the fo
- Efficient medical follow-
- Quality of Welcome
New
Rename
Cancel
Create concept
Reference

FIGURE 19. Style creation

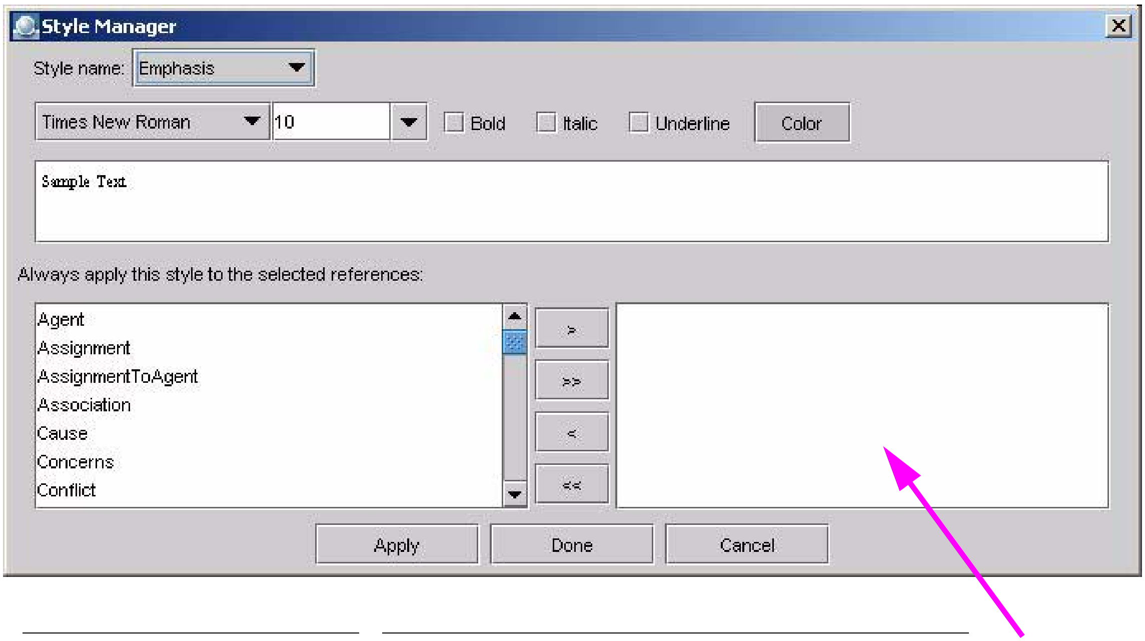
3. Click **Styles**. Objectiver then opens the style manager (see Figure 20).
4. Enter values for style parameters, including: font, size, variation (bold, italic, underline), color.
5. Click **Apply**.

8.3.2 Applying a style to a paragraph or to a sequence of characters

1. Select the paragraph to which you want to apply a character style by clicking anywhere within the paragraph. You can also apply a style to a sequence of characters by selecting them.
2. From the style combo box, select the style you want to apply to the paragraph. Objectiver then applies it to the current paragraph.

8.3.3 Defining styles for concept and document references

Character styles can be defined for concept references. For instance, you can apply this feature to your text documents so that references to goals appear in blue and



Style Manager

Style name: Emphasis

Times New Roman | 10 | ☐ Bold | ☐ Italic | ☐ Underline

Sample Text

Always apply this style to the selected references:

- Agent
- Assignment
- AssignmentToAgent
- Association
- Cause
- Concerns
- Conflict

Selection List

FIGURE 20. Style manager

references to actions appear in red. Concepts having the same type must share the same style. Once a style is associated with a concept type, it will automatically be applied to all references to concepts of that type in all documents. Proceed as follows to define a character style for concept references:

1. In the current character style text field, enter a new name for the character style you want to create. We recommend you give the same name to the style as to the concept type, for instance `Goal`.
2. Press **Return**. Objectiver then asks you whether you want to create a new style (Figure 19). Click **New**.
3. Click **Styles**. Objectiver then opens the style manager (Figure 20).
4. Enter values to style parameters, including: font, size, variation (bold, italic, underline), color.
5. Select the concept type you want to associate a style with, e.g., `Goal`.
6. Transfer your selection in the selection list on the right by clicking on '**>**'.
7. Click **Apply**.

Note: To define a character style for document references (instead of concepts), select `Diagram` or `Text` at step 6.

8.4 Creating concepts and adding references to them

You can create concepts directly from the text editor. Proceed as follows:

1. Select the concept name in the text pane (type it before if needed).
2. There are two different cases:

- If the selection does not match any concept name, then by clicking a concept type in the *palette*, Objectiver creates a new concept, named after the selected text and inserts a reference to this concept in the text.

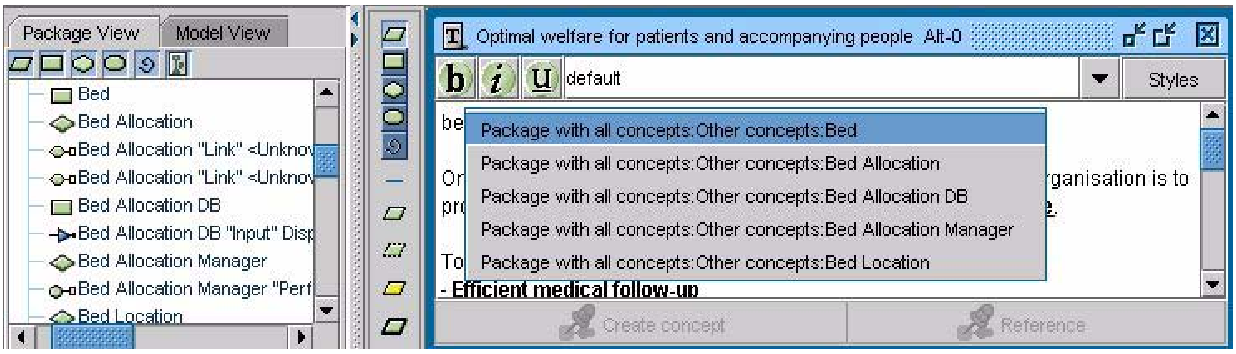
If an item was previously selected in the palette before the text selection was made, you can click the **New concept** button that appears on the bottom of the text editor. The button name actually reflects the currently selected palette item.

- If the text selection matches an existing concept name, you can click the **Reference** button to replace the text with a reference to the existing concept.

Auto-completion:

You can use the auto-completion mode to easily add an existing concept to a text document (see **Figure 21**).

1. Type the first letters of the concept to which you want to refer.
2. Press Ctrl - Space bar.
3. Choose the concept to insert from the list that pops up.



- Bed
- Bed Allocation
- Bed Allocation "Link" <Unknov
- Bed Allocation "Link" <Unknov
- Bed Allocation DB
- Bed Allocation DB "Input" Disp
- Bed Allocation Manager
- Bed Allocation Manager "Perf
- Bed Location

be
Or
pro
To
- Efficient medical follow-up

Package with all concepts:Other concepts:Bed
Package with all concepts:Other concepts:Bed Allocation
Package with all concepts:Other concepts:Bed Allocation DB
Package with all concepts:Other concepts:Bed Allocation Manager
Package with all concepts:Other concepts:Bed Location

rganisation is to
e.

FIGURE 21. Auto-completion inside a text document

Note: see also Section 7.5.2 for adding references to a document.

8.5 Finding missing concept reference

From the context menu choose **Find missing concept references** to search in the text for words that match existing concept names and propose to replace them by references to the related concepts.

8.6 Hypertext and Picture Link

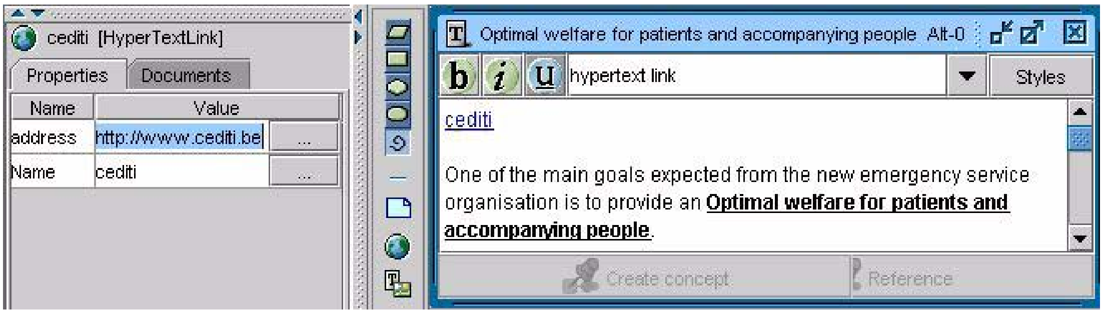
Creating hypertext or picture links in texts is done in the same way as concept creation (refer to Section 8.4). In the concept palette you can select “HypertextLink” or “PictureLink” from under the “Other Concepts” Palette.

Then, click the concept in order to fill in the address property in the property editor with one of the following:

- a web link (URL) (for example http://www.objectiver.com/adocument.doc),
- an absolute file path (for example E:\Objectiver\adocument.doc or \\asterix\projets\Objectiver\adocument.doc),

- a relative file path (for example ../adocument.doc).

Figure 22 shows a text and a property editor with an hypertext link.



cediti [HyperTextLink]

Name	Value
address	http://www.cediti.be
Name	cediti

Optimal welfare for patients and accompanying people Alt-0

hypertext link

cediti

One of the main goals expected from the new emergency service organisation is to provide an **Optimal welfare for patients and accompanying people**.

FIGURE 22. Hypertext Link

Notes:

1. The table below shows how pictures are inserted inside generated documents.

TABLE 3. Picture links inside generated documents

Objectiver document	Web doc (see Section 12.0)	RTF/PDF doc (see Section 13.0)
URL	The picture is displayed when the page is loaded. The browser retrieves the picture by following the URL.	The picture is inserted in the doc at generation time. It can require the specification of an http proxy to allow the tool to retrieve the picture.
Absolute path	The picture is displayed when the page is loaded. The browser retrieves the picture by following the path.	The picture is inserted in the doc at generation time.
Relative path	The picture is displayed when the page is loaded. The browser retrieves the picture from the folders containing the web doc (Objectiver automatically creates a copy of the file to make the distribution easier).	The picture is inserted in the doc at generation time.

2. The table below shows how documents are referenced inside generated documents.

TABLE 4. Hypertext links inside generated documents

Objectiver document	Web doc (see Section 12.0)	RTF/PDF doc (see Section 13.0)
URL	Hyperlink inserted in the page. It points to the URL.	Hyperlink inserted in the doc. It points to the URL.

TABLE 4. Hypertext links inside generated documents

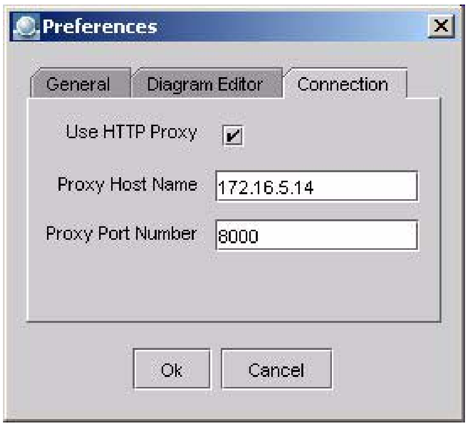
Objectiver document	Web doc (see Section 12.0)	RTF/PDF doc (see Section 13.0)
Absolute path	Hyperlink inserted in the page. It points to the file referred by the absolute path.	Hyperlink inserted in the doc. It points to the file referred by the absolute path.
Relative path (with the Objectiver file)	Hyperlink inserted in the page. The address is relative to the folder in the generated web doc.	Hyperlink inserted in the doc. The address is relative to the report file (the tool does not copy the referred file).

When relative paths are used, it is up to the user to structure his/her workspace so that the hypertext links point to the right files. The following structure makes the relative paths always valid: the model, the reports and the WebDoc are put in separate directories all belonging to the same directory.

```
MyProject
        Model
              MyProject.ob
              MyProject.ob_bak
              MyProject--01--.ob_bak
        Web 1.0
              index.html
              ...
        Web 1.1
              index.html
              ...
        Reports
              MyProject.rtf
              MyProject.pdf
        References
              ReferencedDocument1.doc
              ReferencedDocument2.doc
```

Note about Proxy server: If you use a Proxy server, you will have to fill in the connection preferences in order to include images coming from the web in your reports. Proceed as follows:

1. Choose **Tools>Preferences**.
2. Select the Connection tab. (see Figure 11).
3. Fill in the address and port fields.



Use HTTP Proxy ☑

Proxy Host Name 172.16.5.14

Proxy Port Number 8000

FIGURE 23. Preferences concerning the connections

9.0 Concept Property Edition

9.1 About the Property Editor

The property editor allows you:

- to describe concepts by specifying values for predefined **Properties**.
- to specify the names and types of user-defined **Attributes** that all concept instances will have.
- to visualize the concept neighborhood, that is, the concepts to which the selected concept is related.
- to visualize the documents in which the selected concept appears.

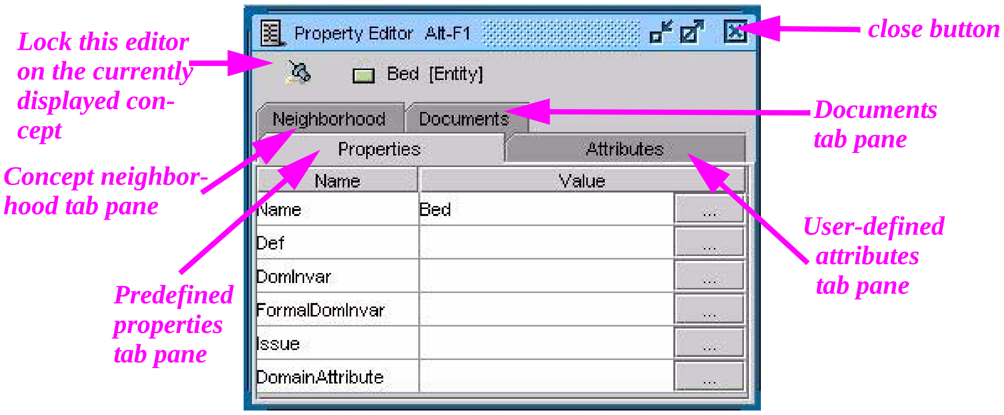
These functionalities are explained in sections 9.4 to 9.8. Section 9.2 describes how the property editor can be activated and closed. Section 9.3 explains how to display concept properties.

Note: When you select a concept for deletion from the model (see Section 7.6), Objectiver displays its attributes in the property editor so that you can check whether it's the right concept to delete.

9.2 Displaying the Property Editor

The Property Editor is located under the explorer (see Figure 1).

You can also activate floating property editors. Choose **Tools>Property Editor.** Figure 24 shows a screenshot of the property editor once activated.



Name	Value
Name	Bed
Def	
DomInvar	
FormalDomInvar	
Issue	
DomainAttribute	

Lock this editor on the currently displayed concept

Concept neighborhood tab pane

Predefined properties tab pane

close button

Documents tab pane

User-defined attributes tab pane

FIGURE 24. Floating Property Editor for an entity

The *lock* icon (shown in Figure 24) prevents the property editor from refreshing its window when you select another concept.

9.3 Displaying a concept properties

There are two ways for displaying a concept properties in the property editor: from the explorer or from any document window.

9.3.1 From the explorer

In the explorer, select the concept for which you want to edit the properties. Objectiver shows the current properties for the selected concept in the property editor. Figure 24 shows an example of entity properties.

9.3.2 From any document (text or diagram) window

In a document, click a concept to display its properties in the property editor.

Notes:

- If you **Alt-click** a concept, the concept is also located in the concept index.
- When you select a concept, all the property editors you did not lock (see Figure 24) display the properties of this concept.

9.4 Predefined properties

Predefined properties associated with each concept type are described in the following table.

TABLE 5. Predefined concept properties

Concept / Relationship type	Property name
Object[a]	Name, Def, DomInvar, FormalDomInvar, Issue, DomainAttribute
Agent	same attributes as Object plus: Load, Category
Event	same attributes as Object plus: Frequency
DomProp	Name, Def, Issue, Category, FormalDef
Objective[b]	same attributes as DomProp plus: Pattern, Priority
Obstacle	same attributes as DomProp plus: Likelihood; Criticality
Operation	Name, Def, DomPre, DomPost, FormalDomPre, FormalDomPost, Issue
Cause	Delay
Link	Multiplicity, Role, RoleOrder
Resolution	ResolutionType
Conflict	BoundaryCondition
Binary Association	Type, Directed, Label, Client Role, Client Multiplicity, Supplier Role Supplier Multiplicity, Ordered
Control, Monitoring	WhichAtt
OperationalizationActionSon	FormalReqPost, FormalReqTrig, FormalReqPre, ReqPost, ReqTrig, ReqTrig, ReqPre
Refinement	Pattern, AltName, Complete

a. Object is a generic name for Agents, Entities, Events and N-Ary Associations. Those concept types inherit the properties defined on Objects.

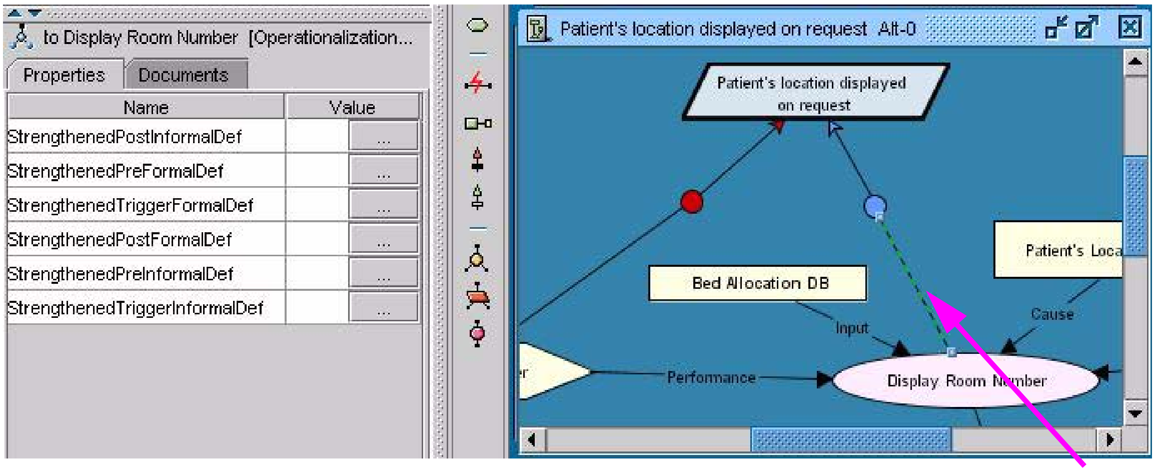
b. Objective is a generic name for Goals, Requirements, Expectations, SoftGoals.

9.4.1 Setting values for concept properties

In the property tab pane, enter a value in the Value field. If the value type is String, by clicking on “**...**” at the right of the value field, you activate a popup window that will allow you to input text in a more convenient way.

9.4.2 Setting strengthened conditions for actions

To display the properties for Strengthened conditions in the Property Editor, click the edge, part of an Operationalization relationship, between the action and the operationalization circle.



to Display Room Number [Operationalization...

Properties | Documents

Name	Value
StrengthenedPostInformalDef	...
StrengthenedPreFormalDef	...
StrengthenedTriggerFormalDef	...
StrengthenedPostFormalDef	...
StrengthenedPreInformalDef	...
StrengthenedTriggerInformalDef	...

Patient's location displayed on request Alt-0
Patient's location displayed on request
Bed Allocation DB
Patient's Loca
Input
Cause
Performance
Display Room Number

click this edge to display strengthened conditions put on the operationalization

FIGURE 25. Strengthened conditions

9.5 Working with user-defined attributes

Attributes can only be added to Objects.[1]

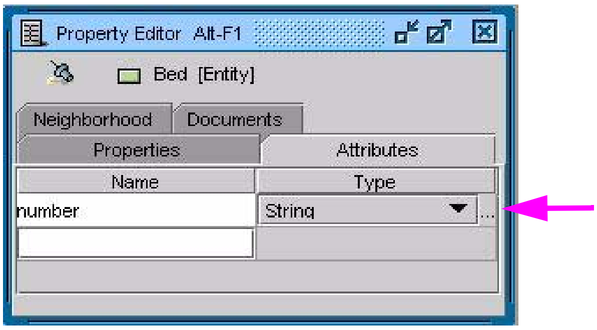
To create or delete a user-defined attribute, display the attributes tab pane by clicking on it (Figure 26).

9.5.1 Creating an attribute

1. Click in the *Name* field (Figure 26).
2. Specify the attribute Name (e.g., number).
3. Press **Return** to record your changes. A new attribute entry appears in the form, its default type is `String`.
4. If needed, click in the *Type* field and change the attribute type. A combo box enables you to select a type (Figure 26). You can choose among different predefined types: Boolean, Date, Formula, Integer, Longstring, Path, Real and String or the type can be Undefined. You can also create your own type by using the attribute type editor (see Section 9.6).

Note: Attributes for Entities are displayed and editable in diagrams.

[1] Object is a generic name for Agents, Entities, Events and N-Ary Associations.



Name	Type
number	String

Create a new attribute type

FIGURE 26. Instance attribute (new attribute)

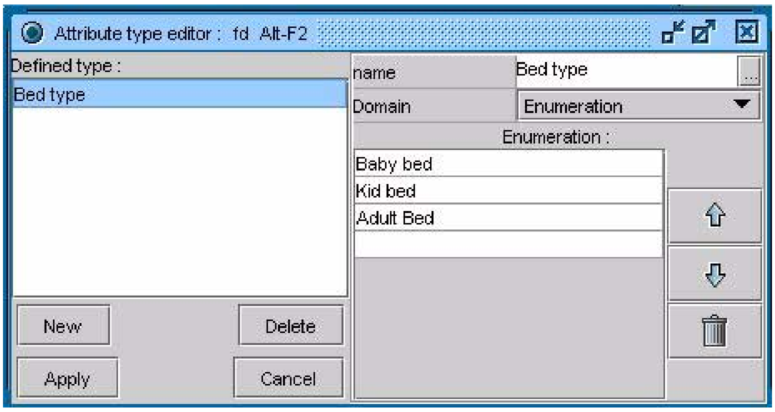
9.5.2 Deleting an attribute

1. Right-click on the attribute you want to delete.
2. From the context menu choose **Delete attribute**.

9.6 Attribute type editor

9.6.1 Displaying the attribute type editor

Choose **Tools>Attribute type Editor** to open it (Figure 27) or click the "**...**" button at the right of the *Type* field (see Figure 26).



Defined type : Bed type

name: Bed type

Domain: Enumeration

Enumeration :

- Baby bed
- Kid bed
- Adult Bed

FIGURE 27. Attribute type editor

9.6.2 Creating a new attribute type

1. Click **New**.
2. Objectiver prompts you to give a name attribute type.
   An attribute type is created and appears in the explorer.
3. Specify the `Domain` (Integer, Real, String or Enumeration).
4. If needed, give the values of the enumeration.
5. Click **Apply** and close the attribute type editor.

From now on, the type you have defined is added to the list of types you can assign to an attribute (see Section 9.5.1).

9.7 Displaying the concept neighborhood

Choose a relationship type from the combo box menu to display the concepts participating to these relationships with the selected concept. Figure 28 shows the concepts related to the ‘Bed’ concept via the ‘Linked To’ relationship.

Note: The concept names can be modified from here.

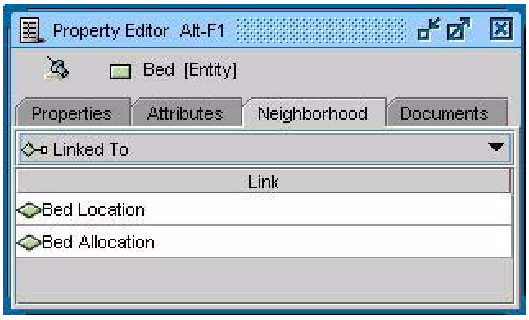


FIGURE 28. ‘Linked To’ relationship

9.8 Displaying the documents in which a concept appears

Display the documents in which the selected concept appears by clicking on the document tab pane.
Note: You can open any of these documents by clicking on the **Open** button.

10.0 Analysing the model and solving modeling issues (Pro & Enterprise Edition only)

10.1 Query & Check Tool

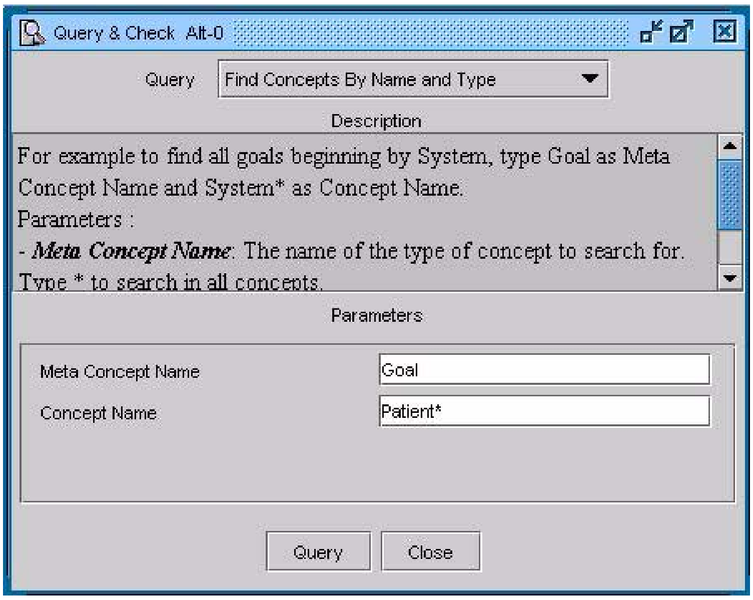
10.1.1 About the query & Check Tool

The tool provides a series of predefined queries you can run on your model.

10.1.2 Opening the Query & Check window

Choose **Tools>Query & Check**.

A screenshot of the Query & Check window is shown in Figure 29.



Query: Find Concepts By Name and Type

For example to find all goals beginning by System, type Goal as Meta Concept Name and System* as Concept Name.
Parameters :
- ***Meta Concept Name***: The name of the type of concept to search for. Type * to search in all concepts.

Meta Concept Name: Goal

Concept Name: Patient*

FIGURE 29. Query Editor

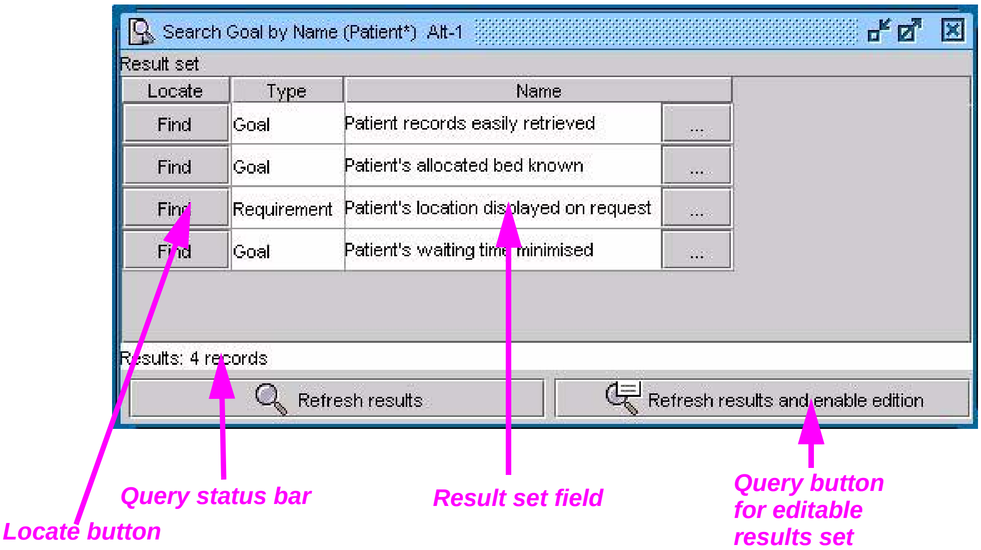
10.1.3 How to perform queries

1. Opening the Query & Check window.
2. Choose a query in the combo box.
3. If needed, give a value to the parameters.
4. Click **Query**. Objectiver returns the result set associated with your query in the *Result set* field. The number of results is displayed in the query status bar (see Figure 30).

Notes:

1. Click on a column header to sort the result set.
2. If you want to save the results, copy and paste it in a document (text document in Objectiver or Excel sheet for example).
3. Two wildcards can be used: ? matches any character and * matches any chain of characters.

4. Click on the Locate button to find the concept in the explorer tool.



Search Goal by Name (Patient*) Alt-1

Result set

Locate	Type	Name	
Find	Goal	Patient records easily retrieved	...
Find	Goal	Patient's allocated bed known	...
Find	Requirement	Patient's location displayed on request	...
Find	Goal	Patient's waiting time minimised	...

Results: 4 records

Query status bar

Result set field

Query button for editable results set

Locate button

FIGURE 30. An example of query with editable results set

10.1.4 Editing the result of a query

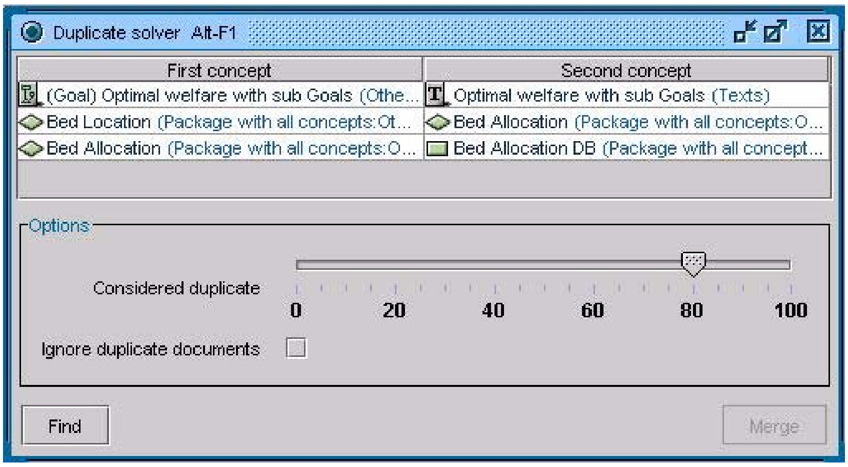
1. Perform your query as explained in Section 10.1.3.
2. Enable edition of the result set by clicking **Refresh results and enable edition**.
3. Click the '**...**' buttons to edit the corresponding value.

10.2 Solving duplicates

10.2.1 Finding duplicate concepts

To find concepts with name similarity, proceed as follows:

1. Choose **Tools>Duplicate solver** to open the solver (see Figure 31).
2. Adjust the name similarity threshold by moving the horizontal slider: a value of 100 means exact matching; decreasing the slider value weakens the matching criterion. It is advised to start looking for duplicates by providing a high slider value (100 or close to 100), then to decrease the value progressively. The lower the slider value is, the more Objectiver will spend time to retrieve potential duplicates.
3. Click **Find** to display the result.



First concept	Second concept
(Goal) Optimal welfare with sub Goals (Othe...	Optimal welfare with sub Goals (Texts)
Bed Location (Package with all concepts:Ot...	Bed Allocation (Package with all concepts:O...
Bed Allocation (Package with all concepts:O...	Bed Allocation DB (Package with all concept...

Considered duplicate: 80

Ignore duplicate documents □

FIGURE 31. Duplicate solver

PART C Working with Open Office

11.0 Editing texts with Open Office NeW

11.1 About Open Office Texts

Section 8.0 has described how to use the **Objectiver Text** (TXT) editor to explain diagrams and to write simple text documents. The TXT editor has the great benefit of being light and easy to use but has the major drawback of limited formatting capabilities as, for instance, in the following situations:

- For some specific, complex diagram explanations, you may want to insert tables, diagrams, images, etc. in the explanations.
- Some requirements analyses need to be based on some sophisticated input document to which you want to trace your model.

The TXT component does not provide any support or very poor support on those matters! That's precisely why Objectiver has been seamlessly interfaced with the **Open Office Text Editor** (**OO Text** editor).

The OO Text editor allows you to do a.o. the following:

- write simple texts as before,
- insert **images** and **objects**,
- compose **drawings**,
- define **tables**,
- introduce **formulas** in elementary spreadsheets.

Additionally, the editor contains a **spelling checker** function that can be configured to automatically check your text on the fly.

Within Objectiver, you can edit several OO Text documents simultaneously. Each OO Text document will be edited in a separated window. Those windows will work independently from the main Objectiver window. They are not pinned to the Objectiver document frame. That allows you to use several screens for your best convenience: for instance, you can display an Objectiver diagram in one screen and its explanation in another screen.

This section does not intend to explain in detail how the OO Text editor works. Please refer to the on-line Open Office help coming with the editor to learn how to use the advanced editor features.

11.2 Create, open, clone, delete, and rename an OO Text document

As OO Text documents are just Objectiver documents, you can create, open, clone, delete, and rename an OO Text document as any other Objectiver document (simple text, diagram, …). Figure 32 shows how to initiate the creation of a new document. Please refer to Section 4.0 to learn how to do those basic operations.

Note. Creating or opening the first OO Text document during a session will require some delay, rather longer than for the next creation and opening operations. The reason is that an Open Office server needs to be started and connected to Objectiver.

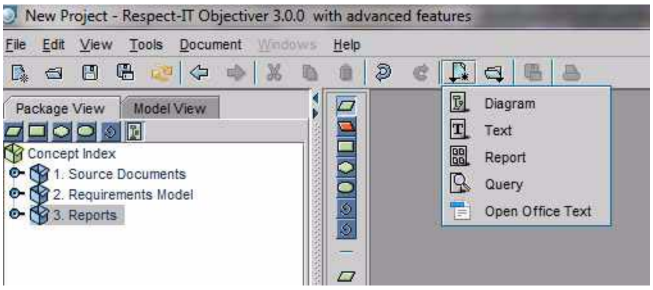
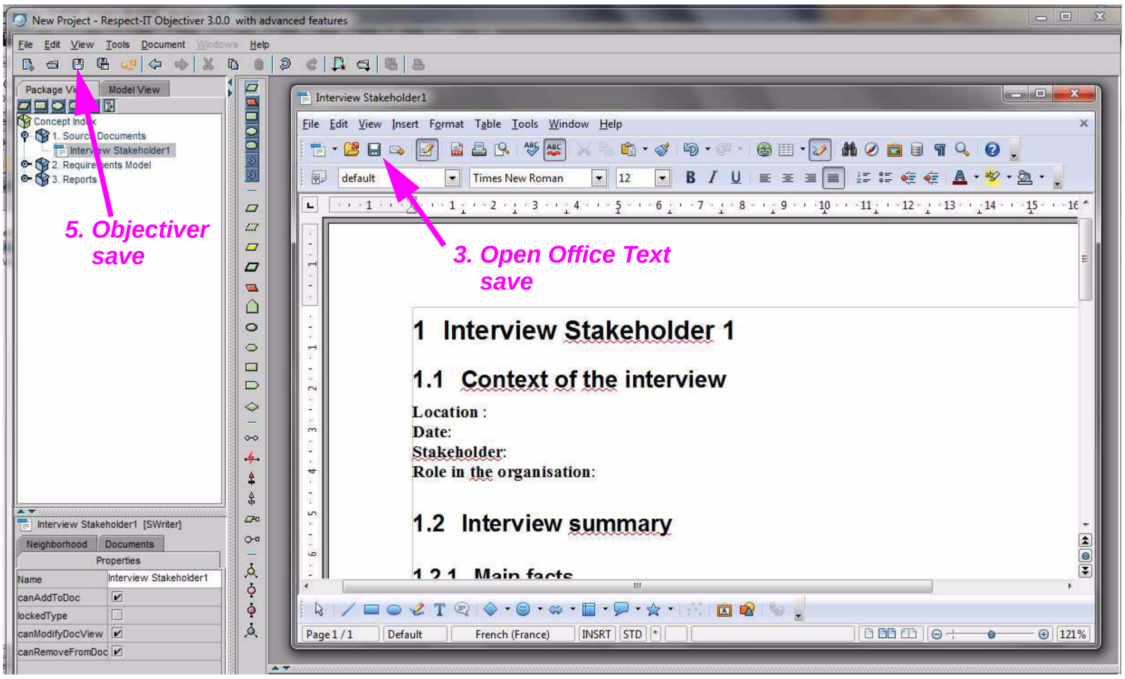


FIGURE 32. Creating an Open Office Text document

11.3 Save an OO Text document

When you choose **File>Save Project**, Objectiver saves all the concepts in your model and all documents including OO Text documents in a unique .ob file on disk. (see Section 3.4).

The OO Text windows which contain your OO Text documents have also a *Save* function (named *OO Text Save* in the following). This *OO Text Save* function must be used to tell Objectiver that the current OO Text document you are editing is ready to be saved in the .ob file. Please note that the *OO Text Save* function has no impact on the .ob file. So a typical (and recommended) session would be like the following one:


5. Objectiver save

3. Open Office Text save

1 Interview Stakeholder 1

1.1 Context of the interview

Location :
Date:
Stakeholder:
Role in the organisation:

1.2 Interview summary

1.2.1 Main facts

Name	Interview Stakeholder1
canAddToDoc	☑
lockedType	□
canModifyDocView	☑
canRemoveFromDoc	☑

FIGURE 33. Saving an OO Text document

1. Create a new OO Text document or edit an existing one
2. Work on the OO Text document as usual

3. Use the *OO Text Save* function to tell Objectiver you have finished with the edition of your OO Text document (see Figure 33). You can quit the OO Text editor if you wish.
4. Continue working on your model or on other documents.
5. When you are happy with your work, click the *Objectiver Save* Button to save all your work on disk in a .ob file (including the OO Text files you have pre-saved at step 3).

If you want to discard the changes you have done on a OO Text document, just quit the OO Text editor without using the *OO Text Save* function. The OO Text editor will prompt you to confirm that you want to quit without saving your changes.

If you forget to save your OO Text documents and just use the Objectiver Save function, Objectiver will check whether you still have unsaved OO Text sessions. If yes, Objectiver will prompt you to save them. If you discard your changes, they will not be saved in the .ob file.

Note. In some OS configurations, the pop-up window inviting you to save or discard OO Text sessions might not come up on top of the other displayed windows. If you are saving an Objectiver model with some open OO Text sessions and have the feeling that the tool is blocked, just move or iconize the top windows to check whether the pop-up is not hidden behind.

Note. You can store large OO Text documents in your model with no consequence at start time. When Objectiver opens a .ob file, it parses your model and loads it into memory. OO Text documents are not parsed by the tool at loading time. So large OO Text documents will just inflate the size of your .ob file.

11.4 Importing and exporting an OO Text document

OO Text documents can be imported from outside Objectiver to an Objectiver model and conversely can be exported from an Objectiver model to an Open Office file.

Import. To import an OO Text document from outside Objectiver, proceed as follows:

1. In the Objectiver explorer, select the folder in which you want to import the OO Text file
2. Right-click the folder name and select Import OO Text file from the pop-up menu (see Figure 34)
3. With the File interaction dialog, select the OO Text file you want to import (a file with a .OO Text suffix) and click Open. Objectiver then copies the file into a new Objectiver OO Text document.

Note. The Import function does not delete the source file on disk. It physically copies the OO file into the .ob file.

Export. To export an OO Text document outside Objectiver, proceed as follows:

1. In the Objectiver explorer, select the OO Text document you want to export
2. Right-click on it and select Export…(see Figure 35)
3. Give a name for the .OO Text file you want to save your OO Text document and click *Save*. Objectiver then copies your OO Text document into a new .OO Text file named according to your input.

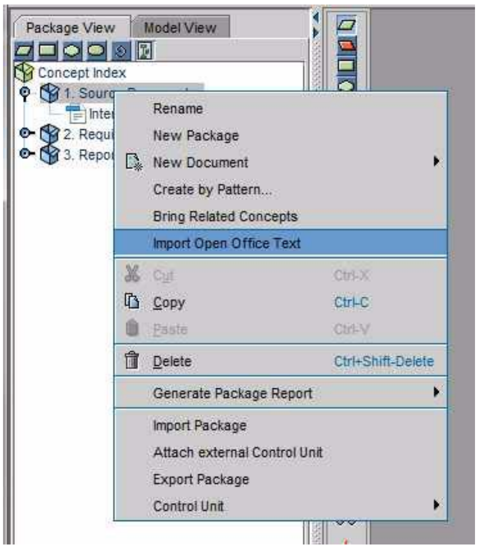


FIGURE 34. Importing an OO Text document

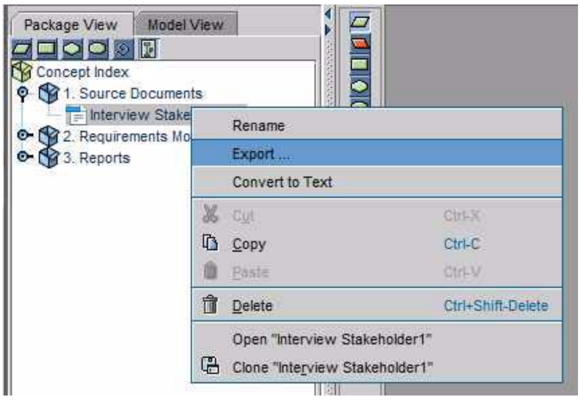


FIGURE 35. Exporting an OO Text document

Note. The export function does not remove the OO Text document from your model.

11.5 Inserting a reference into an OO Text document

You can insert a reference to an existing concept of your model into an OO Text document quite easily:

1. Put the cursor to the location in the OO Text document where you want to insert a reference
2. Right-click and select Objectiver>Insert Concept Reference…(see Figure 36)
3. A dialog 'Select a concept' pops up. Select the concept you want to insert in the list and click Add. Alternatively, to locate a concept quicker in a large list, type the name of the concept in the input field or use the auto-completion mode by

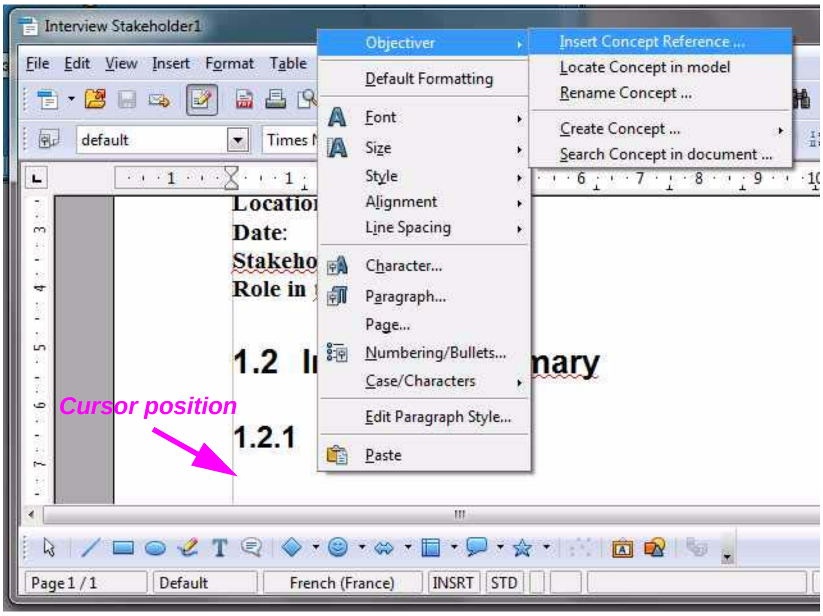
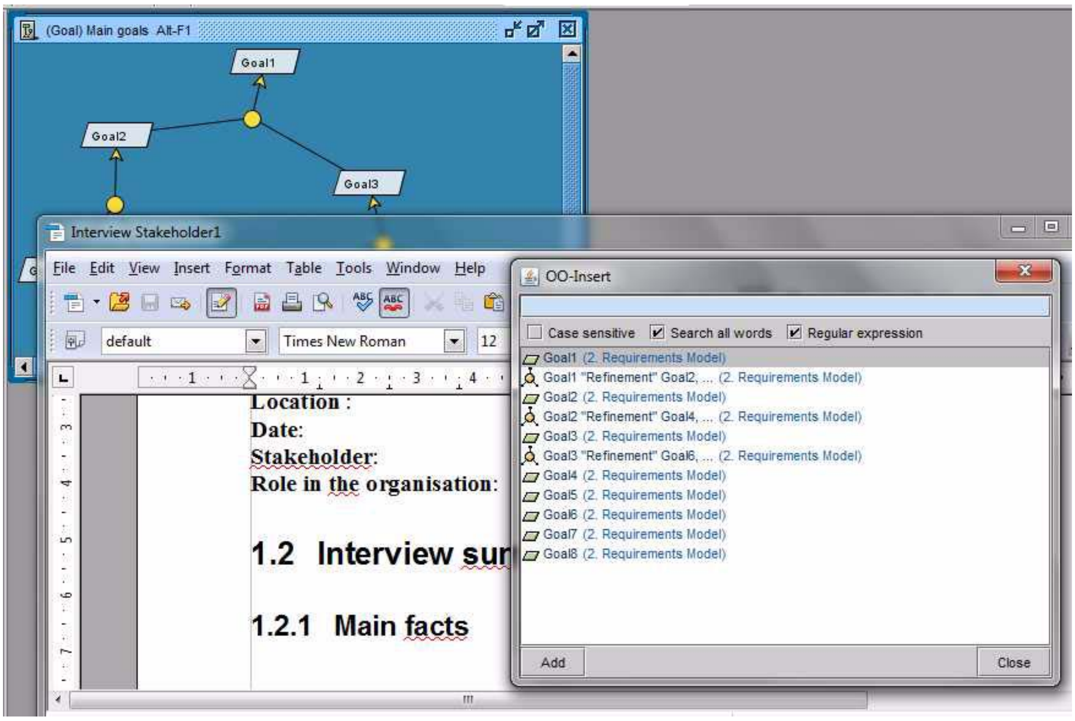


FIGURE 36. Inserting a concept reference

typing only the first characters of the concept name followed by a space character. (see Figure 37)



☐ Case sensitive ☑ Search all words ☑ Regular expression

Goal1 (2. Requirements Model)
Goal1 "Refinement" Goal2, ... (2. Requirements Model)
Goal2 (2. Requirements Model)
Goal2 "Refinement" Goal4, ... (2. Requirements Model)
Goal3 (2. Requirements Model)
Goal3 "Refinement" Goal6, ... (2. Requirements Model)
Goal4 (2. Requirements Model)
Goal5 (2. Requirements Model)
Goal6 (2. Requirements Model)
Goal7 (2. Requirements Model)
Goal8 (2. Requirements Model)

Location :
Date:
Stakeholder:
Role in the organisation:

1.2 Interview sur

1.2.1 Main facts

FIGURE 37. Selecting the concept, you want to refer to

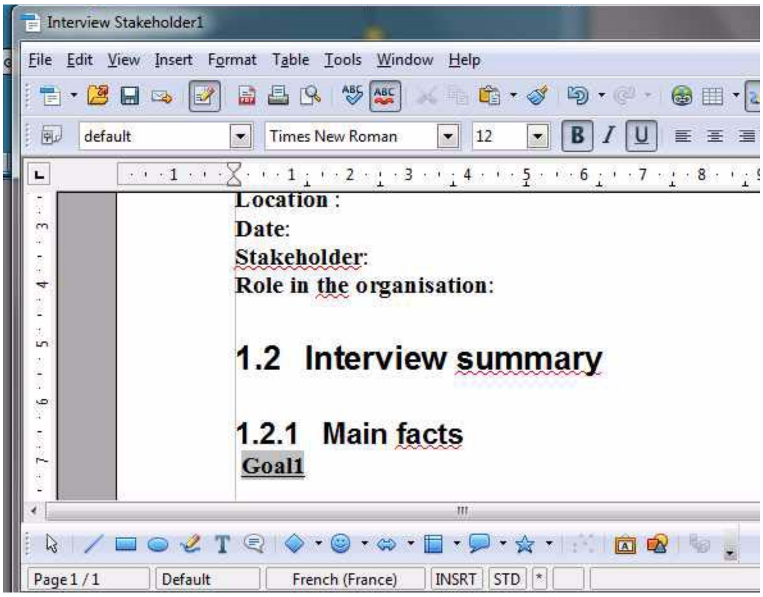
Observe how Objectiver inserts a reference in the OO Text document (see). In the Property editor for the concept, observe that the OO Text document is now listed as one of the document that contains a reference to the concept.

11.6 Creating a new concept from an OO Text document

You can create a new concept in the model directly from an OO Text document as follows:

1. Put the cursor to the location in the OO Text document where you want to insert a reference to a concept that you want to create at the same time
2. Type the name of the concept as a string and select the string
3. Right-click and select Objectiver>Create Concept>Agent to create an agent, Objectiver>Create Concept>Goal to create a goal, and so on

A new concept with the name you type is created in the model and a reference to it replaces the string you typed in the OO Text document. The new concept is created in the folder of the OO Text document. If there is already a concept with the same name in that folder, Objectiver pops up an error message 'Concept … already exists'. Click OK and change the concept name.



Location :
Date:
Stakeholder:
Role in the organisation:

1.2 Interview summary

1.2.1 Main facts

Goal1

FIGURE 38. The OO Text document with the reference inserted

11.7 Searching for a concept reference in an OO Text document

A concept in an OO text document can be located in two ways:

From the property editor.

1. Select the concept in the explorer.
2. In the property editor, click the documents tab.
3. Click the Open… button on the line corresponding to the OO Text document in which you want to look after the concept reference (see Figure 39)

Objectiver opens the document and goes to the first location of the concept it meets from the beginning of the document.

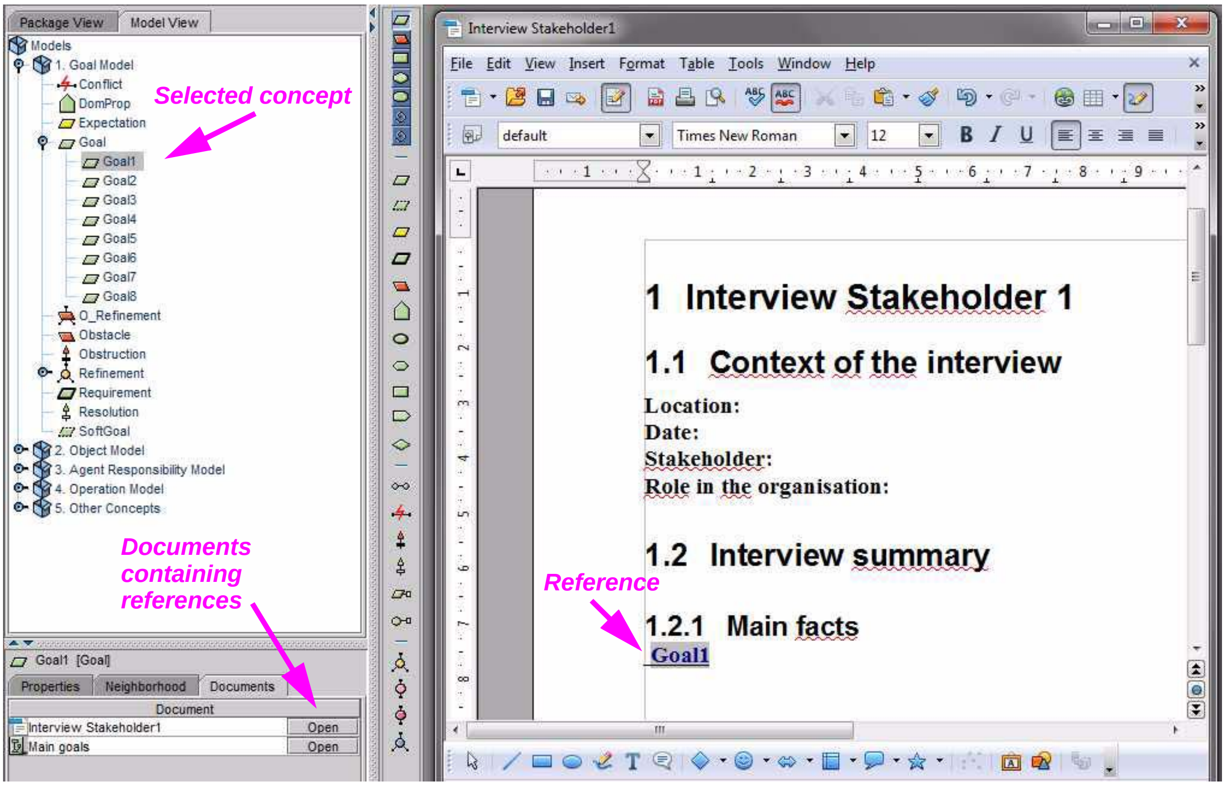
To retrieve other locations of the concept in the OO Text document, please use the Search function described below.

From the search function

In the OO Text document in which you want to locate a concept reference, proceed as follows:

1. Right-click and select *Objectiver>Search concept in document…* (see Figure 40)
2. Type the concept name fully or only the first characters of it and select the concept from the auto-completion pop-up list (see Figure 41)

Objectiver locates the first occurrence of a reference to the concept in the document. You can navigate from a reference to the next or the previous one, to the first or the last one by using the corresponding buttons '>', '<', '<<', '>>'. (see Figure 42).



Document	
Interview Stakeholder1	Open
Main goals	Open

1 Interview Stakeholder 1

1.1 Context of the interview

Location:
Date:
Stakeholder:
Role in the organisation:

1.2 Interview summary

1.2.1 Main facts

Goal1

FIGURE 39. Inserting a concept reference into an OO Text document

11.8 Locating a concept from an OO Text document

You can locate a concept in the Objectiver model from any reference to it in an OO Text document:

1. In the OO Text document, right-click the concept reference you want to locate in the model
2. Select Objectiver>Locate concept in the popup menu (see).

Objectiver locates the concept in the explorer and in the property editor.

Note. Be sure to set/unset the explorer filters correctly in order to allow the tool localize concepts. See Section 7.2 for more details on the explorer filters.

Note. Do not confuse *location* with *search*. Location aims at locating, in the model, a concept for which you provide a reference. Search aims at locating a concept reference in an OO Text document.

11.9 Renaming a concept

A concept can be directly renamed in an OO Text editor by renaming any reference to it appearing in the OO Text document:

1. Search the concept reference corresponding to the concept who want to rename.
2. Right-click on it and select Objectiver>rename concept… in the popup menu

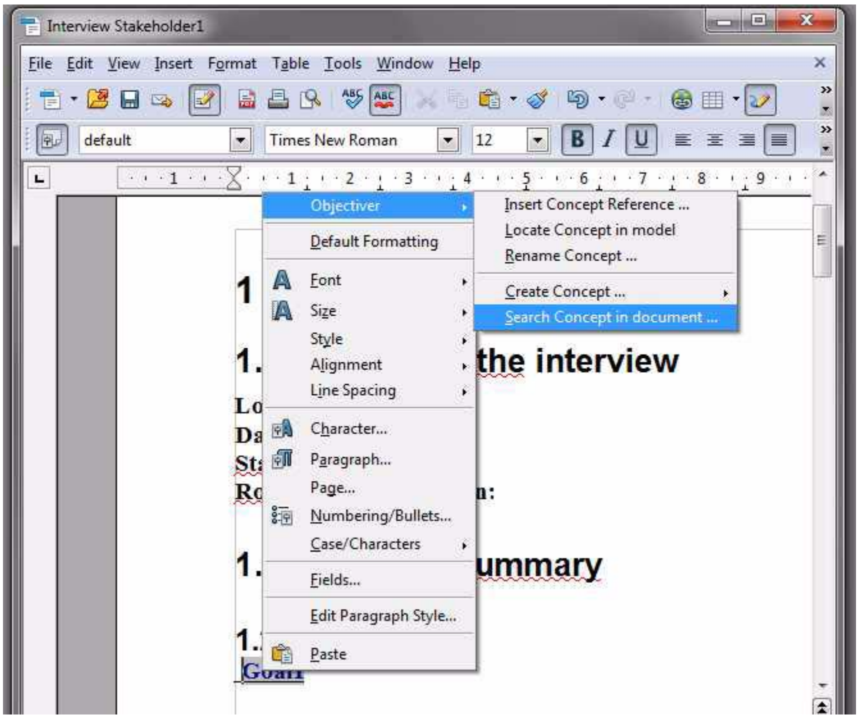
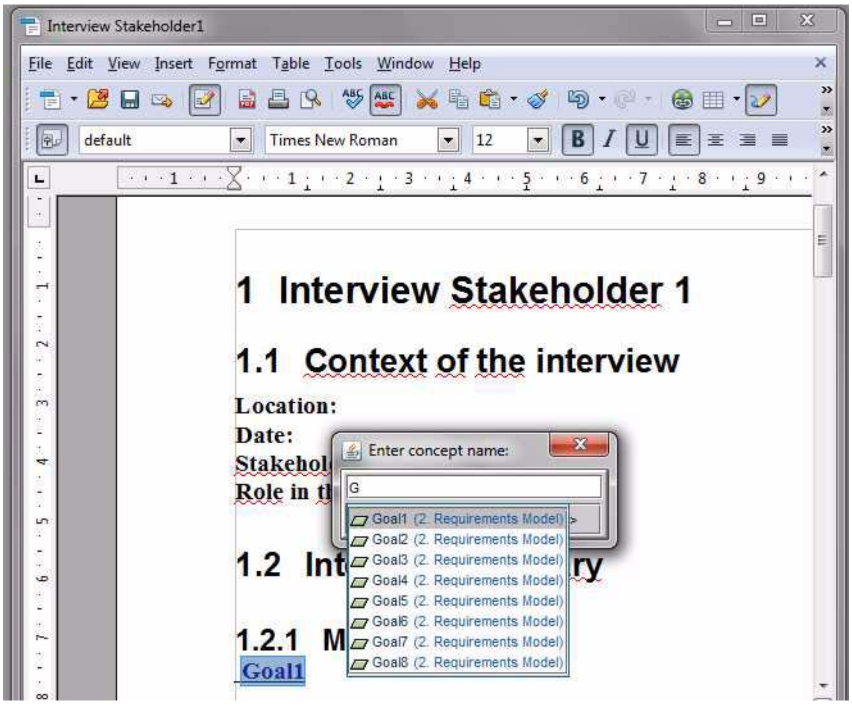


FIGURE 40. Looking for other occurrences in the document


Interview Stakeholder1
File Edit View Insert Format Table Tools Window Help
Enter concept name:
G
Goal1 (2. Requirements Model)
Goal2 (2. Requirements Model)
Goal3 (2. Requirements Model)
Goal4 (2. Requirements Model)
Goal5 (2. Requirements Model)
Goal6 (2. Requirements Model)
Goal7 (2. Requirements Model)
Goal8 (2. Requirements Model)

1 Interview Stakeholder 1

1.1 Context of the interview

Location:
Date:
Stakehol
Role in th

1.2 Int ry

1.2.1 M
Goal1

FIGURE 41. Selecting the concept to look for inside the document

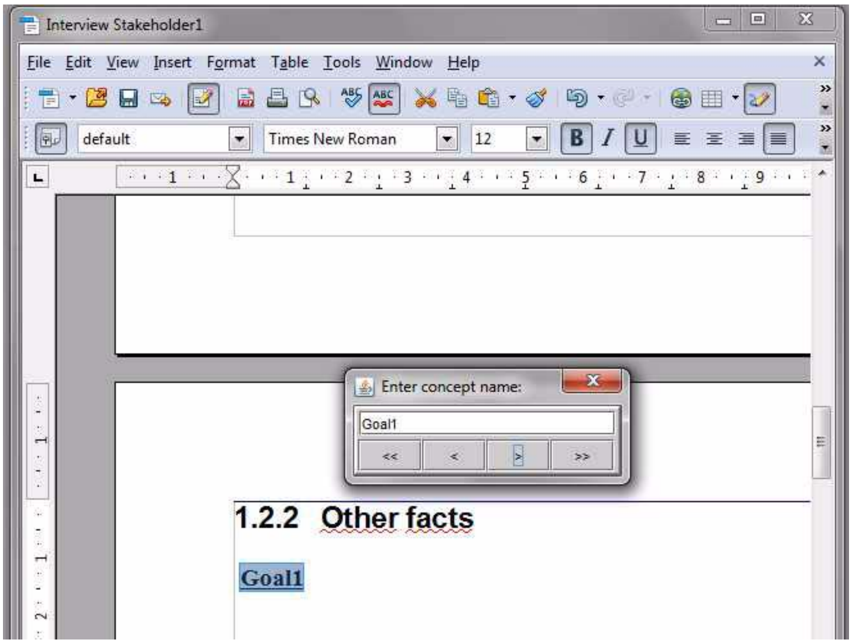


FIGURE 42. Navigating from an occurrence to another

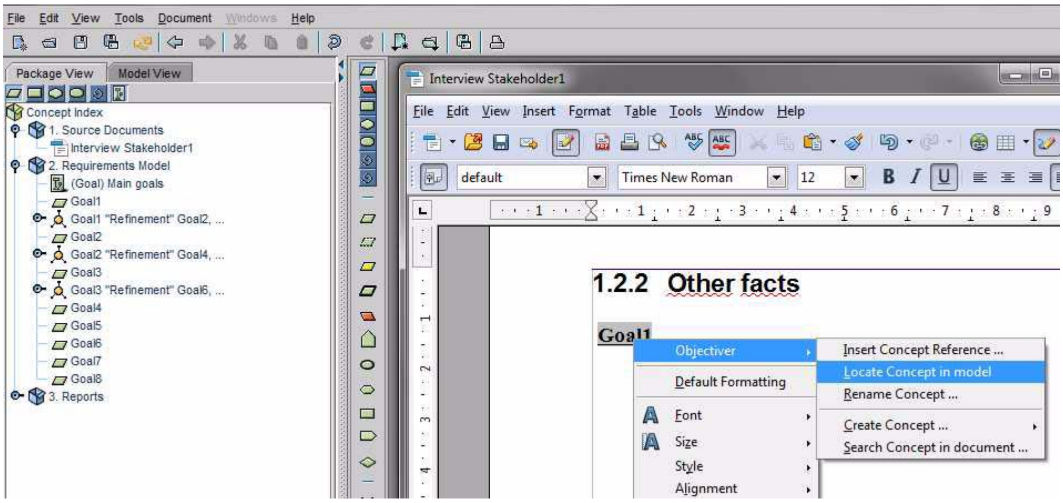


FIGURE 43. Locating a concept in the model

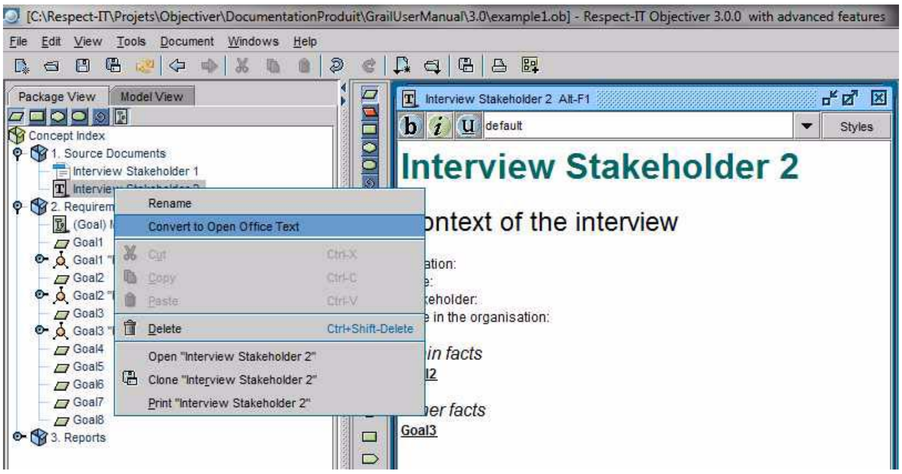
3. The tool activates the Objectiver rename dialog. Provide a new name for the concept and click OK.

The concept is renamed in the model and all references to it as well.

11.10 Converting a TXT document into an OO Text document

You can “upgrade” a TXT document into an OO Text document by using the **Convert to OO Text** function:

1. Select the TXT document you want to convert in the explorer
2. Right-click the entry and select **Convert to OO Text** in the explorer. (see Figure 44)


[C:\Respect-IT\Projets\Objectiver\DocumentationProduit\GrailUserManual\3.0\example1.ob] - Respect-IT Objectiver 3.0.0 with advanced features
File Edit View Tools Document Windows Help
Package View
Model View
Concept Index
1. Source Documents
Interview Stakeholder 1
2. Requirem
3. Reports
Rename
Convert to Open Office Text
Cut Ctrl-X
Copy Ctrl-C
Paste Ctrl-V
Delete Ctrl+Shift-Delete
Open "Interview Stakeholder 2"
Clone "Interview Stakeholder 2"
Print "Interview Stakeholder 2"
Interview Stakeholder 2 Alt-F1
default
Styles

Interview Stakeholder 2

ontext of the interview

ation:
e:
eholder:
in the organisation:

in facts

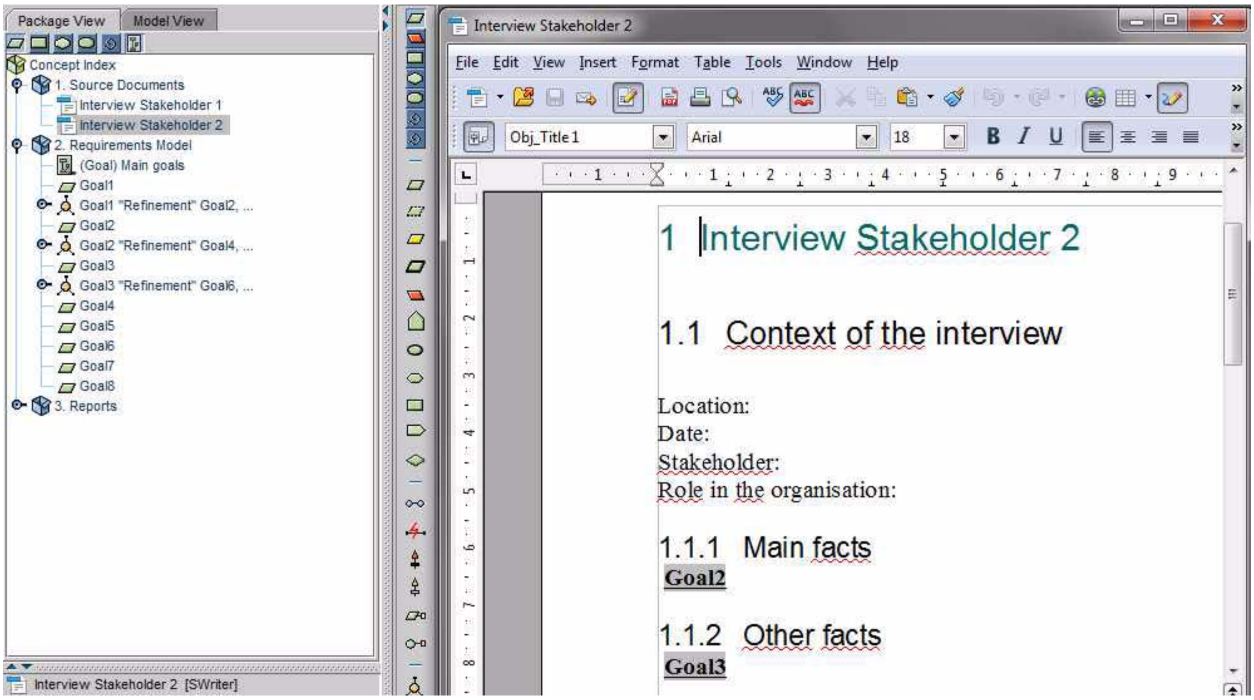
l2

er facts

Goal3

FIGURE 44. Converting a TXT document into an OO text document

The TXT styles are mapped onto corresponding OO Text styles. (see Figure 45)



1 Interview Stakeholder 2

1.1 Context of the interview

Location:
Date:
Stakeholder:
Role in the organisation:

1.1.1 Main facts

Goal2

1.1.2 Other facts

Goal3

FIGURE 45. Result of the conversion to OO Text

11.11 Converting an OO Text document into a TXT document

You can convert an OO Text document into a TXT document. As a TXT document offers less formatting capabilities, be prepared to loose not only formatting information but also some contents that TXT documents are unable to support.

1. Select the OO Text document you want to convert in the explorer
2. Right-click the entry and select **Convert to Text** in the explorer (see Figure 46).

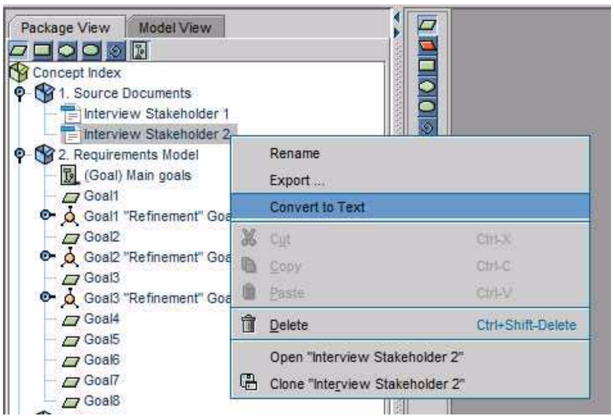
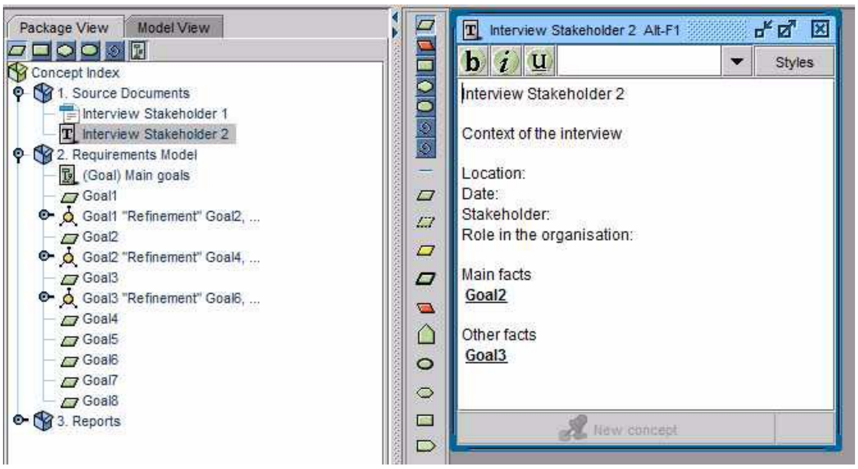


FIGURE 46. Converting an OO Text document into TXT

Limitation. In this release (3.0), only the text contained into the OO Text document is converted into the TXT document. In particular, no style is applied on the converted text (see Figure 47).



Interview Stakeholder 2

Context of the interview

Location:
Date:
Stakeholder:
Role in the organisation:

Main facts
Goal2

Other facts
Goal3

FIGURE 47. Result of the conversion into TXT (with the loss of style information)

PART D Reporting functions

12.0 Web-based Documentation

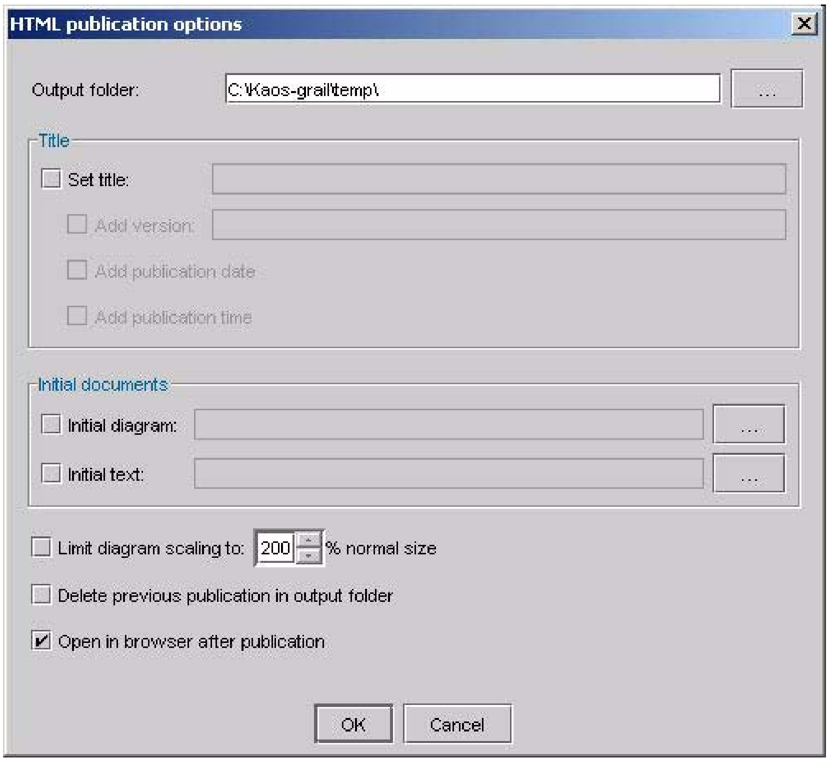
12.1 About Web-based documentation

Objectiver makes it possible to publish documentation in a Web-compatible format. This Web-formatted documentation includes all the information contained in the project. It can be distributed and is easy to navigate by using a standard web browser[1].

The next sections explain how Web-formatted documentation can be published and how it can be browsed through.

12.2 Publishing documentation:

1. Choose **File>Publish HTML...**. Figure 48 shows the available options.



HTML publication options

Output folder: C:\Kaos-grail\temp\ ...

Title

☐ Set title:

☐ Add version:

☐ Add publication date

☐ Add publication time

Initial documents

☐ Initial diagram: ...

☐ Initial text: ...

☐ Limit diagram scaling to: 200 % normal size

☐ Delete previous publication in output folder

☑ Open in browser after publication

OK Cancel

FIGURE 48. Documentation generation options

2. To change the default Output folder, click on the '**...**' button. A file dialog box called 'Select output folder' appears.

[1] In version 3.0, the following browsers fully support the rendering of the web documentation, in particular the SVG diagrams: Internet Explorer (version 6 and above, with Adobe SVG Viewer, only available on Windows), Firefox (version 2 and above, available on most OSes).

3. Locate the folder where you want the output files to be stored. It is recommended to create documentation in its own folder, that is not to store other files in that folder.
4. Click **Select folder**.
5. To set a title at the html document with additional information about the version appended, check the corresponding options and type a title.
6. To choose the diagram and the text document that shall be displayed when opening the web documentation, check the corresponding options and choose the documents.
7. To limit the maximal scaling value of generated diagrams, check that option.
8. When the checkbox *Delete previous generation in output folder* is checked, Objectiver deletes the files resulting of a previous generation.
9. To launch the html publisher, click **OK.** It can take a significant amount of time to complete.

12.3 Browsing through the web-formatted documentation

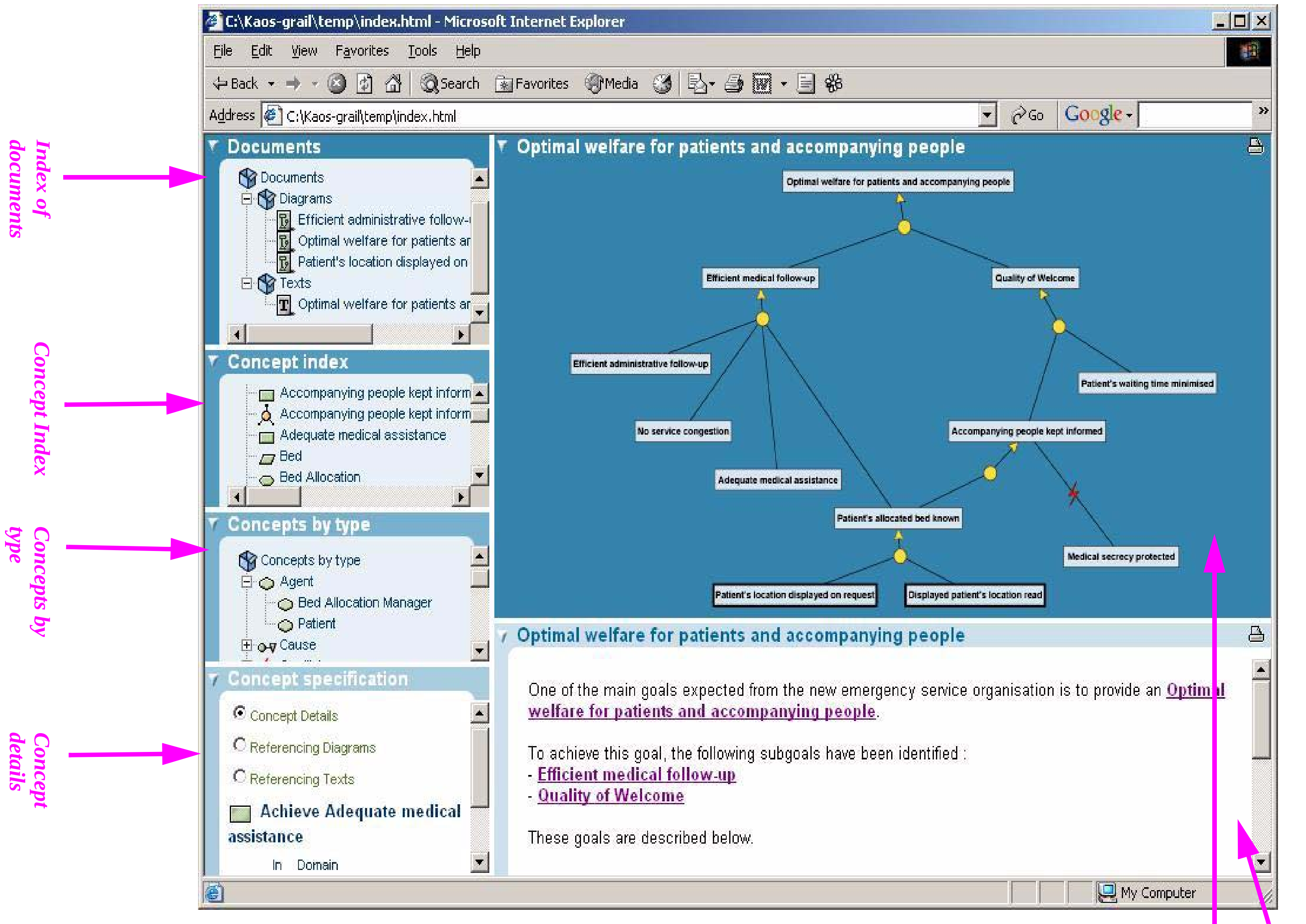
1. Go to the directory containing the specification documentation.
2. Double-click on the file named ‘`index.html`’ to open it with your favourite browser. Figure 49 shows a screenshot of the web documentation as displayed in Internet Explorer.

Note: To be able to visualize diagrams, the SVG plug-in must be installed and configured to work with your browser. To install the plug-in, refer to the Adobe web site (http://www.adobe.com/svg) and follow the instructions to download the plug-in (Adobe SVG Viewer 3.0).

12.4 About documentation contents

The documentation window is divided in several frames (see Figure 49):

- an **index of documents** grouped by package. Clicking on any item will display the corresponding document.
- a **concept index** (grouped by package) and a list of **concepts by type**.
  Clicking on a concept name updates the Concept Specification frame which shows information associated with the concept.
- a **concept specification frame** where you can get all details about the concept currently selected as well as the list of diagrams and text documents where the concept appears
- a **graphical frame** used to display the Objectiver diagrams. Concepts in the diagrams are clickable. The result is that the concept details are displayed in the *concept specification frame*. If the concept is a picture link, a hypertext link or a reference to an Objectiver document, clicking that concept will open the referred document.
- a **text document frame** for displaying the Objectiver text documents. The references appearing in the text are hyperlinks that change the current text displayed (reference to another text), change the current diagram displayed (reference to diagrams) or display concept information in the *concept specification frame* (reference to a concept).


C:\Kaos-grail\temp\index.html - Microsoft Internet Explorer
File Edit View Favorites Tools Help
Back Search Favorites Media
Address C:\Kaos-grail\temp\index.html
Go Google
Documents
Documents
Diagrams
Efficient administrative follow-
Optimal welfare for patients ar
Patient's location displayed on
Texts
Optimal welfare for patients ar
Concept index
Accompanying people kept inform
Accompanying people kept inform
Adequate medical assistance
Bed
Bed Allocation
Concepts by type
Concepts by type
Agent
Bed Allocation Manager
Patient
Cause
Concept specification
Concept Details
Referencing Diagrams
Referencing Texts
Optimal welfare for patients and accompanying people
Optimal welfare for patients and accompanying people
Efficient medical follow-up
Quality of Welcome
Efficient administrative follow-up
Patient's waiting time minimised
No service congestion
Accompanying people kept informed
Adequate medical assistance
Patient's allocated bed known
Medical secrecy protected
Patient's location displayed on request
Displayed patient's location read
My Computer
Index of documents
Concept Index
Concepts by type
Concept details
Text document
Graphical document

○ Concept Details
○ Referencing Diagrams
○ Referencing Texts

Achieve Adequate medical assistance

In Domain

Optimal welfare for patients and accompanying people

One of the main goals expected from the new emergency service organisation is to provide an Optimal welfare for patients and accompanying people.

To achieve this goal, the following subgoals have been identified :
- Efficient medical follow-up
- Quality of Welcome

These goals are described below.

FIGURE 49. Web-based documentation

12.5 Working with diagrams (SVG viewer)

12.5.1 Panning

Click and drag your mouse while holding the **Alt** key down.

12.5.2 Zooming in and out

To zoom in:

Hold down the **Ctrl** key (the cursor changes to a “zoom in” symbol) and click on the diagram.

To zoom in on a defined region

1. Hold down the **Ctrl** key and drag the mouse to draw a selection rectangle around a defined region.
2. Release the mouse button and the **Ctrl** key.

Note: To restore the diagram to its original size, select “Original View”.

To zoom out:

Select **Zoom out** from the context menu.

12.5.3 Finding the diagrams in which a concept appears

1. Select the concept name in the *index of concepts* frame.
2. Go to the concept specification frame and select the *Referencing diagrams* radio button. You can see the list of all diagrams where the concept appears.
   The referencing diagrams radio button is not displayed if the concept does not appear in any diagram.

12.5.4 Finding the text documents in which a concept is referenced

1. Select the concept name in the *index of concepts* frame.
2. Go to the concept specification frame and select the *Referencing texts* radio button. You can see the list of all text documents where the concept appears and the position where the concept appears within them.
   The referencing texts radio button is not displayed if the concept does not appear in any text document.

12.5.5 Locating a concept in an index

1. Click on *Concept index* or *Concepts by type*
2. Press **Ctrl - F** (or use another shortcut depending on your browser).
3. Type the concept name (or a part of it).

The search is done in the selected frame on the visible concepts.

12.5.6 Locating a concept in a diagram

1. Find the diagrams in which the concept you are looking for appears (see Section 12.5.3).
2. Click on the diagram name in the list of *Referencing diagrams* returned by Objectiver. The diagram appears and a big circle shows the position of the concept in the diagram (see for instance Figure 50)

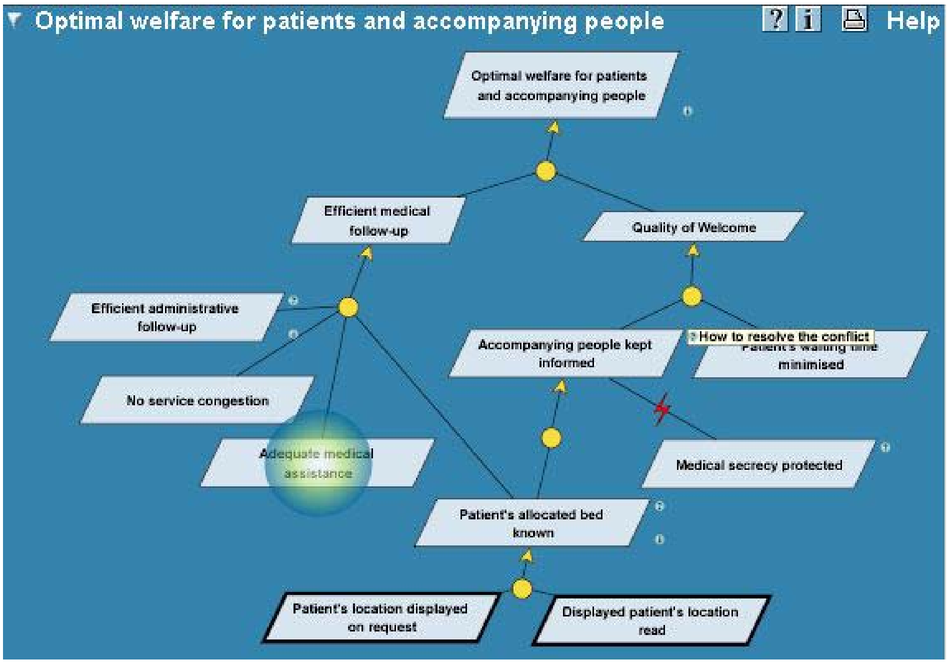

Optimal welfare for patients and accompanying people
? i Help
Optimal welfare for patients and accompanying people
Efficient medical follow-up
Quality of Welcome
Efficient administrative follow-up
No service congestion
Adequate medical assistance
Accompanying people kept informed
How to resolve the conflict
Patient's waiting time minimised
Medical secrecy protected
Patient's allocated bed known
Patient's location displayed on request
Displayed patient's location read

FIGURE 50. Concept location

Move the cursor on the circle to make the circle disappear.

12.5.7 Printing a diagram

To print the content of the graphical frame displaying the diagram in the documentation, proceed as follows:

1. Display the diagram in the graphical frame
2. If needed, resize the diagram to fit the size of the graphical frame (see Section 12.5.2).
3. Click the '**print this diagram...**' icon (the button is located in the upper-right corner of the diagram). This causes the Print dialog of your browser to pop up.

12.5.8 Show informal defs and issues in diagrams

When a issue or an definition have been specified on some concepts, the diagram sports a (?) or a (i) icon respectively near the concept. If you position the mouse cursor over such an icon, the underlying information (issue for instance) is displayed.

You get however get rid of all the (?) or (i) icons from the diagram by selecting the corresponding button located at the top right of the diagram window.

13.0 Report Edition and Publication

13.1 About Reports

A report consists of a sequence of documents. All document types can be inserted in a report. The report editor enables you to define the sequence and edit the documents composing the report.

Reports can be composed manually by taking documents from the model; they can also be generated automatically from the explorer (by using the context menu), by using the document generator using templates (see Section 15.0), or by using the requirements document generator (see Section 16.0).

This section describes manually composed reports and how to use the report publisher in order to create printable reports (see Section 13.7).

13.2 Basic Operations

Refer to Section 4.0 to learn how to create, open, delete or rename a report.

13.3 Report Editor toolbar

Figure 51 shows the buttons of the toolbar which are associated with a report editor. Functions activated from the toolbar can also be activated from the context menu (see Figure 52).

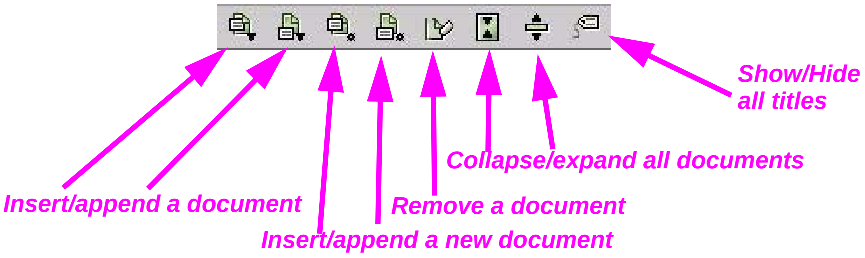


FIGURE 51. Report Editor toolbar

13.4 Adding a document to a report

13.4.1 Adding an existing document

1. Open the context menu and select the **Insert** item (to add a document before the selected document in the report) or the **Append** item (to add a document to the end of the report).

OR

Click the **Insert a document** (or the **Append a document**) button (in the report editor toolbar).

2. Select the type of document you want to add. You can add a diagram, a text or even another report. A dialog box containing the list of all documents of the selected type appears.
3. Select the document you want to add.

You can also add a document by selecting it in the explorer and dragging it to the report. Drop it where you want to insert it.

A screenshot of a report is shown Figure 52

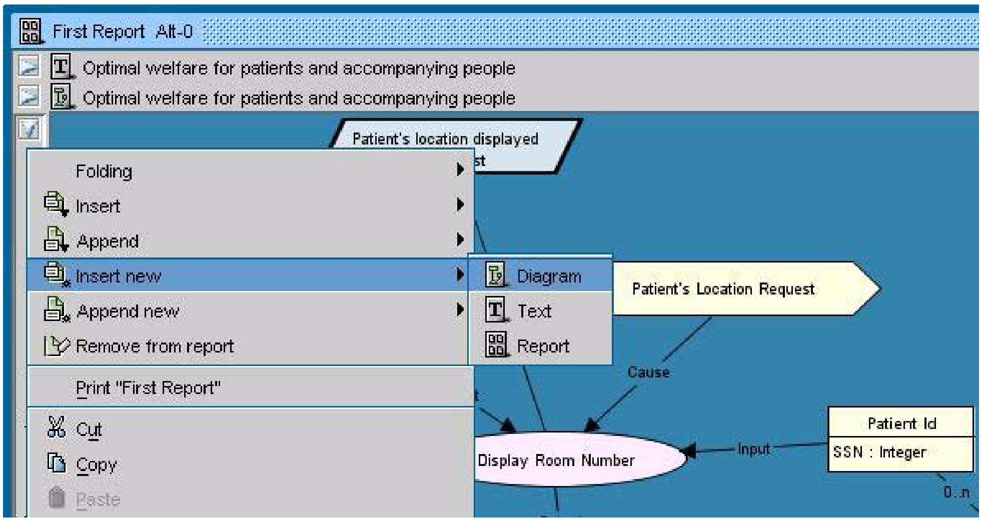


FIGURE 52. Report Editor

13.4.2 Creating a new document

Open the context menu and select the **Insert New** item (to add a new document before the selected document in the report) or the **Append New** item (to add a new document to the end of the report).

OR

Click the **Insert a new document** (or the **Append a new document**) button (in the report editor toolbar).

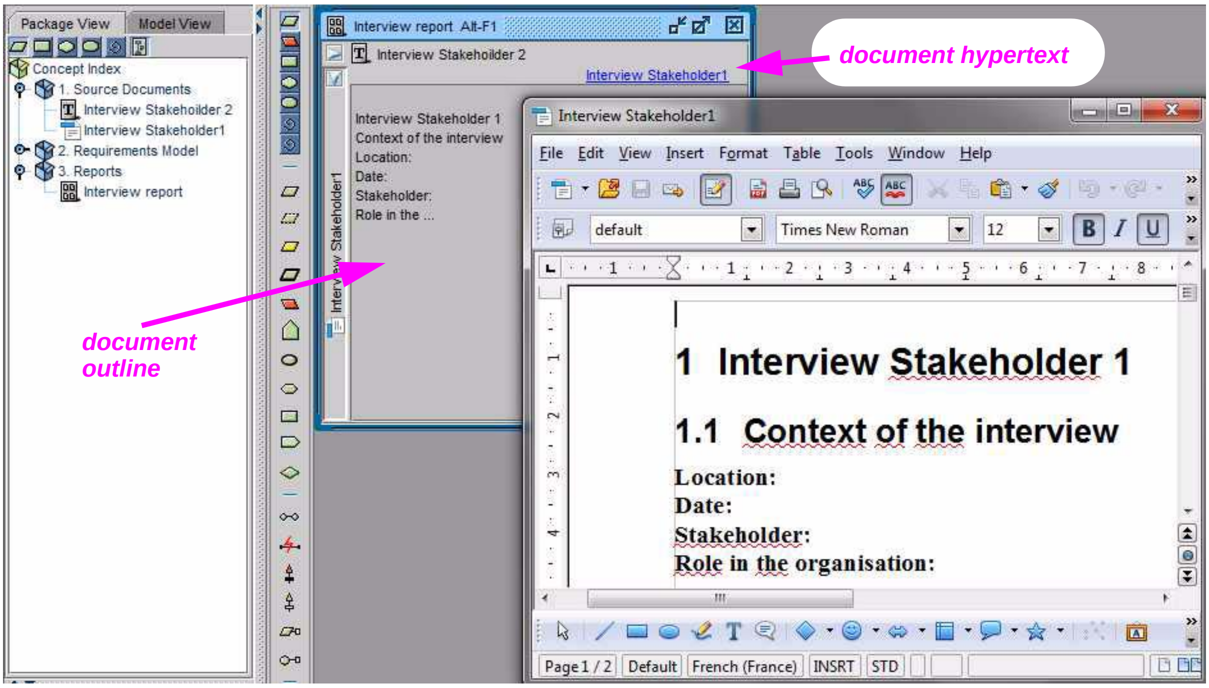
Select the type of document you want to create. You can add a diagram, a text or another report.

13.5 Handling documents inside the report Updated

Reports are managed by Objectiver internally and kept consistent with the rest of the specification.

If you want to modify a document directly from the report editor, you can do it: the diagrams and the text documents within a report are editable. The report editor provides you the same functionalities as the diagram and text editors.

Unfolding a report entry that refers to an OO Text document (see Section 11.0) will start the OO Text editor on the document in a separate window. The unfolded entry displays the first text lines of the OO Text document (if any) and a hypertext link that allows you to reopen the OO Text document from the unfolded entry (see Figure 53).



document hypertext

document outline

1 Interview Stakeholder 1

1.1 Context of the interview

Location:
Date:
Stakeholder:
Role in the organisation:

FIGURE 53. Reports containing OO Text documents

By dragging and dropping documents, the report editor enables you to redefine the sequence of documents.

13.6 Removing document from a report

1. Select in the report the document you want to remove
2. Open the context menu and select the **Remove from report** item or click the **Remove a document** button (in the report editor toolbar).

13.7 Publishing reports Updated

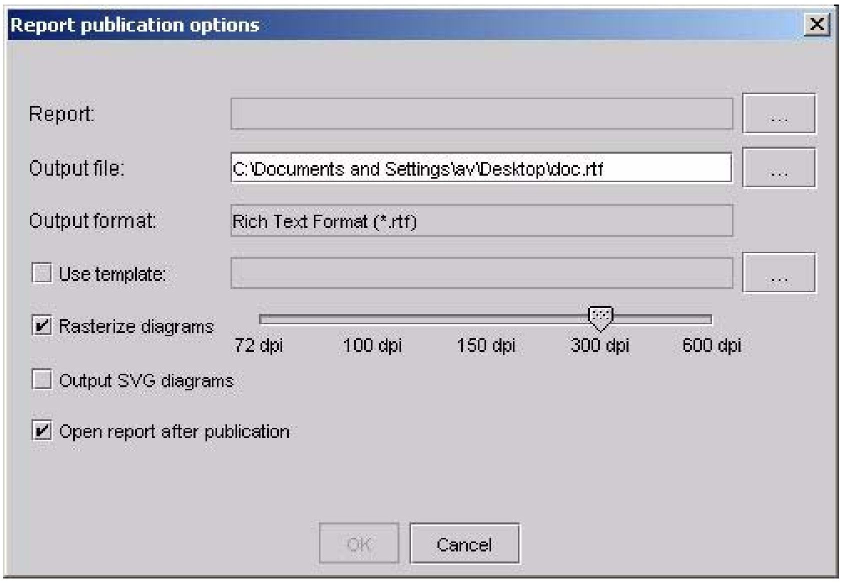
Objectiver allows you to export reports in different formats:

- RTF (“Rich Text Format” , Microsoft Word)
- ODT (for OO Text documents from Open Office)

Note. Publishing reports directly in PDF is no longer supported in version 3.0 as they are plenty of PDF converters existing on the market that allow you to convert Word or Open Office documents into PDF.

To publish a report:

1. Choose **File>Publish Report...**. A dialog box called ‘Report publication options’ appears (Figure 54).
2. Select the report document to export, by clicking on the corresponding “...” button. A dialog box containing the list of all existing reports appears. Select one of them.



Report:

Output file: C:\Documents and Settings\av\Desktop\doc.rtf

Output format: Rich Text Format (*.rtf)

□ Use template:

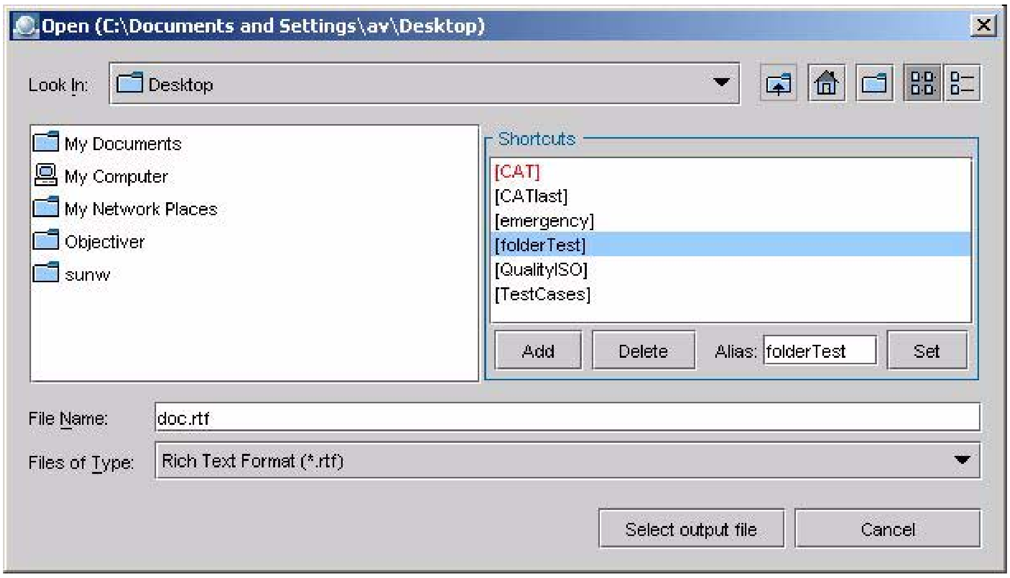
☑ Rasterize diagrams 72 dpi 100 dpi 150 dpi 300 dpi 600 dpi

□ Output SVG diagrams

☑ Open report after publication

FIGURE 54. Report generation options

3. If you want to change the default Output file or the Output format, click on the “**...**” button. A file dialog box called ‘Select output file’ appears (Figure 55).
4. Locate the folder where you want report files to be stored or create a folder if needed.
5. Select a type for your document: either RTF or OO Text.
6. Type a name for your document.
7. Click **Select output file**.
8. Choose a template for your Word or OpenOffice document. The *.dot* or *.ott* templates provided with Objectiver can be found either under the Objectiver installation directory**,** by default, **C:\Program Files\Objectiver3.0\templates** or installed in the directories used by MS Word or OpenOffice for templates (the location of those directories depends on your operating system) if you have checked this option during the tool installation set-up.
9. Check the **Output format option** to include either a bitmap representation of diagrams in your document (**.gif** format) or a vectorial representation of your diagram (**.emf** format).
10. If you select the option **Output diagrams as files**, diagrams that appear in the report will also be output as individual files in a directory structure corresponding to the package tree of your project.
11. To publish the files, click **OK.** It can take a significant amount of time to complete.



Open (C:\Documents and Settings\av\Desktop)

Look In: Desktop

My Documents
My Computer
My Network Places
Objectiver
sunw

Shortcuts

[CAT]
[CATlast]
[emergency]
[folderTest]
[QualityISO]
[TestCases]

Alias: folderTest

File Name: doc.rtf

Files of Type: Rich Text Format (*.rtf)

FIGURE 55. Output file selection

Hint. Once you have open the generated document in Word, save it in a .doc format. This will significantly reduce the size of the file.

Hint. Prefer the .emf format to the .gif format in order to produce light-weight documents.

Limitation: in this release (3.0), you can only use the ODT output format to publish your report if your report contains OO Text documents. If you need to produce an RTF document in such cases, just edit the published OO Text document and save it in RTF. Reports containing OO Text documents published with the RTF output format do **not** include the OO Text documents.

PART E Generating Requirements Documents

14.0 Requirements Documents

Once a requirements model has been created, a requirements document can be generated by Objectiver. There're three different ways to derive a requirements document from a model:

- by creating a custom report by hand
- using a *template* to guide the generation
- using the report generation *wizard*

Whatever method that is used, in the end, a report is produced that can be exported to either the RTF or the ODT file format (the file format used for OO Text documents -- see Section 13.0).

14.1 Creating custom reports by hand

Objectiver provides you with the means to create your requirements document manually. Using this method you can insert any existing Objectiver document (diagram or text) in the report being built. You can reorder the items as needed (by dragging the component name). You can create new documents directly from the report editor. Finally you can modify the content of all component documents (including diagrams) part of a report. Many of the functions of the text and diagram editors are also available when the document appears in a report. Beware that the original document is also modified since Objectiver (by default) does not create copies of the documents that are inserted into reports. You're directly editing the one and only copy of the document. That also means that if any document gets updated later on, any report including it shall automatically be updated. Notice also that you can open the component document with the text or diagram editor at the same time as a report editor is open. You'll then get several views on the same document synchronizing on each other. Opening a component document in its own separate editor may be necessary when you need to access functions (such as the style bar) that are not available in the report editor.

When creating new reports manually, we recommend to first create a new package, then create your report in that package. By default all new documents created within the report editor shall appear in this package. You can insert any document from any package in your model.

To for instance insert a title, a list of contributors, or a reference list, you can create a document right from the report tool. As indicated above these shall appear as documents in the package where the report resides.

We also recommend that you do not number the section titles manually. You should use the predefined styles of the text editor instead, e.g. “Obj_Title1”. These styles can later be mapped to styles of the target word processor (once the report is exported). See Section 16.4.1 for how to apply a Word template.

NOTE. All references appearing in documents and reports are kept synchronized with the model content which means that if you modify the name of a goal, for instance, all texts, diagrams, and reports making references to that goal shall automatically be updated for you.

14.2 Template-based report generation

For long requirements documents, a custom report generation (as described in Section 14.1) can quickly become cumbersome even if you needn’t create more than a single document.

The generation of requirements documents can however be automated in different ways :

- By systematically documenting diagrams
- Automatic generation of diagram reports
- Automatic generation of package reports
- Automatic generation of some of the requirements document sections

Here’s a description of each possibility:

14.2.1 Diagram explanation texts Updated

It’s a recommended practice that a text be associated with each diagram in your model. When such “explanation” texts exist Objectiver includes them automatically within the generated reports wherever the corresponding diagrams are inserted.

Objectiver helps you do it systematically as follows:

1. Right-click the name of the diagram you want to explain in the Explorer.
2. Select **Explain** from the context menu and select whether you want to explain the diagram with a simple Text or with an OO Text document. Updated
   Objectiver then creates a text or an OO text document, the name of which is the diagram name followed by “(text explanation)”.
3. Open the document to enter the explanation text. You can copy/paste concept references from the diagram or from the Explorer (see Section 7.5.2).

Note: The diagram and explanation text documents are associated. This association is recognized when you build a diagram report (Section 14.2.2) when you build package reports (Section 14.2.3) and also when using the requirements document wizard (see Section 14.3)

14.2.2 Creating a diagram report

You can create a report for a diagram that shall contain all related documentation.

As an option to the diagram report, a glossary of all the concepts appearing in the diagram can be requested. The concepts are presented in an alphabetical order and for each of them their definition attribute is included.

To create a diagram report, proceed as follows:

1. Right-click on a diagram in the Explorer
2. Select **Generate Diagram Report** and if you need the glossary select the 'with glossary' option in the submenu.

Objectiver creates a report named according to the diagram name followed by '(Explanation)'. The generated report contains the following four documents:

- the title (which is the diagram name)
- the diagram itself
- an explanation text
- (optionally) the glossary section

If the diagram has got an explanation text already (see Section 14.2.1), it appears within the report. Otherwise the explanation text document is created for you and appears empty in the report.

The glossary is stored in a text document named after the diagram name with a '(Generated glossary)' suffix.

14.2.3 Creating package report (Professional and Enterprise Edition only)

You can create a “package report” that shall contain all diagrams and related documentation that exist in a package. An option lets you include all the diagrams existing under any of the sub-packages.

Proceed as follows to create a package report:

1. Right-click, in the Explorer, a package name
2. Select **Generate Package Report**

The package report includes the following documents:

- the report title (which is the package name)
- a list of diagram reports for the package itself
- a list of diagram reports for all sub-packages
- a glossary

Each diagram report is the result of applying the "Generate Diagram Report" function described in Section 14.2.2 without glossary.
The package reports for subpackages are generated by applying the "Generate package report' recursively.
The glossary consists of all the concepts appearing in the diagrams of the package. The concepts appearing in sub-packages will be incorporated in the glossary of the sub-package reports.

14.2.4 Generating specific sections of the requirements documentation

Objectiver includes a document generator that can produce pieces of information ready for insertion into a requirements document. For instance the document generator allows you to produce the responsibility diagrams (one for each agent in your model). Those diagrams can be included in your requirements document as explained in Section 14.1. You can also generate a flat glossary of all the concepts in the Object model. Section 15.0 explains how to use the document generator.

14.2.5 Updating generated reports and documentation

Generated documents consist of two types of information:

- references to documents that exist elsewhere in the model
- documents generated on purpose to produce the report, such as titles, glossaries, etc. or even text documents you insert yourself in the generated report.

The update of a document in your model is automatically reflected in all the reports that contain the document. However if your report contains generated items, these are not automatically updated when the model evolves. For example, if you have generated a responsibility diagram for an agent and if you add a new responsibility to this agent somewhere in the model, this agent's responsibility diagram will not be updated. You need to regenerate the diagram and discard the previous version. Similarly if you have generated a glossary, modifications to the underlying model can cause your glossary to become obsolete.

Similarly, if you edit a report that was previously generated, all these modifications might get lost if you regenerate the report. It in fact depends on the documents you modify in the report:

- if you modify a document (such as a goal diagram) that is part of the model, the modifications are preserved since the document is not the result of a generation process.
- if you modify a document that was generated along with a report, any modification to this document shall get lost when the same report is generated once again. The report title is stored in such a document. This means that, you’ll need to

update the report title each time the report is generated… if you're not happy with the default title that the report is given by the tool.

14.3 Wizard-based report generation

Section 14.1 presents some way of building your requirements documents while retaining full control of the report structure and content. Section 14.2 explains how this process can be partially automated by having the tool generate specific parts of the final report. In this section we'll see the use of a wizard that takes the generation to a new level of automation. While the use of a wizard offers less flexibility it provides a much faster way to generate reports since most of the tasks are left to the responsibility of the tool.

The wizard available in the current release of the tool supports the generation of requirements documents whose structure is derived from the one proposed by the IEEE-830 specification. Please refer to Section 3.7 "Requirements document generation" of the *Kaos Tutorial* document (pp. 36-38). This document is available from the Objectiver web site. It is also installed on your hard drive when you run the Objectiver installer.

Here're some additional details about the generation process:

- Part of the introduction section, the **Purpose, Scope, References,** and **Overview** sections are generated from dedicated text documents.
- Part of the introduction section, the **Definitions, acronyms** and **abbreviations** sections are generated from the object model. An index of all objects is created along with their definitions.
- The **overall description section** (User Requirements) is generated by exploring the goal model from the high-level goals to the low-level goals and requirements according to a built-in strategy for running through the model (see below).
- The **system requirements section** is also automatically generated. Each subsection contains a list of requirements for some agent in the model.

In the following sections we'll explain how to prepare and generate requirements document using the wizard.

14.3.1 Providing content for the Introduction

The introduction contains the following subsections for which you'll have to provide the content:

1.1 Document purpose
1.2 system purpose
1.3 Definitions, acronyms and abbreviations
1.4 References
1.5 Overview

Proceed as follows to define the subsections 1.1, 1.2, 1.4 and 1.5:

1. (Optional) Create a package named for instance ReqDocIntro
2. For each subsection create a text document (see Section 14.2.1) in the package created in step 1 and name the document according to what it will contain (e.g., Document purpose, ...).
3. Provide the content for each of these sections

For subsection 1.3, verify that all the objects you want to be listed in the glossary have got a definition.
When generating the requirements document, an option allows you to include in the glossary the only objects that have got a definition.

Proceed as follows to provide a definition to objects:

1. Activate the query tool (see Section 10.0)
2. Select and execute the query 'Find concepts without Def' after having specified “Object” for the meta concept name parameter.
3. Enable edition of the result set by clicking the 'Refresh results and enable edition' button.
4. Edit the Definition attributes for all the objects you want to have included to the glossary. Use the Find button to retrieve the context in which the object has been defined (see Section 10.1.3 for details).

14.3.2 Defining the User Requirements section

When the requirements document is generated, each generated subsection of the User Requirements section is composed of a diagram and its explanation text. Depending on the model walking strategy selected during the requirements document generation (see Section 16.2.3), the included diagrams are either pre-existing or generated on the fly. Proceed as explained in Section 14.2.1 to document existing diagrams prior to the generation.

14.3.3 Defining the System Requirements section

The system requirements section is generated as explained in Section 15.4.1.

14.3.4 Applying the wizard

Section 16.0 details the steps that you’ll have to walk through with the wizard.

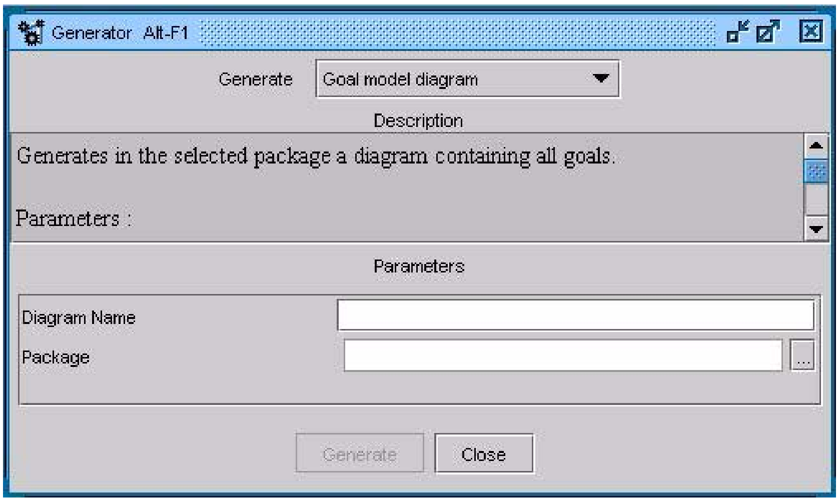
15.0 Document Generator

15.1 About the Document Generator

The document generator allows you to produce new diagrams, texts or reports (reports are described in Section 13.0). Documents are generated according to a template; some templates require parameters you need to type in before generating the document. Templates make use of predefined queries on the model.

15.2 Activating the generator

Choose **Tools>Generator** to activate the generator. Figure 56 shows a screenshot of the generator.



Generate: Goal model diagram

Description

Generates in the selected package a diagram containing all goals.

Parameters :

Parameters

Diagram Name

Package

FIGURE 56. Generator

15.3 Applying a template

1. Choose a template in the combo box.
2. If needed, give a value to the parameters.
3. Click the “Generate” button.

The document is created in the selected package (see Section 7.3) and can be opened from the Explorer.

15.4 Available Templates

Only the second and fourth template are available with the Standard Edition.

15.4.1 All Agent responsibility diagrams

The generator creates a diagram for each agent in the model. Each diagram contains the agent and the requirements and expectations for which it is responsible.

15.4.2 Flat Glossary text

The generator creates a text containing a list of each object[1] in the model (in alphabetical order) along with its informal definition (see Table 5, “Predefined concept properties”).

The parameter `Glossary Name` allows you to give a name to the new text document.
The parameter `Package` allows you to choose the destination package.

15.4.3 Goal model diagram

The generator creates a diagram containing all the goals and refinements of the model.

The parameter `Diagram Name` allows you to give a name to the new diagram.
The parameter `Package` allows you to choose the destination package.

15.4.4 Issue text list

The generator creates a text containing a list of each concept in the model (in alphabetical order) with a non empty issue property value, along with its issue (see Table 5, “Predefined concept properties”).

The parameter `Issues Document Name` allows you to give a name to the new text document.
The parameter `Package` allows you to choose the destination package.

15.4.5 Whole specification report

A report is generated that consists of all the diagrams followed by all the texts (ordered by name) in the model.

The parameter `Report Name` prompts you to give a name to the report.
The parameter `Package` allows you to choose the destination package.

15.4.6 Diagram/Document dependency diagram

The generator creates a diagram of all diagram/document dependencies. The diagram contains all the visible diagrams/documents in the specification as well as their dependencies.

The parameter `Document Name` allows you to give a name to the new diagram.
The parameter `Package` allows you to choose the destination package.
The parameter `Single Link` allows you to choose if there should be only one link between each dependent diagrams or if there should all links in the generated diagram.

15.4.7 Goal diagram dependency diagram

The generator creates a diagram of all goal diagram dependencies. The diagram contains all the visible goal diagrams in the specification as well as their dependencies.

The parameter `Document Name` allows you to give a name to the new diagram.
The parameter `Package` allows you to choose the destination package.
The parameter `Why Must Be Leaf` allows you to create a dependency only if the root text concept of a diagram is a leaf in the other diagram.

[1] Object is a generic name for Agents, Entities, Events and N-Ary Associations.

15.5 Updating the generated documents

As described in Section 14.2.5 reports that only contain documents already existing in the model remain up to date if documents composing the report are modified.

However, modifications to the model, that have an impact on the generated document (such a glossary or a responsibility model), will not be reflected automatically in those documents (it depends actually on the type of the modification).

In this release, obsolete reports cannot be updated incrementally, you need to erase and generate the report again.

16.0 Requirements Document Generator (Pro and Enterprise Edition only)

16.1 About the Requirements Document Generator

The Requirements Document Generator produces structured requirements documents from the model. It generates a report (Section 13.0). It uses templates to query the model and help you define the order in which documents are assembled. The report that is generated should be considered as a draft of your final requirements document.

Please refer to Section 14.3 to understand what the generated requirements document consists of. The following section explains how to use the wizard to generate the report (Section 16.2), optionally make modifications (Section 16.3) and finally publish it by exporting it to one of the RTF or PDF file formats (Section 16.4).

16.2 Using the wizard to generate the Requirements Document

16.2.1 Opening the Req Doc Generator

1. Select a package in the explorer. The goal section of the report that shall be generated includes the goals present in this package and subpackages. All goals refining the package goals are also included.
   Note: You can select another package later on step 3.
2. Choose **Tools>Req Doc Generator**.
3. Enter the report name (Change the generated document destination package if needed) and click **OK**. The first window of the wizard appears (see Figure 57**).**

16.2.2 First wizard window: specify the introduction section parameters

You can include Objectiver texts in the introduction of your report (see Figure 57**)** for the “Purpose”, “Scope”, “References”, “Overview”. Click the radio button to select the corresponding text. The section “Definitions, acronyms, and abbreviations” will be automatically generated. It contains the definition of the concepts of the Object model (Entities, Agents, N-Ary Associations, Events). An option permits to exclude from this section all undefined concepts (concepts with no Def property value).

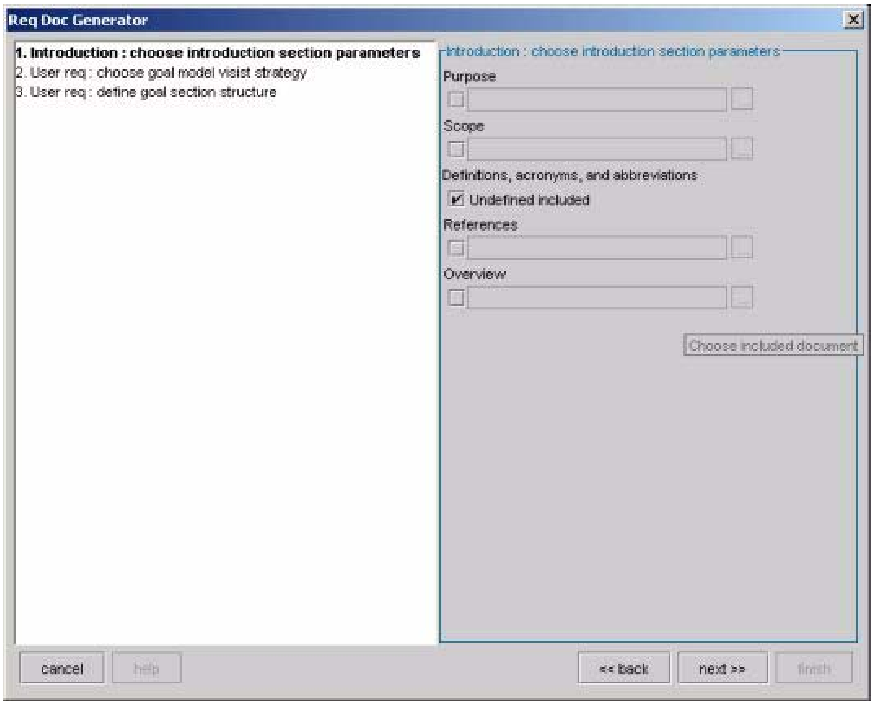
16.2.3 Second wizard window: choose a strategy for visiting the goal model

Choose a strategy for visiting the goal model (see Figure 58**).** This will define the goal section structure (see Figure 59**)**. On next step, you will be able to modify the structure of the goal section.

For the first option “Use default structure” a section will be created for each goal which is a root text of the goal model.

The second option will reuse a previous structure. This option is only available if a requirements document was previously generated for the selected package.

Select the third option to allow Objectiver to reuse the diagrams that already exist. Select the rule “Create a section for each diagram” to propose section entry for each diagram. Select the rule “Create a section for each shared goal” to propose section entry for each shares Goal (meaning a Goal refining two or more goals). These 2 rules are not exclusive.



Req Doc Generator

1. Introduction : choose introduction section parameters
2. User req : choose goal model visist strategy
3. User req : define goal section structure

Introduction : choose introduction section parameters

Purpose

□

Scope

□

Definitions, acronyms, and abbreviations

☑ Undefined included

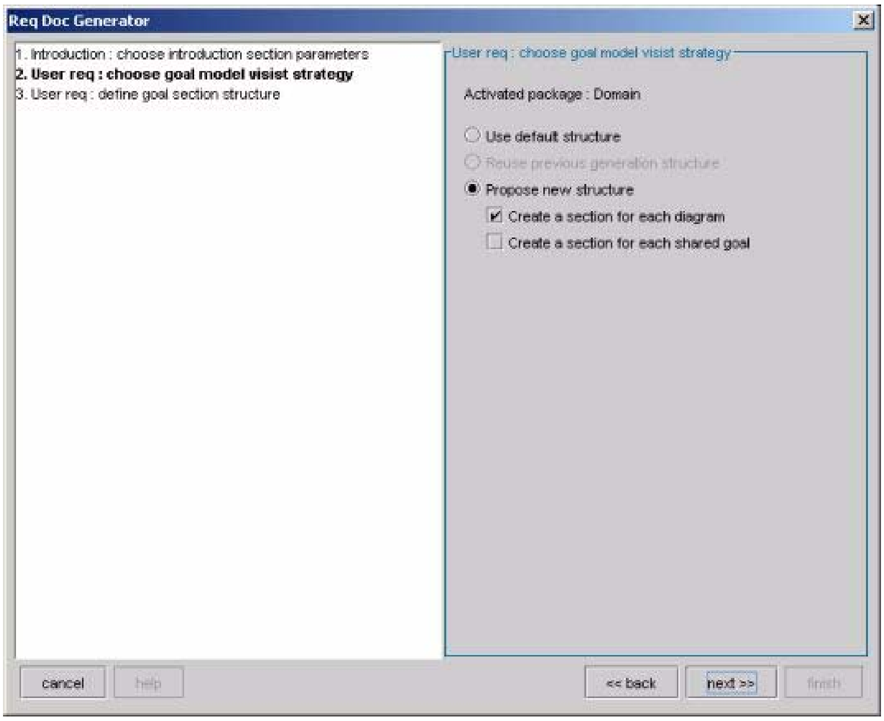
References

□

Overview

□

FIGURE 57. The first window of the Wizard



1. Introduction : choose introduction section parameters
2. User req : choose goal model visist strategy
3. User req : define goal section structure

User req : choose goal model visist strategy

Activated package : Domain

○ Use default structure
○ Reuse previous generation structure
● Propose new structure
☑ Create a section for each diagram
□ Create a section for each shared goal

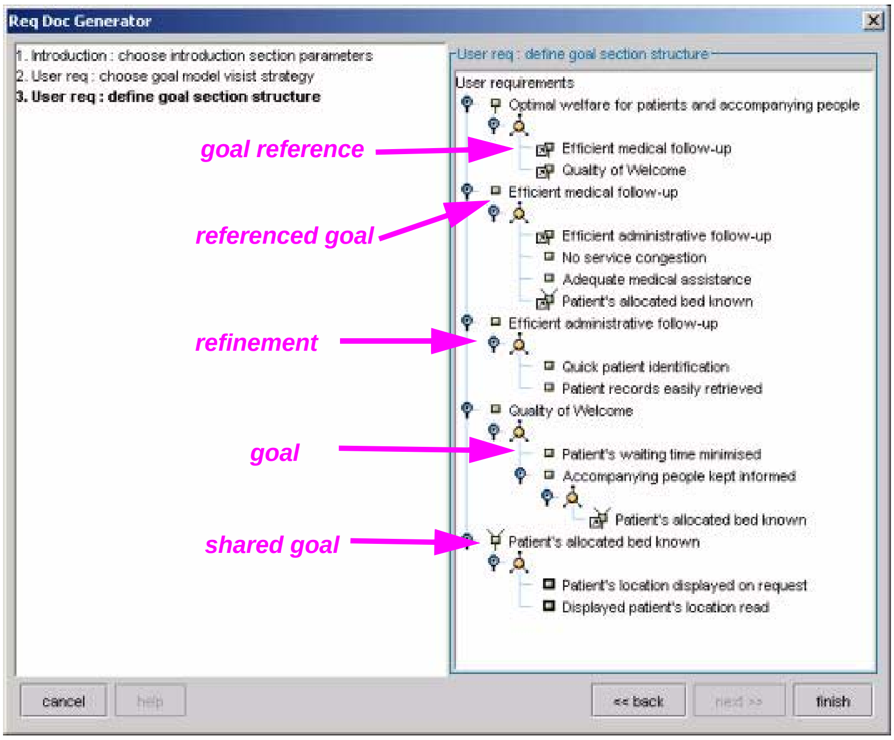
FIGURE 58. The second window of the Wizard

16.2.4 The third window of the Wizard: define the goal section structure

The wizard follows the following rules for deciding how to structure your requirements document:

1. The structure results from visiting the goal graph from top to bottom and from left to right (this is a pre-order traversal strategy).
2. The goal graph is converted into a set of hierarchical structures including references to goals instead of duplicated goals, that is, each goal (or requirement) will be defined once in one place in the reqs doc, but it can be referred to from many other places in the requirements document.
3. Each root text of a hierarchical structure will lead to a subsection of the User Requirements Section.
4. If the “Create a section for each shared goal” option is selected, each shared goal is a root text of a hierarchical structure. All other references to the shared goal point to this root text. If the option is not selected, the shared goal is decomposed when the first occurrence of it is met during traversal and the other become goal references pointing to the first occurrences.
5. If the “Create a section for each diagram” option exists, Objectiver attempts to match the goal structure it generates against existing diagrams. If a perfect match exists, the diagram will be reused during the generation.

Figure 59 shows the goal structure that will be mapped onto the structure of the document to be generated. A section in the requirements document will be created for each root text in the goal tree.



Req Doc Generator

1. Introduction : choose introduction section parameters
2. User req : choose goal model visist strategy
3. User req : define goal section structure

User req : define goal section structure

- User requirements
  - Optimal welfare for patients and accompanying people
    - Efficient medical follow-up
    - Quality of Welcome
  - Efficient medical follow-up
    - Efficient administrative follow-up
    - No service congestion
    - Adequate medical assistance
    - Patient's allocated bed known
  - Efficient administrative follow-up
    - Quick patient identification
    - Patient records easily retrieved
  - Quality of Welcome
    - Patient's waiting time minimised
    - Accompanying people kept informed
      - Patient's allocated bed known
  - Patient's allocated bed known
    - Patient's location displayed on request
    - Displayed patient's location read

FIGURE 59. The third window of the Wizard

You can fold and unfold the structure to display the following information:

- goals
- goal references
- shared goals

Before launching the generation, you can reorganise the structure by dragging and dropping items according to constraints defined in your model.

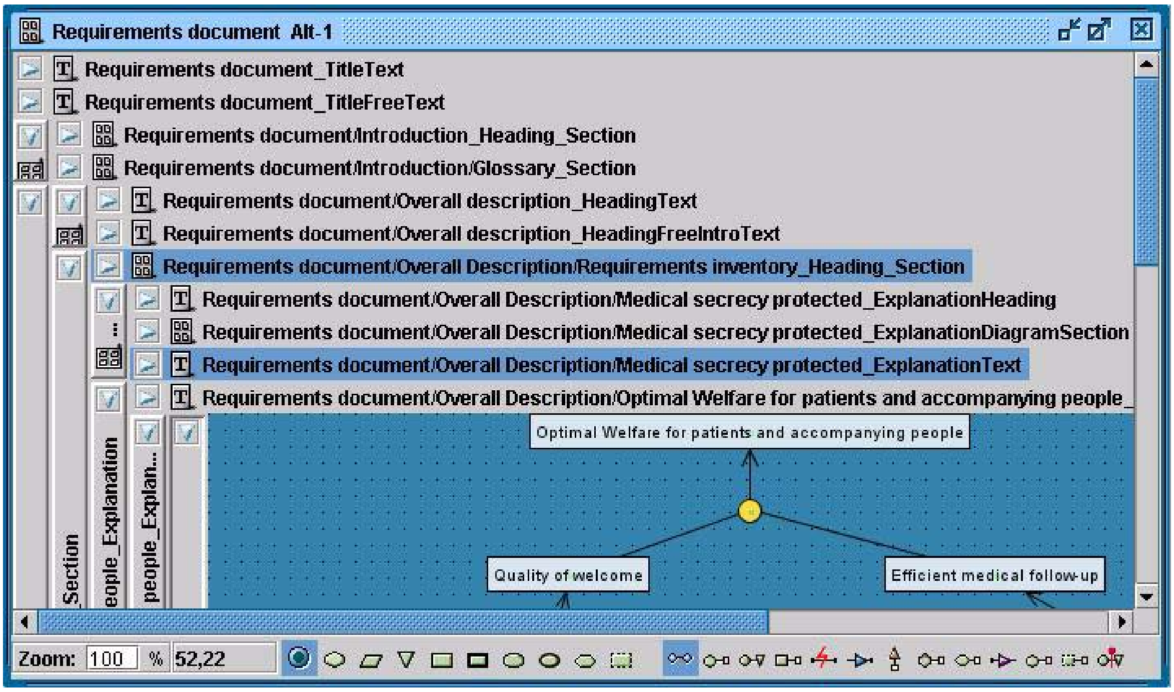
A context menu item “Create section” is available for a goal appearing inside the tree (but not on root text goals). Selecting this action will create a new section at the end of the tree.

The action “Join referred section” is available on goal references and can be used to swap the goal reference with the goal to which it refers.

16.3 Updating the generated report inside the Report Editor

As any report (see Section 13.5), the report generated according to the structure previously defined in the generation wizard (see Figure 59) can be edited. So, before publishing a report, you can modify diagrams and texts. You can also add and remove documents.

Figure 60 shows a screenshot of the generated report.



Requirements document Alt-1

Requirements document_TitleText

Requirements document_TitleFreeText

Requirements document/Introduction_Heading_Section

Requirements document/Introduction/Glossary_Section

Requirements document/Overall description_HeadingText

Requirements document/Overall description_HeadingFreeIntroText

Requirements document/Overall Description/Requirements inventory_Heading_Section

Requirements document/Overall Description/Medical secrecy protected_ExplanationHeading

Requirements document/Overall Description/Medical secrecy protected_ExplanationDiagramSection

Requirements document/Overall Description/Medical secrecy protected_ExplanationText

Requirements document/Overall Description/Optimal Welfare for patients and accompanying people_

Section
eople_Explanation
people_Explan...
Optimal Welfare for patients and accompanying people
Quality of welcome
Efficient medical follow-up
Zoom: 100 %
52,22

FIGURE 60. Generated report

Warning: If you later modify your model, some of your reports may no longer be up to date. In that case you'll have to regenerate your reports and all manual modifications that you have made for the first version of the report might need to be repeated on the newly generated report. In a future release of Objectiver, support will be provided to automatically keep generated reports in sync with the model content.

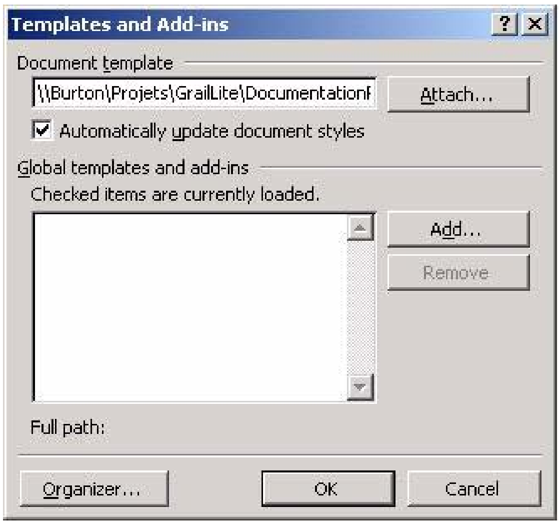
16.4 Publishing your requirements document in RTF or ODT Updated

Section 13.7 explains how Objectiver allows you to export reports in the RTF or ODT file format. Here follow some specificities for Word.

16.4.1 Word Templates

You can also assign a template explicitly to your document once it is opened as follows in Word 97-2003 and earlier versions as follows:

1. Choose **Tools>Templates and Add-Ins...** (see Figure 61).
2. Check the box in front of the adapted *.dot* template if any is displayed else click the **Add...** button and select the template (A default *.dot* template adapted to your specific MS Word release can be found under the Objectiver installation directory**,** by default, **C:\Program Files\Objectiver\templates**)



Templates and Add-ins

Document template

\\Burton\Projets\GrailLite\DocumentationF

☑ Automatically update document styles

Global templates and add-ins

Checked items are currently loaded.

Full path:

FIGURE 61. Templates and Add-ins (MS-Word)

Figure 62 shows a screenshot of a report exported to the RTF file format and making use of the *word2002-EN*-template.

You can also assign a template explicitly to your document once it is opened as follows in Word 2007 as follows:

1. Select the Office button in Word and click the Word options button at the bottom of the window
2. Select the Add-In entry, select Templates in the combo box at the bottom of the dialog window, and click GoTo. The template window displayed in Figure 61 pops up.
3. Proceed as above to specify the template you want to use.

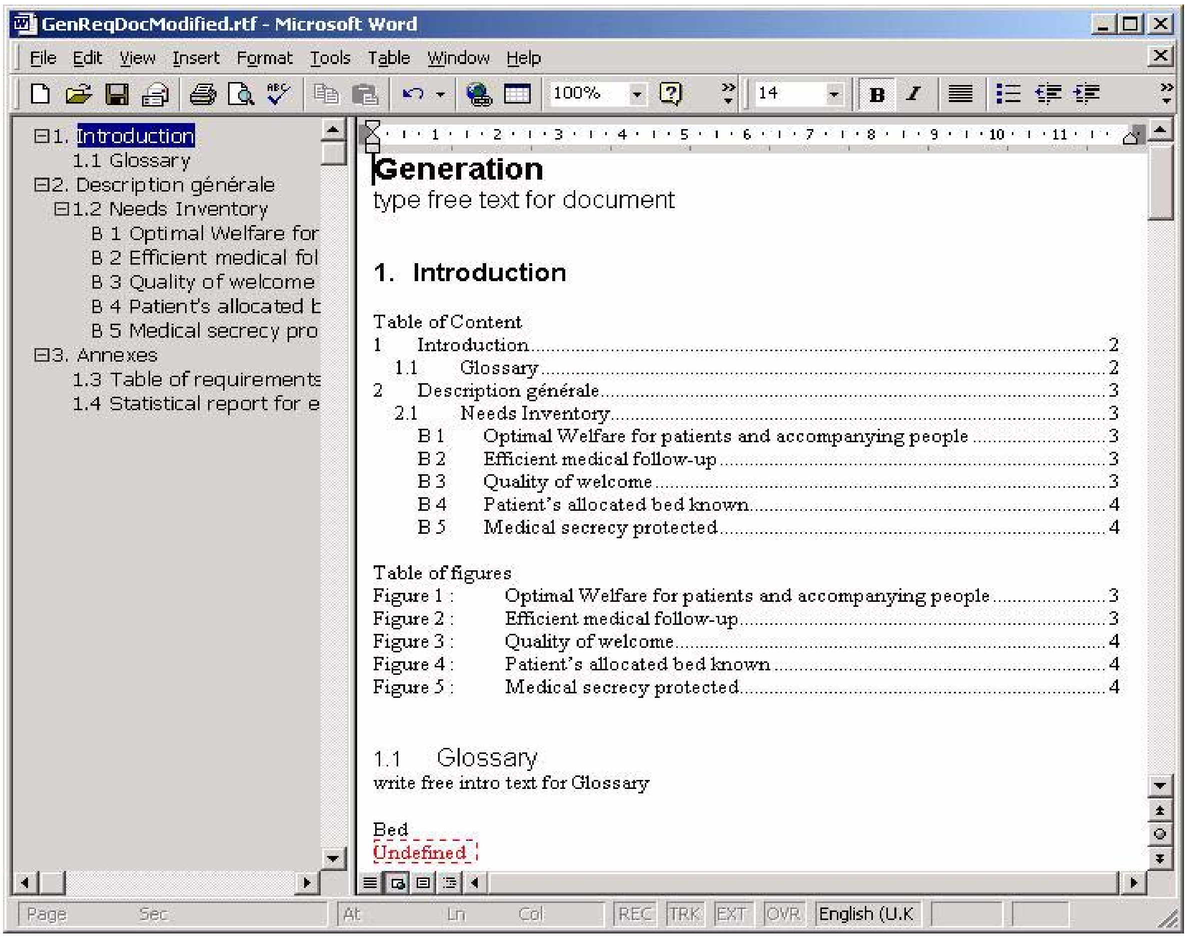
16.4.2 Word macros

The word templates provided contain Word macros that might be useful to post-process your generated requirements documents. To use those macros, check first that macro execution is enabled in your Word program (refer to the Word documentation to learn how to turn on this functionality).

1. Choose **Tools>Macro>Macros...** to activate the *Macros* frame. Figure 63 shows a screenshot of this.
2. Select the Macro you want to execute, for instance:

- *BuiltTable* to build requirements and expectations table.
- *InsertTOC* to insert a table of contents at the cursor location.

3. Click **Run.**


GenReqDocModified.rtf - Microsoft Word
File Edit View Insert Format Tools Table Window Help
100%
14
1. Introduction
1.1 Glossary
2. Description générale
1.2 Needs Inventory
B 1 Optimal Welfare for
B 2 Efficient medical fol
B 3 Quality of welcome
B 4 Patient's allocated b
B 5 Medical secrecy pro
3. Annexes
1.3 Table of requirements
1.4 Statistical report for e
Page Sec At Ln Col REC TRK EXT OVR English (U.K

Generation

type free text for document

1. Introduction

Table of Content

1 Introduction 2
1.1 Glossary 2
2 Description générale 3
2.1 Needs Inventory 3
B 1 Optimal Welfare for patients and accompanying people 3
B 2 Efficient medical follow-up 3
B 3 Quality of welcome 3
B 4 Patient's allocated bed known 4
B 5 Medical secrecy protected 4

Table of figures

Figure 1 : Optimal Welfare for patients and accompanying people 3
Figure 2 : Efficient medical follow-up 3
Figure 3 : Quality of welcome 4
Figure 4 : Patient's allocated bed known 4
Figure 5 : Medical secrecy protected 4

1.1 Glossary

write free intro text for Glossary

Bed

Undefined

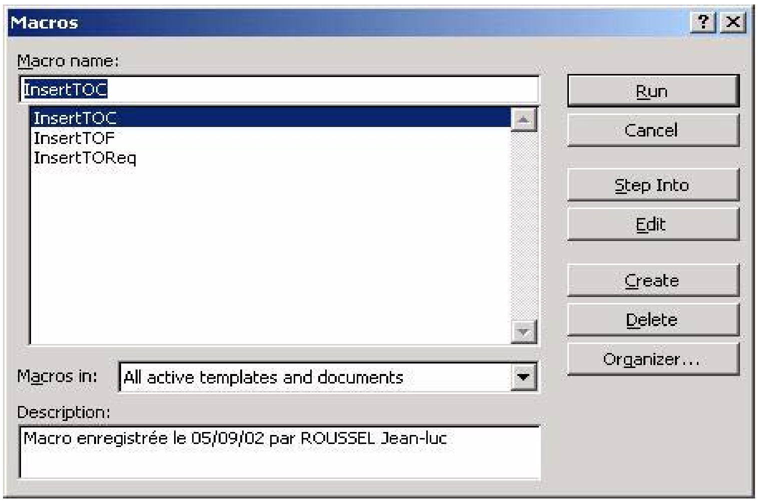
FIGURE 62. Report exported in Word

16.4.3 Using your own template

You can use predefined templates for producing requirements documents that comply with your comapny standards for instance. We provide a generic template to this aim in the distribution. You can customise it to fit your needs or use any other .doc file. In the foillowing, we explain how to use our generic template.

Directory **C:\Program Files\Objectiver\templates** contains a file named *RD-template.doc*. You can use it as a genric template to produce your own requirements document as follows:

1. Customise the *RD-template.doc* file to meet your needs (add your company logo, etc.).
1. Generate your requirements as usual in a file, let's say *RD-content.rtf*.
2. Copy the *RD-template.doc* file to a file, let's say *MyReqsDoc.doc*.
3. Edit *MyReqsDoc.doc* and go to the location at which the generated requirements document should appear.



Macro name: InsertTOC

InsertTOC
InsertTOF
InsertTOReq

Macros in: All active templates and documents

Description: Macro enregistrée le 05/09/02 par ROUSSEL Jean-luc

FIGURE 63. Macros

4. Select the Word function I*nsert Text File* and specify *RD-content.rtf* as the file to be inserted.

16.4.4 Inserting Objectiver diagrams manually in a Word document

You should have the Adobe SVGViewer plugin Version 3.0 installed.
Publish the Web-based documentation (see Section 12.2).
Then, there are two possibilities:

- By opening the Web documentation

1. Open the Web documentation (see Section 12.3) and look for the diagram to insert in Word. Right-click in the SVG diagram and select Copy SVG.
2. In Word, choose "**Edit>Paste special...**"
3. Select **"Paste"** and **"Bitmap"**

Note: To get a snapshot of the diagram on a white background, right-click on the link to the diagram in the document index and select **Open In New Window**. Then proceed as described above.

- By using Windows Explorer

Simply drag the SVG file you want to insert from Windows Explorer to your open Word document.

PART F Reviewing Objectiver Models (Pro and Enterprise Edition only)

17.0 Review process

The topics covered by this section are the following ones:

- Section 17.1 presents the different techniques that can be used during the review phase of a Requirements Analysis project.
- Section 17.2 focuses on a particular kind of review: the reviews of an objectiver model and describes different ways to organize them.
- Section 17.3 explains how Objectiver can be used to support the review process.
- Section 17.4 prompts the analyst to provide the reviewers with adequate information for the review. A structure and contents for a review instructions guide is proposed.

17.1 Types of review

There are three core activities for an Objectiver analyst:

- collecting information from all stakeholders by interviewing them, by reading documentation, by observing systems, etc.
- analyzing the information collected and building a requirements model
- deriving a requirements document from the model.

In most cases, adopting a black-box approach for these core activities is not the best way to proceed as earlier feedback from the stakeholders may be profitable for getting a higher quality result. Feedback from the stakeholders is classically achieved by means of reviews. With Objectiver, stakeholders have the opportunity of reviewing three kinds of results:

- **reviewing interview summaries**. It could be interesting for the stakeholders to review the summaries of the interviews they attend to check that their point of view has been correctly and completely understood by the analyst.
- **reviewing the model**. Experience with Objectiver has shown that reviewing the model is worth the value for the stakeholders for the following reasons:
  - validating models is generally more efficient than reading a large textual document
  - model reviews by the stakeholders contribute to make them more involved in the project
  - conflicts between stakeholders can pop up more easily, be more focused and more easily solved because they are clearly objectified. A lot of time can be lost by the analyst while going around in circles if conflicts are not addressed in an adequate way.
- **reviewing the requirements document**. The requirements document written once the requirements model has been validated, results from a large consensus. The modifications to bring on the draft to review are generally very localized and do not question large sections of the document.

This section explains how to achieve the model reviews with Objectiver.

17.2 Organizing a model review

For small organizations and small projects, reviews can be organized interactively. A review meeting is planned and all reviewers attend it. The analyst browses the model and collects feedback from the reviewers. This is probably the most efficient way of validating a model as the feedback is immediate and a consensus can be negotiated on the spot.

For larger organisations and larger projects, it is more difficult to review models interactively for the following reasons:

- it is not easy to schedule a common meeting which will put all reviewers together
- several meetings should be planned in order to browse the model entirely
- all stakeholders are probably not involved in all aspects of the model. Attending all these meetings can be perceived as a lost of time.

Therefore, in those situations, it is probably more efficient to adopt another style of review by distributing the model to review to all reviewers and ask them to review the model individually. There are typically two ways of organizing such reviews: sequentially or in parallel.

17.2.1 Sequential reviews

In this mode, a total order is defined on the set of reviewers so that there is a first reviewer, a second reviewer, ..., and a last reviewer. Each reviewer will review the model in turn. Each reviewer will benefit from the reviews of the other reviewers preceding him/her in the list. Figure 64 illustrates a sequential review with 5 reviewers. The analyst sends the model to the first reviewer who reviews the model. Then the first reviewer transfers the reviewed model to the second reviewer in the list and so on. The last reviewer sends the reviewed model with the contribution of all reviewers back to the analyst.

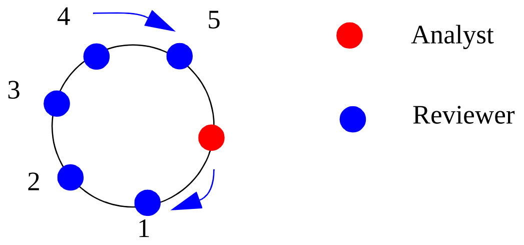


FIGURE 64. Sequential review

17.2.2 Parallel reviews

In this mode, the analyst dispatches the model to review to all reviewers simultaneously. Each reviewer reviews the model independently and returns the reviewed model to the analyst. The analyst merges the contribution of each reviewer. Figure 65 shows a parallel review organized by one analyst with 5 reviewers.

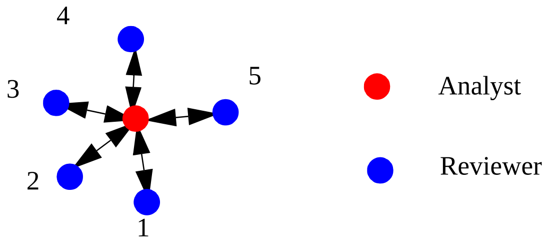


FIGURE 65. Parallel review

17.2.3 Combined schema

The sequential and parallel review modes can be combined in many ways. The following combination could represent for instance a multi-site review. The analyst dispatches the model to review on each site. Reviewers of each site reviews the model in turn and then sends a site-consolidated reviewed model back to the analyst. The analyst has then only to consolidate as many reviews as there are sites instead of as there are reviewers in the parallel review mode. Figure 66 shows for instance a review located on 3 different sites, each site having 6 reviewers. The analyst has to merge 3 reviews instead of 18 as with the parallel mode.

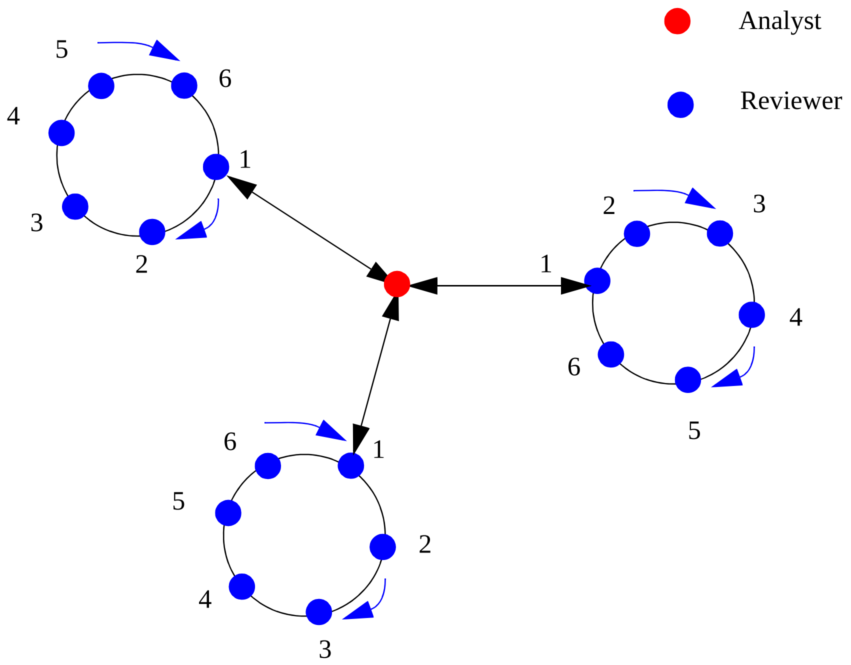


FIGURE 66. Combined review mode

17.2.4 Review mode comparison

The following table sums up some benefits (+, ++) and drawbacks (-, --) of each mode.

TABLE 6. Review mode comparison

	sequential	parallel	combined
Rapidity of feedback	--	++	+/-
Independence of reviews	--	++	+/-
Analyst’s workload to consolidate reviews	++	--	+/-

17.3 Tool support

17.3.1 Review based on the Web-based documentation

The Web-based documentation generated by Objectiver can be used by each reviewer as a support for their reviews. Once the model to review is ready, the analyst generates the Web documentation (see Section 12.0), zip the files, of which the Web documentation consists, into one archive and dispatches the archive to the reviewer(s) according to the reviewing mode selected (see Section 17.2). As the Web documentation is read-only, the reviewer has to write his/her comments in a separate file. With the archive, the analyst can therefore also prepare and send a review file that will contain the review notes from the reviewer. Depending on the review mode, the reviewer having filled in his/her comments, has next to send the review file back to the analyst or to the next reviewer.

Alternatively, the Web documentation and review file can be made available to the reviewers on a Web site (intranet or extranet).

This way of proceeding becomes quickly cumbersome for large projects. On the one hand, one has to read the model and the notes of other reviewers in separate documents. On the other hand, one has to put review notes at the right place in the review file.

17.3.2 Review based on Objectiver reports

Another way to support a review consists, for the analyst, in generating from the model a MS Word report containing all the needed information. Each reviewer annotates the report in turn according to the selected review mode.

Annotating Objectiver diagrams in Word can be a bit tricky. Moreover, the analyst has to map the review notes back in the model.

17.3.3 Review based on the Reviewer Edition of Objectiver

The Reviewer Edition of Objectiver aims at helping the analyst and the reviewers to cooperate efficiently during reviews. On the one hand, unlike the Web-documentation, the reviewer can annotate the diagrams directly while browsing the model. Review sessions during which a group of reviewers annotate the model together can be organized. On the other hand, the tool automates the merge of the review notes coming from different reviewers in the model itself.

The Reviewer Edition is a limited version of Objectiver. It allows a reviewer to annotate an Objectiver model by adding review notes and by linking those notes to elements of the model. The reviewer is not allowed to modify the model contents.

The analyst can import review notes one by one in his/her model and visualize them all on the spot. The tool displays also the owner’s name of the review notes.

Figure 67 shows a typical review process in which the analyst and the reviewer cooperate to produce a consolidated review.

Section 18.0 explains how to perform each step of this process.

17.4 Review instructions

Reviewers working on their own need some guidance, specially if it is the first time they are reviewing a model with Objectiver. Therefore we recommend to brief them on the methodology, on what is expected from them and how they can review the model with the tool.

The best way of doing it is probably, whenever possible, to organize a meeting with the reviewers to explain them the review objectives and to expose them the tool.

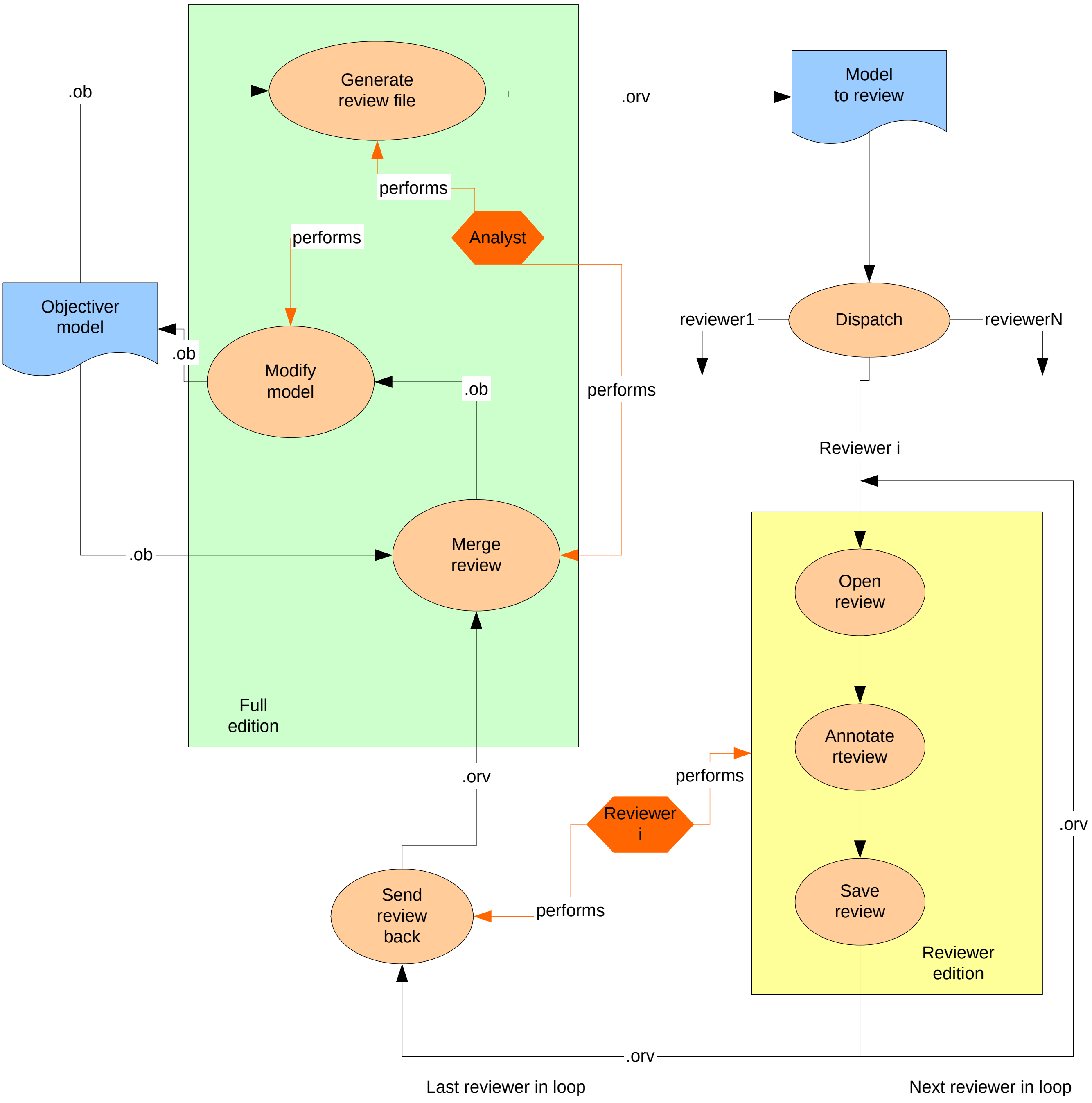


FIGURE 67. Review process with Objectiver

Even if such a meeting is organized, it is a good practice to provide the reviewers with a document they can refer at any time during the review and which contains all the information they need. This document can be appended to the model as a text document so that reviewers can refer to it at any time while reviewing the model.

A typical structure for this review manual is as follows:

1. Review management
    - 1.1 Scope of the review
    - 1.2 Reviewers
    - 1.3 Review process
    - 1.4 Deadline
    - 1.5 Support

2. Technical aspects
    - 2.1 Model structure
    - 2.3 Review activities
    - 2.2 Annotation style

The next paragraphs below explain the contents of each section in detail.

Scope of the review. This section aims at specifying which part of the model is to be reviewed. For large projects, it can be interesting to split reviews in different phases, for instance, the current and the next system and/or the user requirements and the system requirements.

Reviewers. This section provides the list of people appointed to the review. The list for instance could provide the following information for each reviewer: name, organization, position, contact information.

Review process. This section describes how the review is organized: sequential, in parallel or a mix of both. It specifies a partial or total order for the reviewers to be involved. If some reviewers have to review only parts of the model, this section specifies it.

Deadline. This section provides information on when the review is expected to be achieved for each reviewer or set of reviewers.

Support. This section describes how reviewers can find support during the review and where they can find additional documentation

- at the tool level (e.g., how to annotate a diagram)
- at the methodological level (e.g., what is the difference between a goal and a requirement)
- at the model level (e.g., what is the meaning of a specific goal).

A user manual dedicated to reviewers comes with the Objectiver Reviewer Edition. This manual specifies how to use the tool for reviews and provides a short definition of the types of concepts that can appear in an Objectiver model.

Model structure. This section is important as an Objectiver model is a network of concepts and reviewers might need some guidance to start the review. This section therefore should explain how the model is structured and should propose a traversal order for it. Typically, it consists of a table of contents listing the documents to review.

Review activities. This section describes what is expected from the reviewer, for instance:

- review the source documents (e.g., interview summaries) in order to validate how they have been parsed and linked to the model (traceability links)
- review goal diagrams to validate correctness and completeness of the description (are the goal refinements complete and correct?, etc.)
- review responsibility diagrams to check completeness and agreement on what is assigned to each agent
- review definitions of the object model
- ...

Annotation style. Types of reviewer annotations can be typically classified into three categories:

- opinion: e.g., disagreement about a requirement, about a note of another reviewer, about how the model is structured, etc.
- request for change: e.g., add a goal, rename a concept, etc.
- advices: e.g., refer to additional information, etc.

In any case, constructive notes are generally the most appreciated. Just annotating the model by qualifying some concepts as “wrong”, “bad”, etc. will not help the analyst very much to improve the model. A good note not only raises an issue but also suggests a solution or hints to a solution for it.

18.0 Reviewing operations

This section describes the operations an analyst has to do with the tool in order to implement a review, mainly how to generate a review file and how to merge reviews from several reviewers.

Description of how the reviewers can annotate models is provided in the user manual dedicated to reviewers: *Reviewing an Objectiver model*.

18.1 Generating a review file

Choose **File>Create Review file...**
A file dialog window pops up and prompts you to enter a filename.
A file with the "orv" extension is created.

18.2 Merging review files

After having loaded the file to review, import each review file like this:

1. Choose **File>Import Review...**
2. Select the directory where your review file resides.
3. Double-click the file corresponding to your review.

Notes are placed in the "Review Notes" package and are added inside diagrams.

18.3 Handling notes

To be sure to have taken into account all notes, you can for example:

1. When a note has been reviewed, change the state of this one in the property editor from `Pending` to `Reviewed.` And possibly, remove it from the diagram.
2. Checking all notes have been reviewed:
   The query `Search notes to review` (see Section 10.0) displays all pending notes.
   Right-click a reviewer in the explorer (in the "Reviewers" package) and click **Get notes** to display all notes submitted by that reviewer.

PART G Team working with Objectiver (Enterprise Edition only)

19.0 Control units

In large organizations, the requirements engineering phase has to be performed by several analysts working in team. The duration of the requirements engineering phase can indeed be significantly reduced

- if several analysts collect the information in parallel (interviews, system observation, reading documents)
- if they can feed the Objectiver model concurrently with the information collected
- if the effort for merging the contribution of each analyst by removing potential duplicates is not too high.

The teamwork features of Objectiver have been designed with these goals in mind. The main points are the following ones:

- An Objectiver model can be decomposed into **control units**. Each control unit can be **locked** and **unlocked** for edition by the user. Locking a control unit provides the user with the guarantee that nobody else can edit the control unit while he/she is modifying it.
- The tool informs the user about the **control unit status** (i.e., whether the control unit is locked or unlocked), who is locking a control unit and when a control unit is locked and unlocked.

The next sections are structured as follows:

- Section 19.1 defines control units and explains how they are managed by the tool.
- Section 19.2 and Section 19.3 explain how to define and use control units in Objectiver.
- Section 19.4 provides hints about how to use control units on large projects.

19.1 Overview

19.1.1 Definition

Section 7.3 has shown how to decompose an Objectiver model into packages. Packages are containers for model concepts. That enables the analysts to put related concepts at the same location in the model. This conceptual proximity of concepts in a same package means also that the analysts often need to edit several concepts at a time in the same package. Therefore the package appears also to be the best level at which a group of concepts must be locked by one user for edition so that all the other users are prevented from editing the same concepts at the same time.

Control units in Objectiver are a particular case of packages: they are containers that can be locked by a user to edit alone the concepts contained in the control unit.

The following two sections explain how control units are stored in Objectiver and general principles that are worth knowing in order to work with control units in an appropriate way.

19.1.2 Representation of control units

In Objectiver, each control unit is stored in a separate file. When the user defines a control unit on top of a package, Objectiver splits the .ob file containing the whole original project into two pieces:

- a .ocu file containing all the information related to the control unit
- a .ob file containing all the remaining information

Figure 68 shows an example of a project containing two packages A and B which the user wants to convert into control units. Section 19.3.1 and Section 19.3.2 explain how to convert a package into a control unit and conversely. The figure shows the state before and after the transformation.

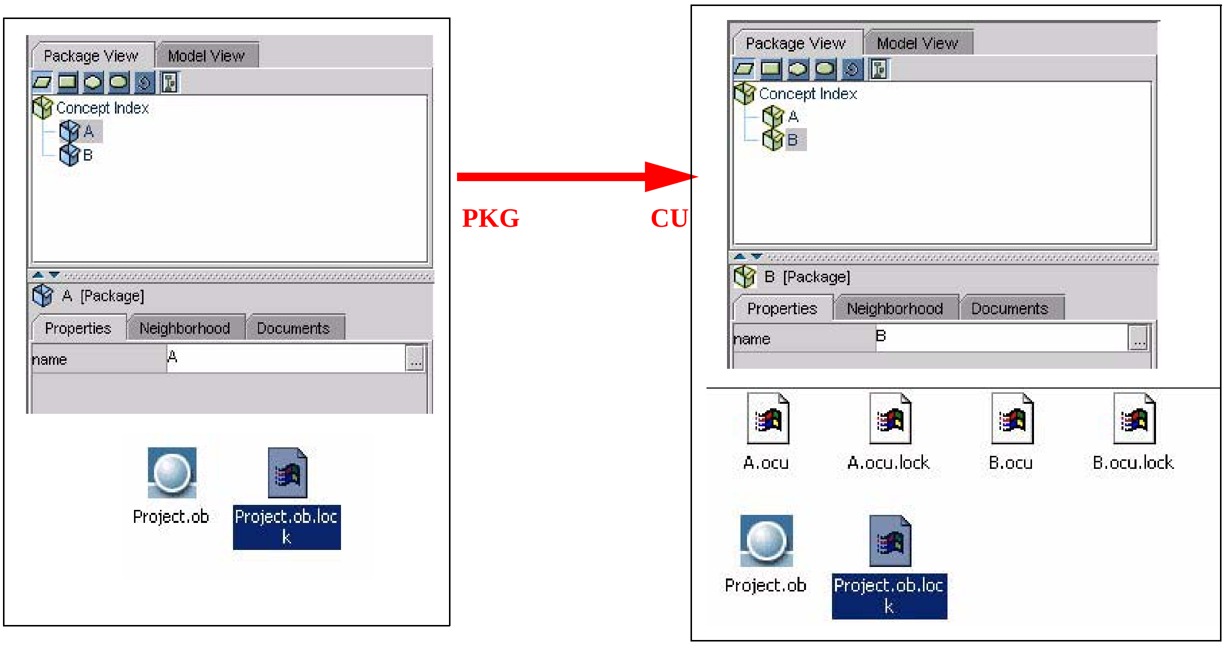


FIGURE 68. Project with and without Control Units

When a user opens an Objectiver project (.ob file), the tool reconciles in memory the .ob file with all the .ocu file.

Notes

1. .ocu files are intended to be used with the same original .ob file, from which they come. .ocu files may not be shared between different projects.
2. The .ob file is split into several pieces so that Objectiver can manage concurrent accesses to each piece by using a locking mechanism on the files.

19.1.3 General rules

Access rights are defined on the control units to restrict modifications to one single user at a time, namely, the one who has locked the control unit.

R1. A user can change the contents of a control unit only if he/she has locked it. Only one user at a time can lock a control unit.

If control units limit *writing* accesses to concepts, they do not introduce any limitation on *reading* accesses to them.

R2. All concepts, packages, control units and documents in a project are always visible to any user having opened the project, whatever the location of those items is: in packages or in locked/unlocked control units.

To manage concurrent accesses to the global project as for a single control unit, the root text of the package hierarchy behaves like a control unit.

R3. In any project, the root text of the package hierarchy, namely, Concept Index behaves like a control unit.

Control units can also contain packages but cannot contain other control units. We need therefore to specify how a lock put on a control unit propagates to the packages defined inside it.

R4. Locking a control unit A locks all the packages included in A. Sub-packages and the concepts they contain are also locked in a recursive way.

For instance, consider the following project structure (Figure 69).

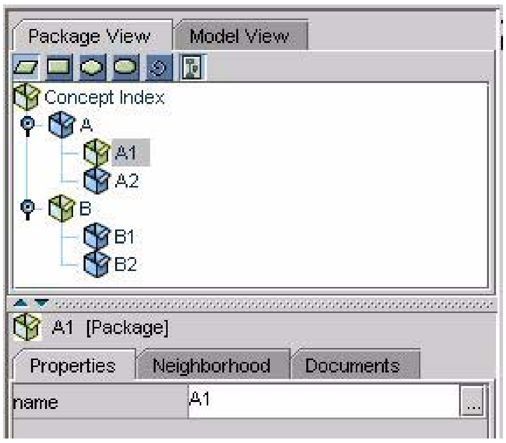


FIGURE 69. Packages and control units

Entries with a blue box icons (A, A2, B1 and B2) stand for packages. Entries with a green box icons (Control Index, A1, and B) stand for control units.

Locking the Concept Index also locks package A and package A2. Control units A1, B and package B1, B2 are not locked. They can be locked by any other user.

The next rules explains how control units impact information display in documents.

R5. In any diagram, all the concept references belonging to a control unit you haven't locked are dimmed and read-only. All the concept references belonging to a control unit you have locked are displayed in plain text.

R6. In a diagram belonging to a control unit not locked by you, no concept references can be added, deleted from the model, removed from the diagram or moved.

R7. In a diagram belonging to a control unit you have locked, references to concepts defined in any control unit on which you don’t have the lock cannot be deleted from the model but they can be moved, removed from the diagram and linked to other concepts.

The preceding rules mean that diagrams in control units the user has not locked are completely frozen. If the diagram belongs to a control unit the user has locked, he/she can control the diagram contents but never change a concept referred in the diagram if this concept belongs to a control unit he/she has not locked.

The following rules are similar to the previous ones. They apply on text documents.

R8. In a text document belonging to a control unit you haven’t locked, the text and concept references cannot be edited. No concept references can be added, deleted from the model, removed from the diagram or moved.

R9. In a text document belonging to a control unit you have locked, references to concepts defined in any control unit on which you don’t have the lock are read-only. They cannot be deleted from the model but they can be moved or removed from the document.

19.2 Status of Control units

As many users can simultaneously work on different control units of a same project, it is important for them to understand at any time which parts of the model they control, which parts of the model are being edited by other users, who has a lock and how users are notified about changes made by the other users.

The explorer uses colors, tips and character styles to provide this kind of information regarding the control units.

R10. The explorer always displays in plain text the name of the packages and control units locked by the user.

R11. The explorer always displays in dimmed text the name of the packages and control units not locked by the user (that is, locked by another user or free to edit).

R12. The tip appearing on top of a control unit entry in the explorer provides the name of the user who has locked the control unit.

The following color codes are used to inform the user about the status of a control unit:

TABLE 7. Color codes for control units

Color code	Status of the control unit
Blue	package
Green	control unit edited by the user or free for edition
Orange	control unit edited by another user
Purple	control unit that has been saved by another user since the last time the user has loaded it

The following table combines the text and color attributes for control units:

TABLE 8. Status of the control unit

Status	Plain	Dim
Blue	package belonging to a control unit you have locked for edition	package belonging to a control unit nobody or another user has locked for edition
Green	control unit you have locked for edition	control unit nobody has locked for edition
Orange	NA	control unit locked by another user
Purple	NA	control unit that has been or is still being locked by another user and that has been saved since the last time the user has loaded it

Notation. In the following sections, status of control units will be represented with a couple of values, for instance, (Blue, Plain).

19.3 Operating control units

This section specifies two groups of operations on control units.

The first group consists of specific operations on control units. They are listed in the following table:

TABLE 9. Operations on control units

Name	Effect
transform into control unit	convert a package into a control unit
transform into package	convert a control unit into a package
edit	lock a control unit for edition
commit	save changes made on the control units and release the lock
refresh	load the latest saved version of the control units

The second group of operations consists of operations existing in the Objectiver environment that impact the management of control units, namely, opening a project, saving a project and quitting. They address the following topics:

- what are the status of packages and control units when you open a project?
- how does the Save project command behave regarding control units?
- what happens when a user who has edited some control units quits or close the project?

19.3.1 Creating a control unit

Control units are created by transforming packages as follows:

1. Right-click the package you want to convert into a control unit.

2. Select **Control unit>Transform into control unit** in the pop-up menu. Objectiver opens the file manager dialogue window to prompt you to enter a name for the control unit.
3. Select a location for the control unit and type its name. Click OK. Objectiver then creates a .ocu file and transfers the contents of the control unit from the .ob file to the .ocu file. The status of the container changes from (Blue, Plain) to (Green, Plain).

Notes

1. A good practice consists in putting all control units in a directory defined at the same level as the .ob file.
2. To convert a package into a control unit:
    - you need to have locked the Concept Index (See Section 19.3.3 for how to acquire it)
    - the package to transform cannot be inside another control unit
    - the project must already be saved

If one of these conditions is not valid, the **Control unit>Transform into control unit** entry will be disabled.

19.3.2 Converting a control unit back to a package

Control units can be converted back into packages:

1. Right-click the package you want to convert into a control unit.
2. Select **Control unit>Transform into package** in the pop-up menu. Objectiver then merges in memory the contents of the .ocu file associated to the control unit with the contents of .ob file.

Notes

1. To convert a control unit into a package, you need to have locked the Concept Index. If the **Control unit>Transform into package** entry is disabled, it means that you haven’t locked it. See Section 19.3.3 for how to do it.
2. .ocu files are not deleted by Objectiver. It is up to you to delete .ocu files which are no longer in use.

19.3.3 Locking a control unit for edition

Locking a control unit for edition means that you acquire a lock on the control unit. The lock guarantees that nobody else can change the contents of the control unit.

To edit a control unit:

Right-click the control unit and select the **Control unit>Edit** entry

- Make sure the control unit you want to edit is free, that is, with a (Green, Dim) status.
- Objectiver changes the status of the control unit to (Green, Plain).
- If other users are editing the same project as you, Objectiver notifies then about the fact that another user (you!) has started editing the control unit by changing the status of the control unit in their environment: the status changes from (Green, Dim) to (Orange, Dim).
- You can also learn who is locking a control unit by looking at the tip appearing when you put the mouse on the control unit entry in the explorer. The tip has the

following shape:

Control Unit <Control unit name> in <Package or control Unit name> edited by <user>

Once you have locked a control unit, you can change its contents (see Section 19.3.6).

When you have finished the edition session, you can do one of the following:

- save your changes and release the lock (see Section 19.3.4)
- save the project (you need the lock on Concept Index) (see Section 19.3.8 and Section 19.3.9)
- quit the project without saving it (in that case, all your changes made since your last save operation will be lost) (see Section 19.3.10).

Notes

1. It can take a few seconds to be notified about a status change in a control unit.
2. Control units that have the (Orange, Dim) or (Purple, Dim) status can be locked by you as well ... This feature shall only be used to recover from crashes or to acquire locks from people having forgotten to release them. Acquiring locks in those conditions implies actually to “steal” the lock to another user and shall only be used if no other way can be found.

19.3.4 Committing a control unit

Committing a control unit means saving the changes made on all the edited control units and release the lock on this one.

A control unit can only be committed if you have a lock on it (see Section 19.3.3).

To commit a control unit:Right-click the control unit you want to commit and select **Control unit>Commit**

- Objectiver then saves **all** the edited control units and releases the lock on **only** that control unit.
- Objectiver informs you that you have released the lock on the control unit by changing its status from (Green, Plain) to (Green, Dim).
- If other users are editing the same project as you, Objectiver notifies them about the fact that a new version of the control units is available by changing the status of the control units in their environment: the status changes from (Orange, Dim) to (Purple, Dim).

Note

It can take a few seconds to be notified about a status change in a control unit.

19.3.5 Refreshing control units

Refreshing control units allows you to get access to the latest available version of control units that have been edited by another user.

A control unit can only be refreshed if its current status is (Purple, Dim).

Click the **Refresh** button in the toolbar to refresh control units. Objectiver will save all your changes to the disk, prior to refresh the model.

- Objectiver then replaces in memory the current version of **all** the control units to be refreshed it has with the ones it gets by reading the .ocu files associated with the control units.
- Objectiver will also inform the user about the status of the refreshed control units. Two cases are possible:
  - the control units are still locked by another user. In this case, Objectiver changes the control units status from (Purple, Dim) to (Orange, Dim)
  - the control units are no longer locked by another user. In this case, Objectiver changes the control units status from (Purple, Dim) to (Green, Dim).

19.3.6 Editing the contents of control units

Once you have acquired a lock on a control unit (see Section 19.3.3), you can change its contents

- by creating, adding, moving or deleting documents in the control unit
- by changing the documents contents.

In particular,

- you can create a new package only at a location belonging to a control unit you have locked
- you can delete a package only if the package belongs to a control unit you have locked.

Section 19.1.3 explains the rules in force to display and modify concepts in documents you have locked.

19.3.7 Opening a project

This section aims at describing precisely which is the status of each package and control unit when a user loads a project.

When you open a project, Objectiver sets the package and control unit status as follows:

1. All control units (and the Concept Index) are set to (Green, Dim) or (Orange, Dim) if somebody else is editing it: you do not own the lock on the control units.
2. All packages are set recursively to (Blue, Dim).
3. However the status of the Concept Index is set to (Green, Plain) if the project doesn't contain any control unit and nobody else is editing it. This is true in particular if you are the first user to open the project. In that case, each package in the Concept Index is set to (Blue, Plain) recursively all along the hierarchy of package-subpackages.

19.3.8 Saving a project

When a user invokes the **File>Save Project** command, the tool proceeds as follows:

1. Objectiver saves the .ob file
2. Objectiver saves all the control units on which the user owns the lock
3. Objectiver does not save the control units on which the user does not own the lock
4. The operation does not impact locks: all users keep all the locks they have.

Note: The **File>Save Project** command is enabled only if the user has locked the Concept Index (that allows the tool to write the .ob file).

19.3.9 Saving a project in another file

The **File>Save Project As...** command provides the user with the opportunity of creating a completely new copy of the project.

When Objectiver writes the copy of the project, all the control units are converted into packages in the copy. This feature provides the user with a quick way to remove all control units from a project.

19.3.10 Ending a session

There are many ways for ending a project session in Objectiver:

- by invoking **File>Quit**
- by invoking **File>Close project**
- by invoking **File>Open** or **File>Reopen**
- by invoking **File>New project**

All those operations ask the user to confirm the command if changes to the project have not been saved.

If you decide to save those changes, the tool invokes the **File>Save project** command (see Section 19.3.8).

In both cases, the tool arrives up to a point where it has to quit. If the user has locks on control units, those locks are released and the other users are notified as follows:

1. If you have a control unit with the (Orange, Dim) status and if the user who locks this control unit quits, Objectiver changes the status of this control unit from (Orange, Dim) to (Green, Dim).
2. Control units with the (Purple, Dim) status are unaffected by the fact that other users quit.

19.4 How to use control units

The teamwork facility of Objectiver is powerful but can become tricky if not managed in an appropriate way.

Here follows advices based on past experiments you might consider interesting if you plan to use those facilities for the first time.

1. Choose a person responsible for defining the control units, even if any user can define control units in the current release. The project leader is a good candidate for it.
2. Keep it simple.
3. Define a teamwork strategy. There is no strategy that works for any project in any company. Here follows a strategy which has been used successfully in the past.

At the beginning of the project

1. Define a package Model that will contain the final, shared model. Structure the package according to your needs by defining a hierarchy of empty packages.
2. Export the Model package in a ‘Model.opk’ file (see Section 7.3.2).

3. Define a package ‘Workspace username’ to contain the workspace dedicated to a specific user. Import the ‘Model.opk’ file to replicate your standard model structure in the workspace.
4. Define as many workspace packages as they are users having to work together.
5. Convert the Model package and all the workspace packages into control units.

Figure 70 shows an example of project structure containing one shared model and two worskspaces:

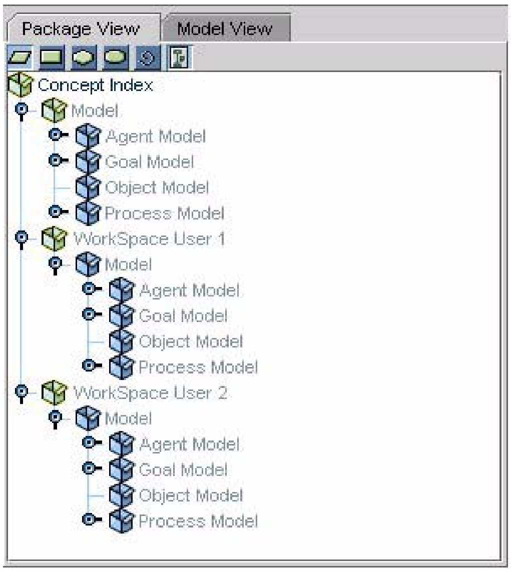


FIGURE 70. Project structure with shared model and worskspaces

In large organisations, each workspace can be shared by different users. Control units can be defined lower in the hierarchy to identify the contribution of each user in the workspace if needed.

Similarly, for the shared model, it is possible to assign parts of the model to different users. For instance, the Object model could be put under the responsibility of one user and the Goal model under the responsibility of another one.

During the project

Each user works inside his/her workspace. He/she is strongly encouraged to refer as often and as early as possible the shared model in which he/she has full reference access, but no permission for changing it. It is therefore a good idea to start building a first shared model with the set of concepts that will be used by most of the users at the very beginning of the project.

At synchronization steps

At regular time intervals, the user responsible for the shared model reviews the contribution of each user in his/her workspace. He/she can decide to move concepts from the user workspace to the shared model.

Operations like Clone Diagram, Copy/Paste of concepts between diagrams and the Bring related concepts on packages can be used to move the concepts easily. Concepts can also be merged during this step in order to remove duplicates.

PART H Appendices

20.0 Appendix A Additional information

20.1 Memory usage status

Starting with Objectiver, the memory status bar is shown in blue color in normal conditions. If it turns orange or red, you are under heavy memory usage conditions. Try clicking the garbage icon to collect as much memory as possible, then save your specification. You should now quit and restart Objectiver to resume your work in better memory conditions.

20.2 System Requirements

20.2.1 Required RAM

512 Mbytes is the minimum recommended RAM size for using the Objectiver. 1 Gbytes RAM will be more confortable for large requirements models and documents. Running Objectiver with less memory might cause disk swapping which will result in a drastic loss of performance.

20.2.2 Java VM Configuration Updated

The default maximum java VM heap size has been set to 512 MB for running Objectiver. To change this value, proceed as follows:

1. Go the directory in which Objectiver is installed (typically Program Files/Objectiver3.0).
2. Edit the Objectiver.l4j file with a text editor (notepad, ...)
3. Change the line containing the flag -Xmx and specify the size of the heap you want; for instance, -Xmx1000M allows the tool to require 1 Gb max for the heap size.

You need to restart Objectiver for this change to take effect.

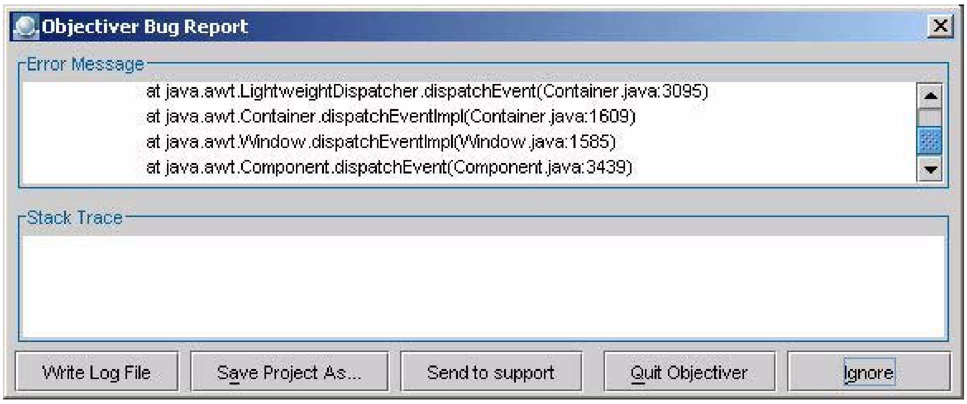
20.2.3 SVG plugin

To visualize diagrams in the web generated version of your specifications, the **SVG plugin** must be installed and configured to work with your browser. To get and install the plugin, please go Adobe’s web site at the following address: http://www.adobe.com/svg and follow the instructions.

20.3 Bug Report

If a bug report appears, click **Ignore** to skip the error and return to Objectiver. Click **Save Project As...** first to be sure not to lose data.

If you want to send the bug report to support, if you use a Proxy server, you will have to fill in the connection preferences (see Figure 23).



```
at java.awt.LightweightDispatcher.dispatchEvent(Container.java:3095)
at java.awt.Container.dispatchEventImpl(Container.java:1609)
at java.awt.Window.dispatchEventImpl(Window.java:1585)
at java.awt.Component.dispatchEvent(Component.java:3439)
```

FIGURE 71. Bug Report Window