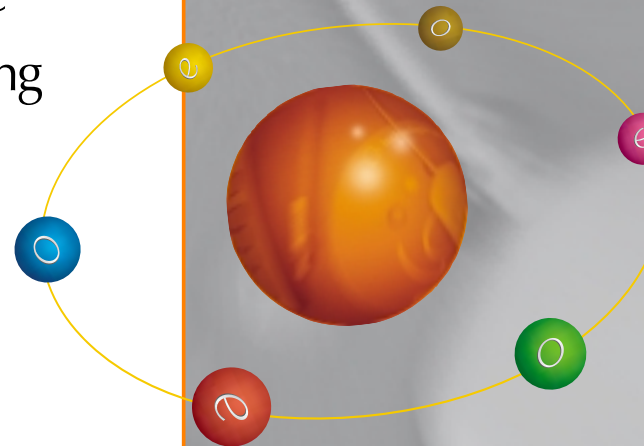


Objecteering/UML

Objecteering/Introduction User Guide
Objecteering/Administrating Objecteering
Sites User Guide

Version 5.2.2



Objecteering

Software

www.objecteering.com

Taking object development one step further

Information in this document is subject to change without notice and does not represent a commitment on the part of Objecteering Software. The software described in this document is furnished under a license agreement. The software may be used or copied only in accordance with the terms of the agreement. It is against the law to copy the software on any medium except as specifically allowed in the license agreement. The purchaser may make one copy of the software for backup purposes. No part of this manual may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying, recording, or information storage and retrieval systems, for any purpose other than the purchaser's personal use, without the express written consent of Objecteering Software.

© 2003 Objecteering Software

Objecteering/UML version 5.2.2 - CODOBJ 001/002

Objecteering/UML is a registered trademark of Objecteering Software.

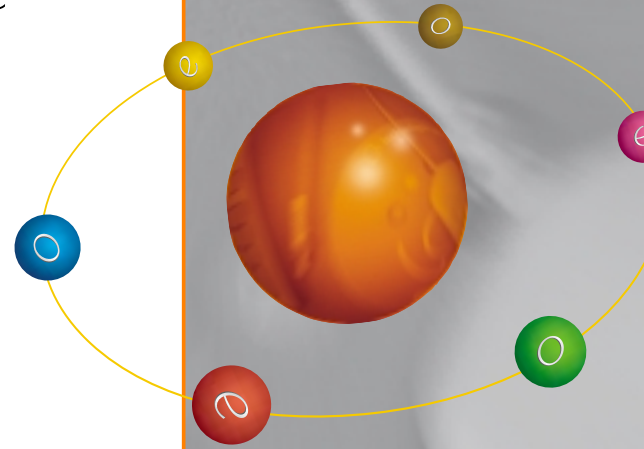
This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>).

UML and OMG are registered trademarks of the Object Management Group. Rational ClearCase is a registered trademark of Rational Software. CM Synergy is a registered trademark of Telelogic. PVCS Version Manager is a registered trademark of Merant. Visual SourceSafe is a registered trademark of Microsoft. All other company or product names are trademarks or registered trademarks of their respective owners.

Objecteering/UML

Objecteering/Introduction User Guide

Version 5.2.2



Objecteering

Software

www.objecteering.com

Taking object development one step further

Contents

| | |
|---|------|
| Chapter 1: Presentation | |
| Introduction | 1-3 |
| Objectteering/UML user guides | 1-9 |
| Chapter 2: Installing Objectteering/UML | |
| Overview of Objectteering/UML installation procedures | 2-3 |
| Installing in Windows | 2-5 |
| Stand alone installation in Windows | 2-6 |
| Server installation in Windows | 2-24 |
| Client installation in Windows | 2-40 |
| Objectteering/UML tools in Windows | 2-49 |
| Installing in UNIX..... | 2-50 |
| Client-Server installation in UNIX | 2-59 |
| The LM_LICENSE_FILE environment variable | 2-62 |
| Installing licenses..... | 2-64 |
| Using lmttools..... | 2-65 |
| Modifying your Objectteering/UML installation | 2-67 |

| | |
|--|-------|
| Chapter 3: First Steps | |
| Preparation..... | 3-3 |
| Creating a new UML modeling project | 3-4 |
| Changing UML modeling project | 3-8 |
| Selecting modules in the current UML modeling project..... | 3-10 |
| Working in the explorer | 3-13 |
| Creating an operation | 3-18 |
| Creating a diagram | 3-23 |
| Consistency and help mechanisms | 3-29 |
| Assisted data entry: Example of link creation..... | 3-32 |
| Completing our first model..... | 3-40 |
| Organization..... | 3-49 |
| Showing elements in a diagram | 3-57 |
| Creating a sequence diagram | 3-64 |
| Creating a state diagram | 3-71 |
| Notes and tagged values | 3-80 |
| Implementing modules..... | 3-90 |
| Setting module parameters..... | 3-94 |
| Generating documentation | 3-97 |
| Creating a document work product | 3-100 |
| Generating code | 3-105 |
| Conclusion | 3-115 |

Index

Chapter 1: Presentation

Introduction

Objecteering/UML drives your applications

Welcome to *Objecteering/UML* and congratulations on choosing one of the best UML CASE tools in the world.

Objecteering/UML is a sophisticated object modeling workshop, providing impressive support for UML modeling. It guarantees model consistency and has considerable capacity for generating code.

Objecteering/UML is distributed in three different editions:

- ◆ *Personal*: Objecteering/UML Personal is destined for individual developers and provides advanced documentation generation and modeling services. It cannot, however, be used to exchange models, and does not provide group work facilities. No code generation facilities are provided. Furthermore, it does not include the powerful parameterization functions provided in the "*Enterprise*" version.
- ◆ *Professional*: Objecteering/UML Professional is dedicated to people working alone on a single UML modeling project and provides fully-fledged model-driven CASE tool development. It includes all the modeling and code and documentation generation functions of the Enterprise Edition, and allows fine-grain import/export operations and updates between projects, but does not support multi-user team work, and is delivered with a single user license.
- ◆ *Enterprise*: Objecteering/UML Enterprise is a tool destined for team application development. It provides a universal model element identification mechanism and inter-model exchange services, as well as group work functions. This version also allows you to parameterize Objecteering/UML, both at UML level and at code/documentation generation, model transformation and model request or operation levels, through the separate *Objecteering/UML Profile Builder* module.

Objecteering/UML is multi-platform (PC Windows and UNIX).

Objecteering/UML Enterprise Edition is an expandable tool, which may be configured by the user. Objecteering Software offers a permanently increasing set of modules, which can be integrated into the tool itself. These modules, such as *Objecteering/Java*, *Objecteering/C++* or *Objecteering/Design Patterns for Java/C++*, can be supplied and added at any time to any Objecteering/UML Enterprise site.

The Objectteering/UML Profile Builder tool

Objectteering/UML Profile Builder, which allows the user to create new models himself and to adapt existing models, is a unique tool, in which the powerful notion of the "UML Profile" is dealt with. It is provided in the form of an additional module in the "Enterprise" and "Open" editions of Objectteering/UML. The *Objectteering/UML Profile Builder* module allows you to define model transformation rules, and consistency checking rules, using Profile editors and its powerful J language. It is also used to define model requests, specific code generation and customization of the generators supplied by Objectteering/UML. Note types or tagged values types, specific menu items (commands), document templates, and so on, can also be defined.

All these functions mean that Objectteering/UML provides considerable customization possibilities and can thus handle and automate specific processes for each client site.

The generation modules available with Objectteering/UML (C++, Java, RDBMS, etc.) are examples of modules developed with the *Objectteering/UML Profile Builder* tool.

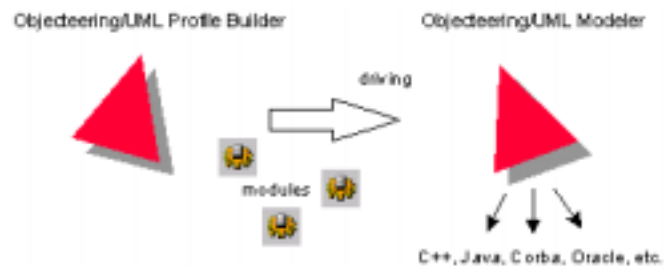


Figure 1-1. The Objectteering/UML Profile Builder is used to drive the Objectteering/UML Modeler tool

Model driven development

With Objectteering/UML, you can work in the modeling tool at all times during development. It automatically generates the usual development work products:

- ◆ documentation
- ◆ C++ code, Java code or code in other languages

The user selects his modules according to the desired application type (client/server, C++, Oracle, documentation, Java etc.) and then builds his UML modeling projects, annotates them with tagged values, enters associated notes and automatically produces his application.

All information, including a target's specific characteristics (C++ code, SQL instructions, etc.), is handled in the Objectteering/UML repository. The user constantly works in Objectteering/UML, as it guarantees consistency between the model, generated code and documentation at all times.

Version history

| Version ... | New functions ... |
|---------------------|---|
| Objectteering 3 | Created in 1992. Many applications have been developed with this workshop, based on an entirely generated code , and on the Class-Relation model (UNIX). |
| Objectteering 4 | Has a completely different architecture. It is multi-formalism (OMT, UML, Class-Relation), and multi-platform (UNIX and PC Windows). |
| Objectteering 4.2.1 | Introduces the "Use Case" model (use case), the document templates editor, HTML documentation generation and configuration management modules. |
| Objectteering 4.2.2 | Offers a new and very high-performance storage system (up to ten times more compact and up to twice as much RAM) which increases performance. It also offers a very secure and sophisticated system of database administration, whilst at the same time improving its ergonomoy. |
| Objectteering 4.3 | Presents significant improvements in the ergonomics of the graphical user interface. It also supports version 1.3 of the UML and incorporates several new developments, such as the integration of MFC package types in the C++ module, and the externalization of all modules from the database. |
| Objectteering 4.3.1 | Allows certain consistency checks to be deactivated, for freer use of models. Adds the read only mode used in group work, which can be applied to model elements. Database creation from the Objectteering home page. Migration of version 4.3 databases into version 4.3.1. Addition of activity diagrams. |
| Objectteering 4.3.2 | Introduces improvements in the structure of modeling features, with new presentation of module parameter configuration and a feature allowing you to change project, metaproject or document template without quitting the tool. Groups all administration functions together in the baseadm tool. |

| Version ... | New functions ... |
|-------------------------|--|
| Objectteering/UML 5.1 | Introduces major developments in the <i>Objectteering/Java</i> module, which now includes a properties editor to facilitate the modeling of Java applications. The <i>Objectteering/EJB</i> and <i>Objectteering.JUnit</i> modules are also now available. XProgramming is now supported. The Objectteering/UML GUI has been overhauled: dockable windows, a properties editor and the evolution of the model import window. Significant improvements have been made to the ergonomics of the tool: Objectteering/UML is now easier to launch, and UML modeling and profiling projects can be launched by double-clicking in a file browser. The procedure for upgrading projects developed using earlier versions of the tool has also been simplified. The user work context can now be saved. Other improvements include the possibility of creating macros and the simplification of the notion of generation work products. |
| Objectteering/UML 5.1.1 | Presents further improvements in the ergonomics of the Objectteering/UML CASE tool, making it even easier to use. The <i>Objectteering/C++</i> module now includes a properties editor, used to enter or modify certain information for C++ generation. The performance of the <i>Objectteering/Multi-user</i> module has been significantly improved, with gains in performance of almost 70%. The <i>Objectteering/XML</i> module now handles sequence and object diagrams, and works with both version 1.3 and version 1.4 of the OMG UML standard. The <i>Objectteering/Visual Basic</i> module is now available. |
| Objectteering/UML 5.2 | Features easier client/server deployment, as well as the possibility of installing Objectteering/UML in <i>C:Program files</i> (blanks now handled). Objectteering/UML is now also available for Linux. A search engine is provided for use with the Objectteering/UML on-line help. The <i>Objectteering/Documentation</i> module now provides table management, and improved handling of links. The <i>Objectteering/Java</i> module now provides parameterization of accessors, attribute and association visibility. <i>Objectteering/SQL Designer</i> is now integrated as standard. |

| Version ... | New functions ... |
|----------------------------|--|
| Objectteering/UML 5.2.1 | <p>Presents Objectteering/UML for the Linux platform, as well as making it possible to update client installations to either lightweight client or heavyweight client mode. Users of the <i>Objectteering/Java</i> module can take advantage of further improvements made to the Java properties editor, which now provides easy access to Java documentation generation on packages, and update and edit operations on classes. Forte markers, used in Forte to prevent the modification of code generated by the Objectteering/Java module, can now be generated, and useful See Also links added to your Java documentation using Javadoc @see markers.</p> <p>Further improvements have also been made to deployment diagrams.</p> |

Objecteering/UML user guides

Copyright: Please read this on-line help copyright information carefully.

Objecteering/Introduction User Guide

- ◆ *Installation procedure:* This chapter provides details on how to install the various different editions of Objecteering/UML, including information on multi-user, multi-platform configurations.
- ◆ *First Steps:* We recommend that every user take this trip into the world of Objecteering/UML. You will become an expert in about an hour and a half.
- ◆ *Objecteering/Administrating Objecteering Sites User Guide:* Data management and repair (models, UML modeling projects, UML profiling projects, modules).

Objecteering/UML Modeler

- ◆ *Objecteering/UML Modeler User Guide:* Graphic editors, the Objecteering/UML explorer, the properties editor and the console.
- ◆ *Objecteering/Model Dialog Boxes User Guide:* Definition of every field in the dialog boxes.

Objecteering/Process Wizard User Guide: Assistance during process development, UML modeling project creation, modeling activities and model completion.

Objecteering/UML Teamwork User Guide: (Objecteering/UML Enterprise Edition only) Management of multi-user work and development spaces. Information on the *Objecteering/Multi-user*, *Objecteering/SCC* and *Objecteering/ClearCase* modules.

Objecteering/XMI User Guide: Used to generate and re-read XMI files.

Objectteering/Documentation

- ◆ *Objectteering/Documentation User Guide*: Production of model driven documentation
- ◆ *Objectteering/Document Template Editor User Guide*: Definition of the form and content of documentation.

Objectteering/C++ User Guide: (Not available in the Objectteering/UML Personal Edition) Model driven generation of C++ code and Makefiles

Objectteering/C++ Reverse Engineering User Guide: (Not available in the Objectteering/UML Personal Edition) Based on the "SNIFF" environment, this facility is used for reversing legacy C++ code into Objectteering/UML.

Objectteering/Java User Guide: (Not available in the Objectteering/UML Personal Edition) Model driven generation of Java code and Makefiles, and automation of the most frequent patterns in Java.

Objectteering/Design Patterns for Java/C++ User Guide: (Not available in the Objectteering/UML Personal Edition) Automation of the most frequently used general design patterns (GOF).

Objectteering/SQL Designer User Guide: (Not available in the Objectteering/UML Personal Edition) Model driven generation of DDL code.

Objectteering/Metrics User Guide: (Not available in the Objectteering/UML Personal Edition) Implementation of a set of metrics, which are used to evaluate the quality of the models produced.

Objectteering/CORBA User Guide: (Not available in the Objectteering/UML Personal Edition) Model driven generation of IDL code.

Objectteering/UML Profile Builder User Guide: (Objectteering/UML Enterprise Edition only) defining UML profiles used to customize UML and Objectteering/UML, editing modules, stereotypes, etc.

Objecteering/The J language development user guide

- ◆ *Objecteering/J Libraries User Guide*: (Objecteering/UML Enterprise edition only) set of libraries used to build modules, handle diagrams, etc., using the J language.
- ◆ *Objecteering/Metamodel User Guide*: (Objecteering/UML Enterprise edition only) close to the UML 1.4 metamodel, the Objecteering/UML metamodel provides all the accessible and manageable information with J and the UML Profile Builder tool.

Objecteering/J Language User Guide: (Objecteering/UML Enterprise edition only) Objecteering/UML parameterization language working on the metamodel with a Java-like syntax.

Chapter 2: Installing Objecteering/UML

Overview of Objectteering/UML installation procedures

Delivery of Objectteering/UML

Objectteering/UML is delivered with the following elements:

- ◆ an Objectteering/UML installation CD-ROM. It contains all the necessary executables, the on-line help (HTML for UNIX and Windows) and examples. The "*setup*" executable is used to launch the installation.
- ◆ documentation on *Objectteering/UML Modeler* and on different modules.

Your site has a unique identification number, labelled "site number", which universally identifies each model element.

If you are using Objectteering/UML for the first time, we recommend that you try out our "*First Steps*" workshop (chapter 3 of this user guide).

Different installation modes

- ◆ Installation over a previous version
- ◆ Single station configuration
- ◆ Server configuration
- ◆ Client configuration, for both lightweight clients and heavyweight clients

Further information

Two files are available on the Objectteering/UML installation CD-ROM:

- ◆ "*readme.txt*": this file contains information which does not figure in the Objectteering/UML user guides, as well as information which must be read before beginning the installation procedure.
- ◆ "*release.txt*": this file contains corrections and evolutions in the different versions of Objectteering/UML.

After installation, these two files are available in your installation's "*help*" directory.

Objectteering/UML license file

In order to install the Professional Edition or the Enterprise Edition of Objectteering/UML, you must have the appropriate license file. This file will have been provided by the Objectteering Software technical support team.

To contact the Objectteering/UML support team, send a mail to *support@objectteering.com*.

No license file is needed to install the Personal Edition of Objectteering/UML.

FlexIm tool

The "*Imtools*" flexIm tool is delivered as standard with the Objectteering/UML CASE tool. This program groups together a set of tools used to manage license installation on the network.

Objectteering/UML version

When installing Objectteering/UML in client/server mode, both the client and server posts must have the same version of Objectteering/UML installed.

Installing in Windows

Prerequisites

Objectteering/UML 5.2.2 can be installed in Windows 95, Windows 98, Windows 2000 or Windows NT.

Here are the product's general installation characteristics:

- ◆ disk resources: 200 to 350 Mo, depending on modules selected
- ◆ minimum RAM: 128 Mo, 256 recommended
- ◆ minimum processor power: PentiumII 500MHz
- ◆ CD-ROM drive
- ◆ administrator licenses (Windows NT)

Re-installing

When a new version of Objectteering/UML is installed over a former installation, the previous version is overwritten, but modeling, profiling and document template projects are retained.

Where version 5.2.2 is installed over an earlier version, UML modeling, profiling and document template projects can be upgraded, by simply double-clicking on the relevant project file in the Windows explorer or selecting the project in question in the "*Open a UML modeling project*" or "*Open a UML profiling project*" windows. This automatically launches the migration procedure. For further details, please refer to the "*Receiving or upgrading UML modeling projects*" section in chapter 3 of the *Objectteering/UML Modeler* user guide.

Stand alone installation in Windows

In a stand alone installation in Windows, UML modeling and profiling projects, and document template projects are accessed directly by Objectteering/UML without going through the site server. Only one user can connect to a given project at a time. Projects are called "*local*" and the server is always called "*LOCAL*".

Installation procedure

Before launching the installation procedure, you must be in possession of a license file provided by the license administrator (requested from license@objectteering.com) or from our web site (www.objectteering.com).

Note: The Personal Edition of Objectteering/UML comes with a built-in license file.

To install Objectteering/UML in the Windows environment, you must:

- 1 - Insert the Objectteering/UML CD-ROM.
- 2 - Select and click on "*setup.exe*".

The Objectteering/UML setup procedure is then automatically launched. Let yourself be guided by the questions asked during installation.

If an earlier version of Objectteering/UML has already been installed on your site, a dialog box of the following type will appear (Figure 2-1):

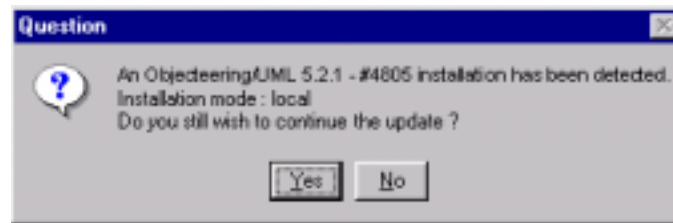


Figure 2-1. Question when installing Objectteering/UML over an earlier version of the tool

In this case, the installation procedure will run using the parameters set for the previous installation. If this is not the case, the installation path will be requested.

The software license agreement

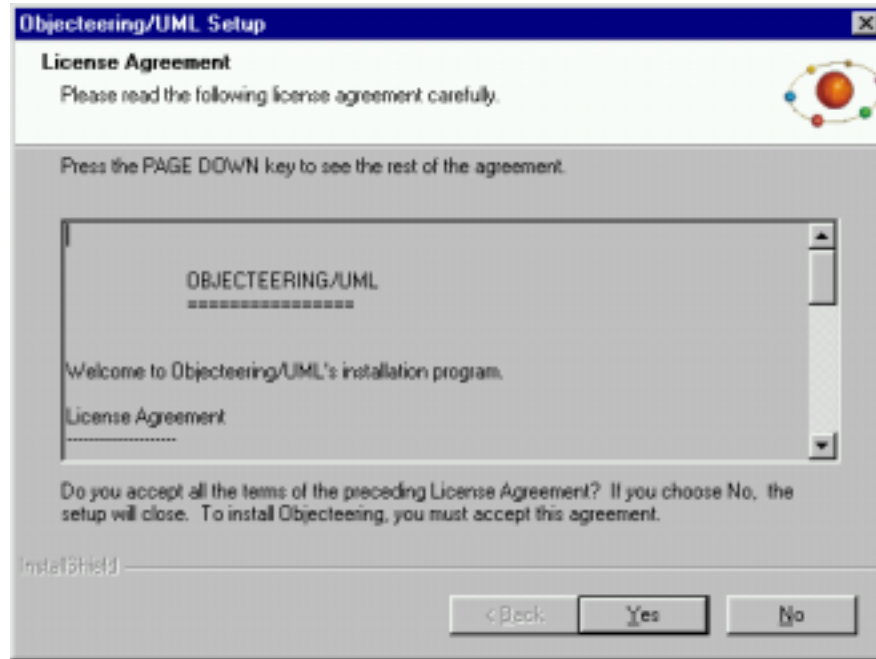


Figure 2-2. The software license agreement page

Read the information given on this software license agreement page, and then click on the "Yes" button.

Selecting the installation mode

Select the mode in which you wish to install Objecteering/UML - "Server", "Heavyweight client", "Lightweight client" or "Stand-alone" (as shown in Figure 2-3).

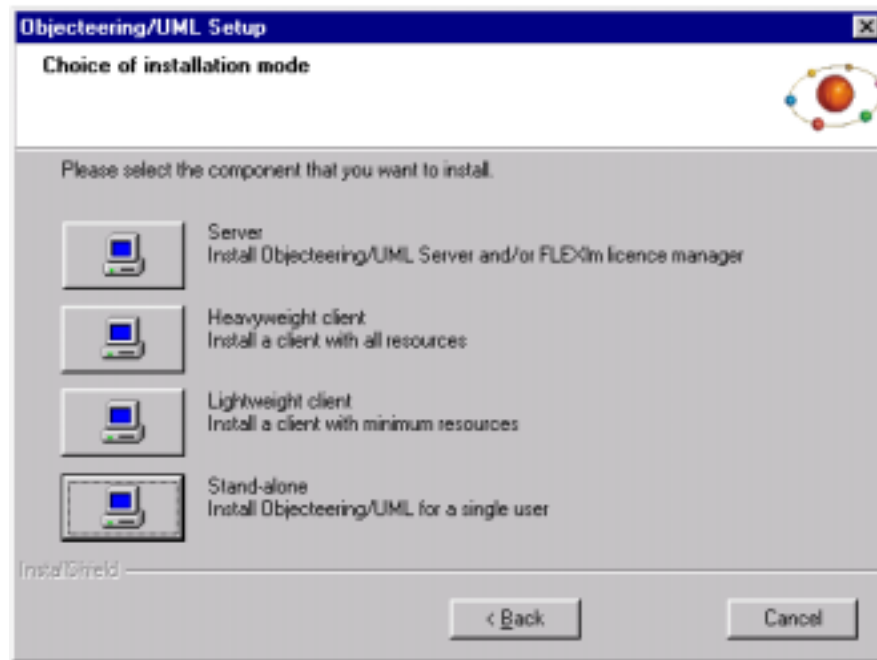


Figure 2-3. Selecting the installation mode

Select the stand-alone mode, and click on the "Next" button.

Chapter 2: Installing Objecteering/UML

If you are installing either the Professional Edition or the Enterprise Edition, a question concerning the license file then appears (as shown in Figure 2-4).

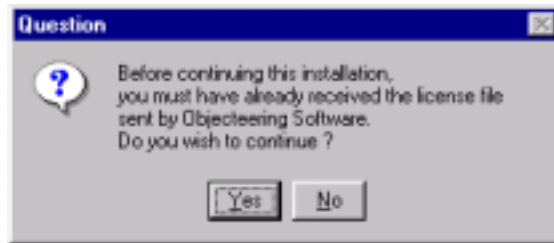


Figure 2-4. Question on licenses

If you do not have a license file, click on the "No" button. The installation procedure will then be stopped.

If you have a license file, click on the "Yes" button. The installation procedure will then continue, and the screen shown in Figure 2-5 will be used to indicate the positioning of your license file.

Note: The Personal Edition of Objecteering/UML comes with a built-in license file, which means that the screen shown in Figure 2-4 will not appear when you are installing.

Note: This window only appears after selecting either the "Stand alone" or "Server" installation modes.

Selecting your license file

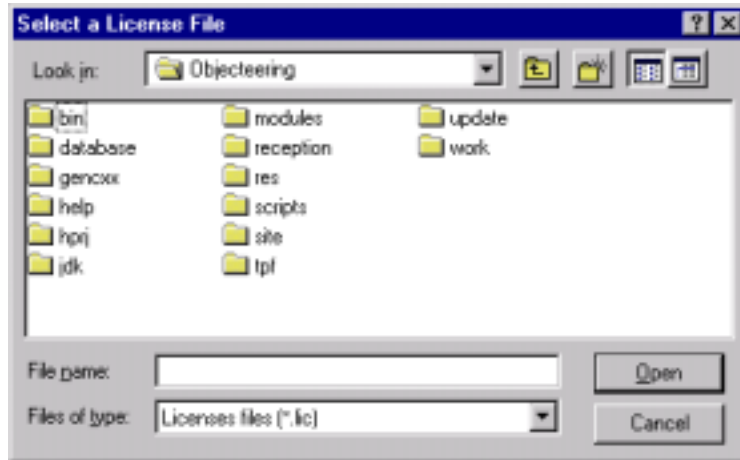


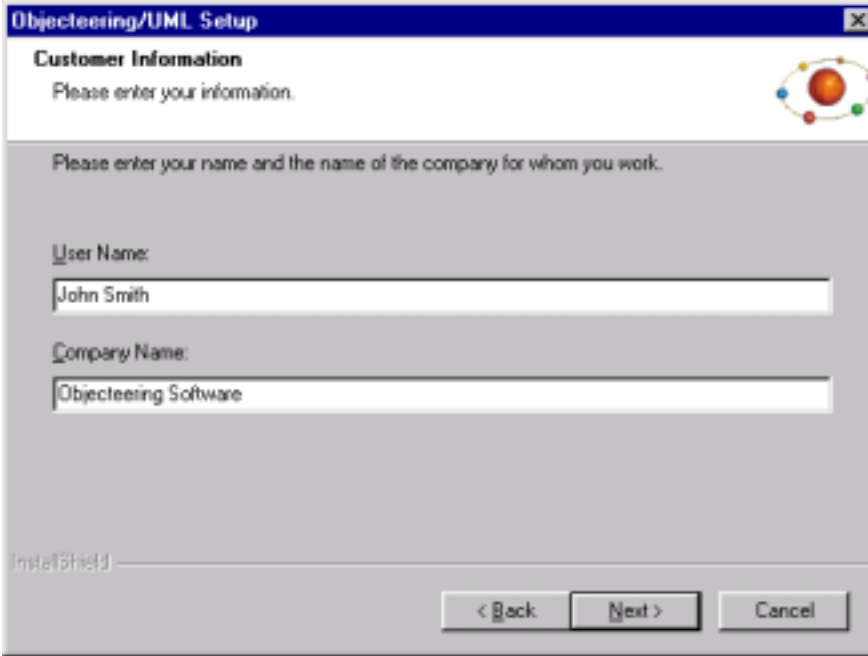
Figure 2-5. The license selection window

Browse in this explorer and select the license file sent to you by Objecteering Software.

Warning: Do not use shortcuts to browse!

Entering user information

This screen is used to enter certain information about you, the user (Figure 2-6).



The screenshot shows a Windows-style dialog box titled "Objectteering/UML Setup". The main heading is "Customer Information" with the instruction "Please enter your information." and a small logo of a globe with colored dots. Below this, it says "Please enter your name and the name of the company for whom you work." There are two text input fields: "User Name:" containing "John Smith" and "Company Name:" containing "Objectteering Software". At the bottom left, it says "InstallShield". At the bottom right, there are three buttons: "< Back", "Next >", and "Cancel".

Figure 2-6. Entering user information

Enter your name and your company name, and then click on the "Next" button to continue.

Choosing the destination location

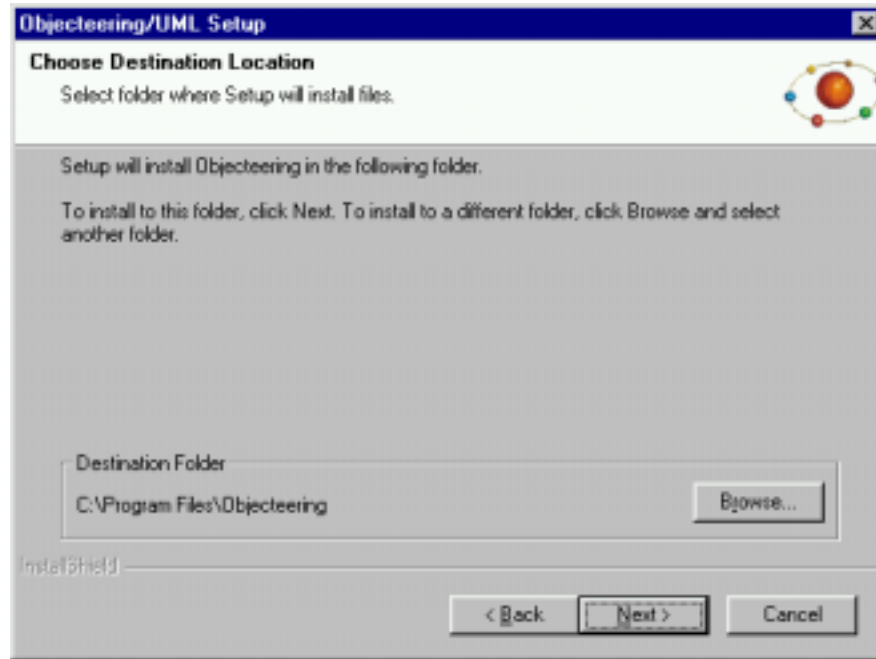


Figure 2-7. Choosing the destination location

This window only appears if you do not already have a version of Objecteering/UML installed on your machine. If a version of Objecteering/UML already exists on your computer, the screen shown in Figure 2-1 will appear.

Choosing the setup type

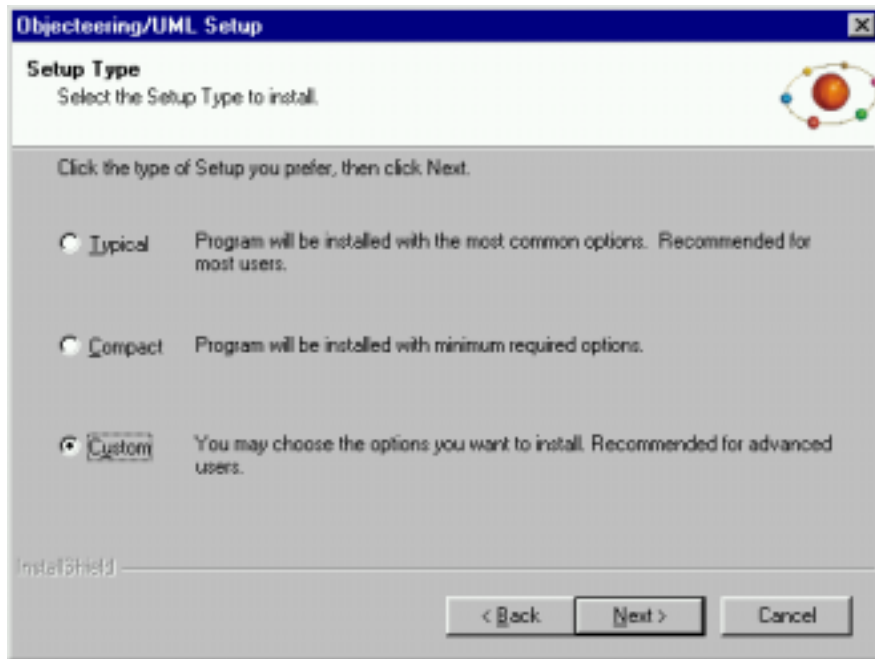


Figure 2-8. Selecting the setup type

Select the installation type ("*Typical*", "*Compact*" or "*Custom*") and click on the "*Next*" button.

If you select the "*Typical*" installation, a question box then appears, asking you whether or not you wish to install the Objecteering/UML on-line documentation (Figure 2-9).

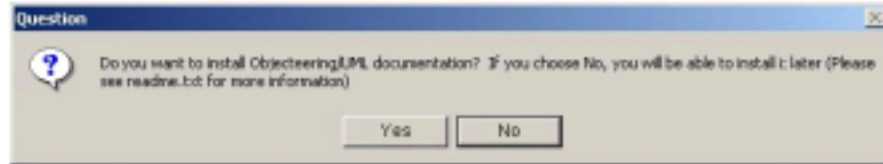


Figure 2-9. Choosing whether or not you want to install Objecteering/UML on-line documentation

Simply click on "Yes" or "No" and proceed with your installation.

If you select the "*Custom*" installation type, the "*Select Components*" window then appears (Figure 2-10).

Selecting components

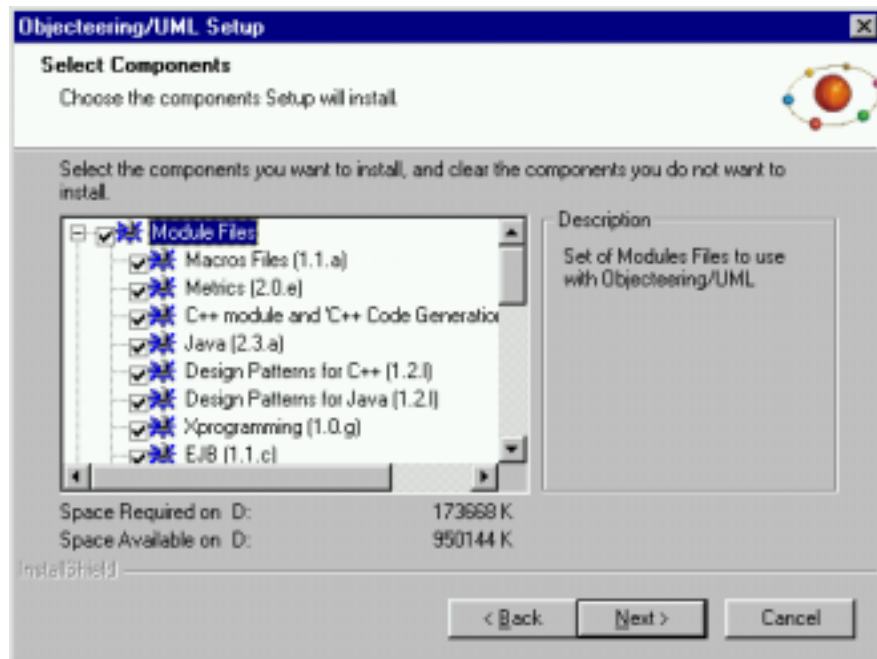


Figure 2-10. Selecting components

Select the components you wish to install by checking the relevant tick boxes, and then click on the "Next" button to continue.

Note: This window is automatically displayed during reinstallation of Objecteering/UML.

To define those modules you wish to install, click on "Module Files" and then select the modules you wish to install. Confirm, by clicking on the "Next" button.

Migrating databases

After selecting components, the "Selecting database for migration" window appears.

If your site contains databases which must be migrated into the 5.2.2 version of Objecteering/UML, you should simply select them in this window, by checking the relevant tickboxes (as shown in Figure 2-11).

The databases which appear in this window are those which are declared on your site, and which are visible from the post on which you are running the installation procedure.

If there are no databases to be migrated, the "Selecting database for migration" window is empty.

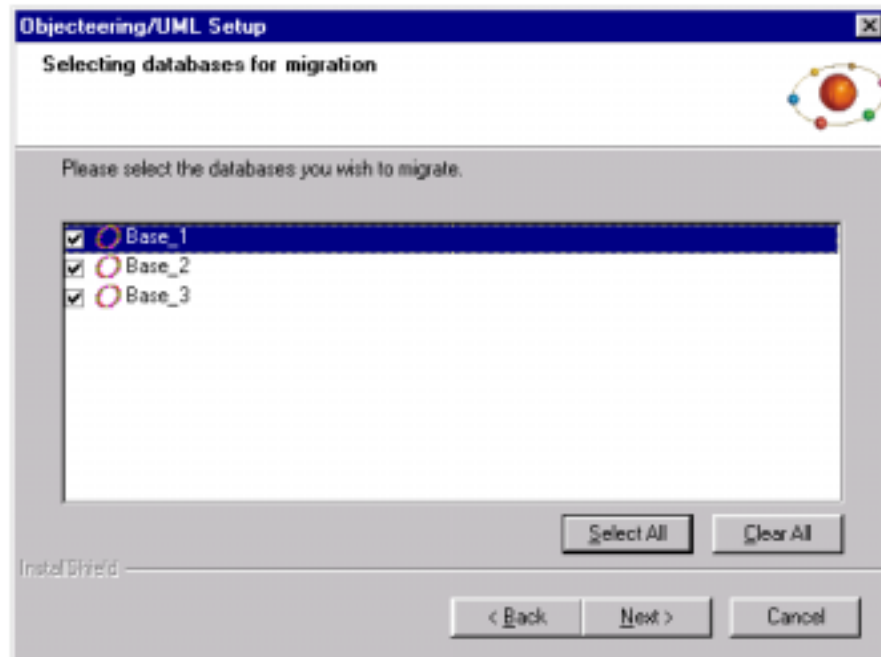


Figure 2-11. Selecting databases to be migrated



Chapter 2: Installing Objectteering/UML

Simply select the databases you wish to migrate, and click on "Next" to continue. To select all the databases in the list, click on the "Select all" button. To unselect all selected databases, click on the "Clear all" button.

Note: If you do not migrate databases at this stage, when you open the databases in question, their migration is automatically proposed (for more information, please refer to the "Receiving and upgrading UML modeling projects" section in chapter 3 of the *Objectteering/UML Modeler* user guide).

Delivering and migrating modules

After selecting databases to be migrated, the "Selecting modules to deliver and migrate" window appears.

This window contains a list of modules either already installed on your site and now available in a later version, or not installed on your site and available with the version of Objecteering/UML you are installing. Modules presented with the  icon are modules included in the packaging of Objecteering/UML and which can be delivered to your site, whilst modules presented with the  icon are those which already exist on your site in an earlier version and which can be migrated.

To select the modules you wish to deliver and/or migrate, you should simply check the relevant tickboxes (as shown in Figure 2-12).

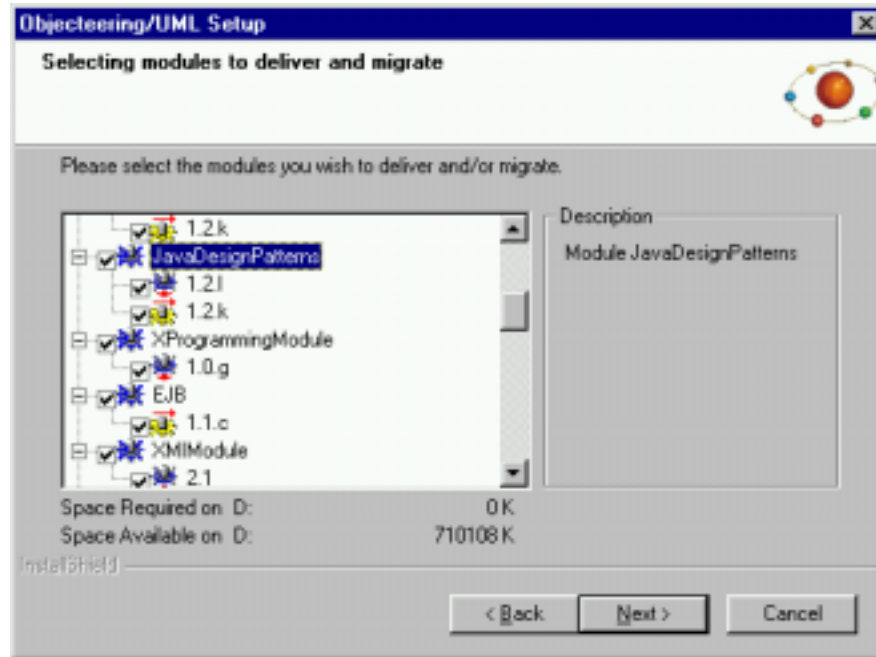


Figure 2-12. Selecting modules to deliver and migrate

Selecting the program folder

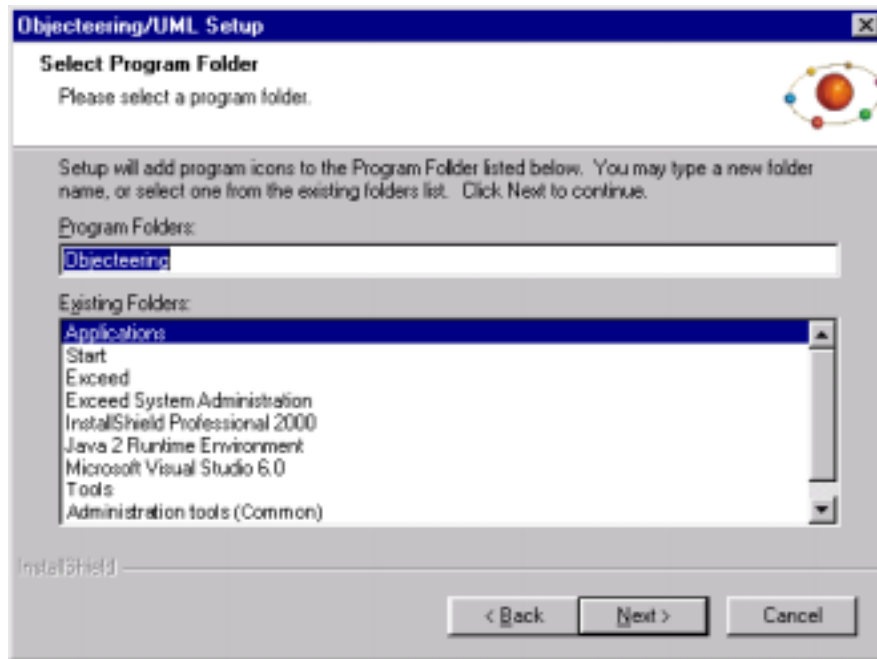


Figure 2-13. Selection of the program file

Select the "Program Folder" and click on the "Next" button.

Starting to copy files

Once you have completed all the above dialog boxes, the following window will appear (Figure 2-14):

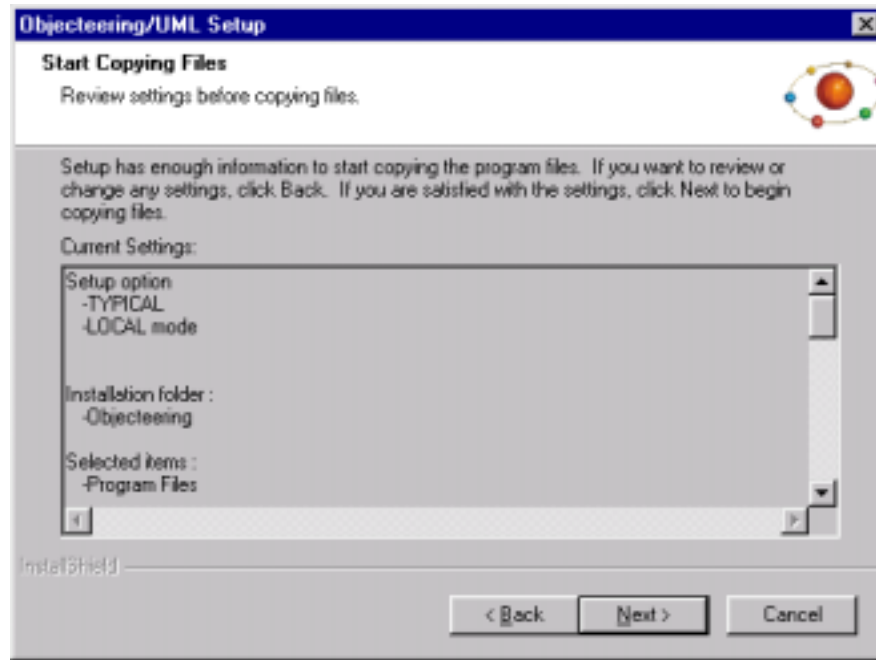


Figure 2-14. Information dialog box for copying the program files

Click on the "Next" button. You can visualize the progress of your installation.

Information on modules

Just before the end of your installation, the following window (Figure 2-15) appears, indicating where the Objecteering/UML modules which have not yet been delivered are located.



Figure 2-15. Window indicating the location of the modules delivered

To use these modules, simply double-click on the relevant ".*prof*" file in OBJING_PATHModules, and then select them in your UML modeling project.

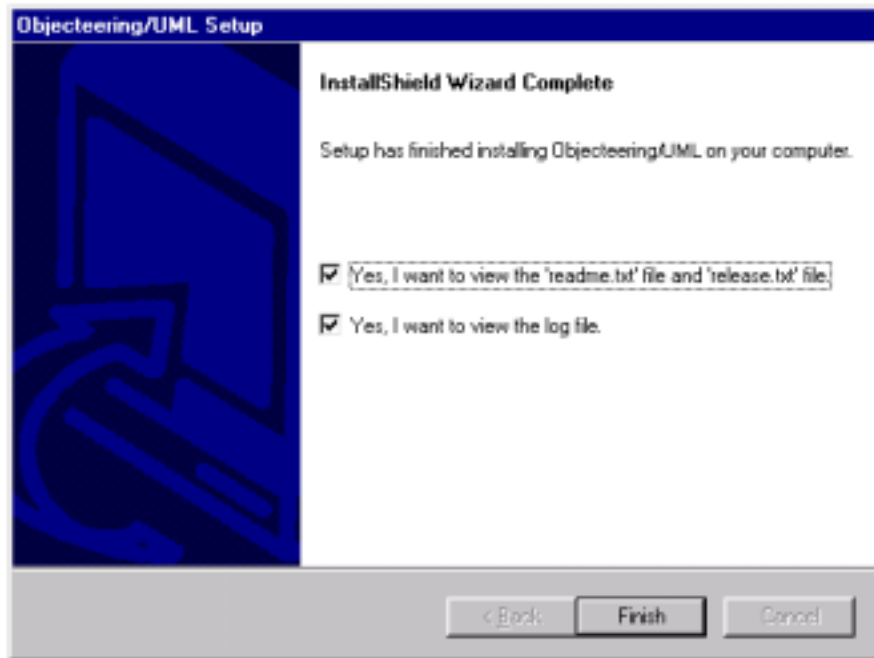
End of installation

Figure 2-16. The "Setup complete" window

Click on the "Finish" button to complete installation and visualize the readme.txt and release.txt files, which provide additional information on the Objecteering/UML tool, and the log file, which provides information on the installation just carried out. If you do not wish to visualize these files, uncheck the tick boxes.

Server installation in Windows

Several users can connect to the same "site". In this configuration, UML modeling, UML profiling and document template projects are accessed directly by the Objectteering/UML site server. This type of installation requires an update of the *TCP/IP services* and of the registry, and must be carried out on the server and on all the client stations. The server must be in Windows NT (Windows NT 4.0 Server or workstation).

Figure 2-17 below shows the client and server access.

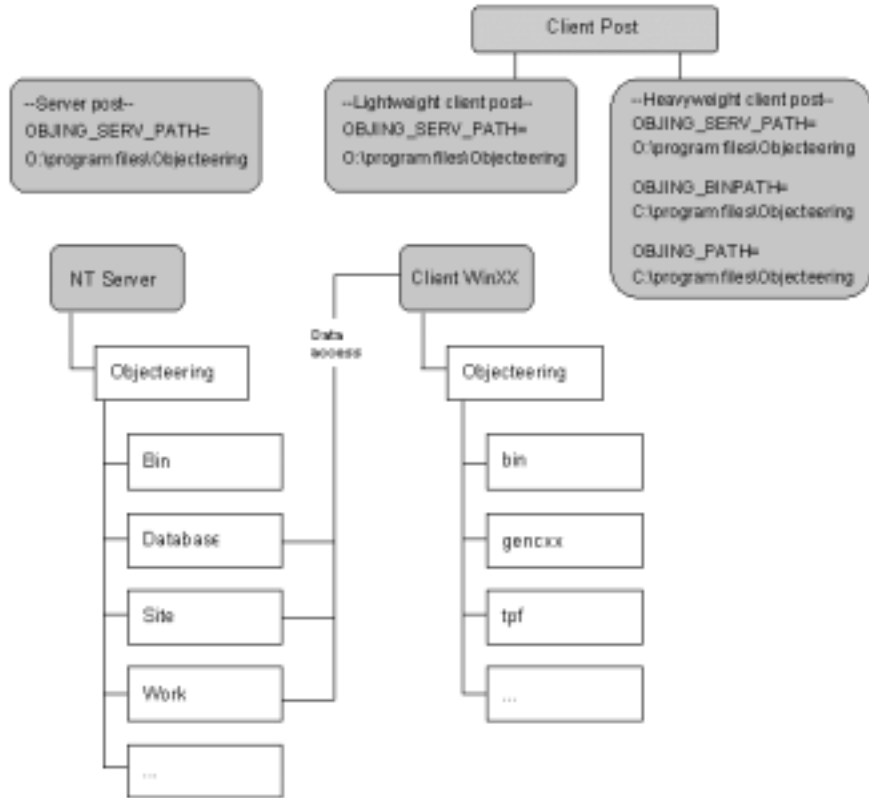


Figure 2-17. Client-server functioning

Note: The TCP/IP layer on the server machine must have previously been installed.

It is strongly recommended that you follow the example below when installing the Objecteering/UML server.

Steps to carry out before starting server installation

Before launching the server installation procedure itself, you must make sure that the installation directory is accessible from the network.

If this is not the case, the installation directory or its parent directory must be shared.

You can now proceed with server installation.

Selecting the Server installation mode

Run the installation procedure in the same way as for a "single station" installation. When entering the installation mode, select the "server" mode (figure 2-18).

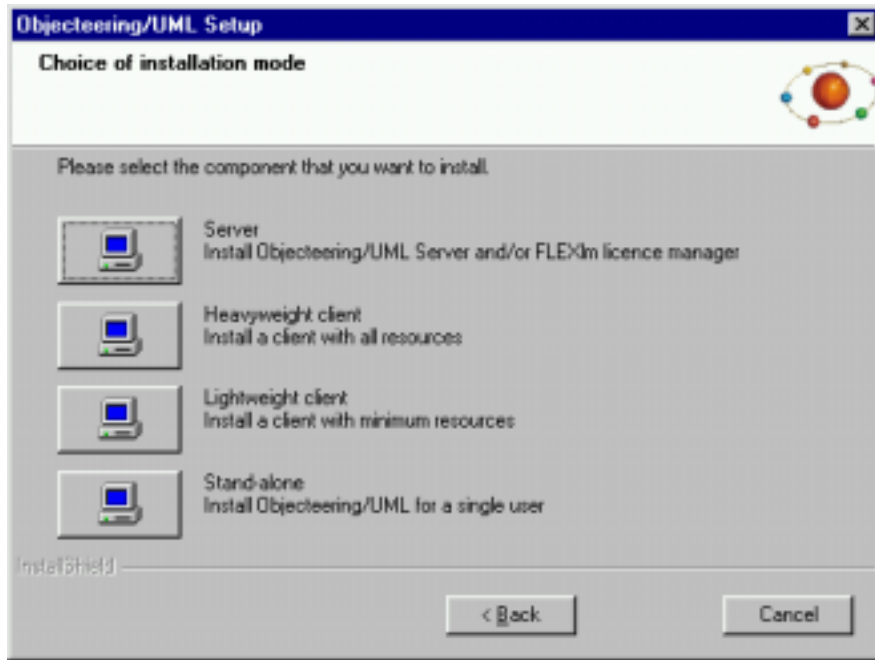


Figure 2-18. Choice of installation mode

Confirm by clicking on the "Next" button.

Chapter 2: Installing Objecteering/UML

The following window (shown in Figure 2-19) then appears. In this window, indicate what kind of server you wish to install.

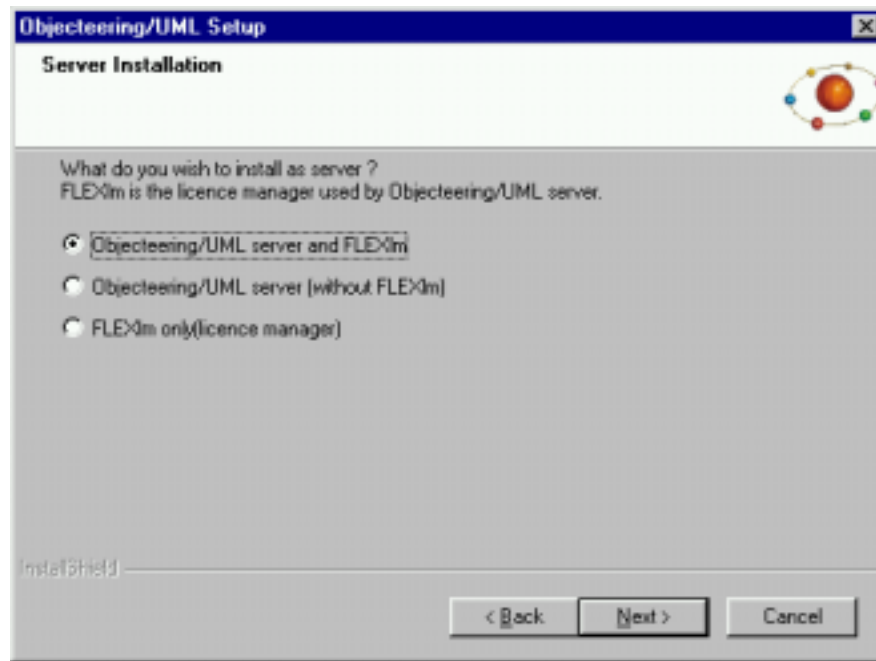


Figure 2-19. Selecting what kind of server you wish to install

The "*Objectteering/UML server and FLEXlm*" option is used to install the FLEXlm license server and Objectteering/UML in server mode on the same machine.

The "*Objectteering/UML server (without FLEXlm)*" implies that you already have a FLEXlm installation (version 7.2 or later) on the server or accessible via the network.

The "VENDOR" line and all the "FEATURES" of the Objectteering/UML license file must be inserted into the license used by FLEXlm, with the exception of the "*Objing_Site*" feature. These modifications will be carried out by the Objectteering Software technical support team.

The "*FLEXlm only (license manager)*" options is used to install only FLEXlm. A license file will be requested during installation.

If you provide the Objectteering/UML license file, you must already have withdrawn the "*Objing_Site*" feature from it. These modifications will be carried out by the Objectteering Software technical support team.

If you opt for an installation which separates FLEXlm from Objectteering/UML, please inform your Objectteering Software at support@objectteering.com.

Make your choice and confirm by clicking on the "Next" button.

Chapter 2: Installing Objecteering/UML

The following window (shown in Figure 2-20) then appears. In this window, enter the location of your license file.

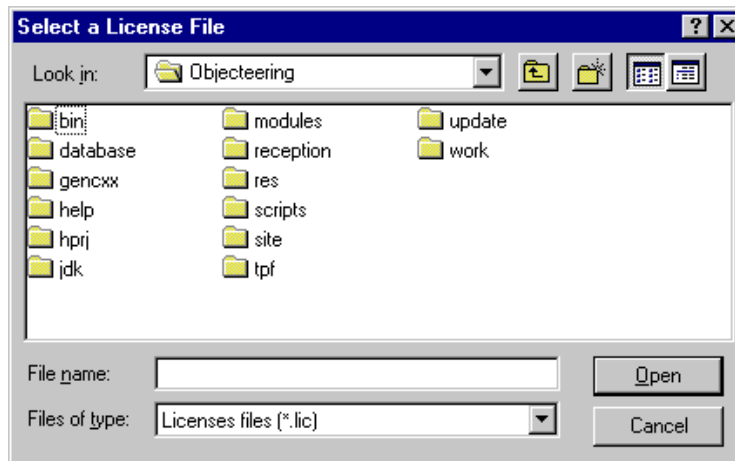


Figure 2-20. The license selection window

Browse in this explorer and select the license file sent to you by Objecteering Software.

Once you have selected your license, the next window then appears. In this new window you should specify the name of your server (as shown in Figure 2-21).

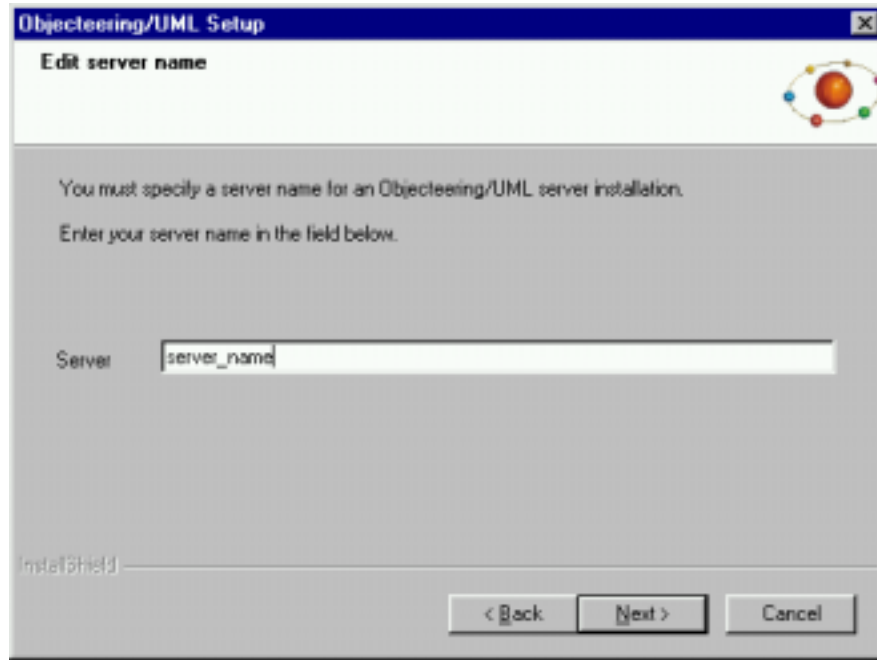


Figure 2-21. Entering your server name

Once you have entered your server name, continue by entering the installation location (as shown in Figure 2-22).

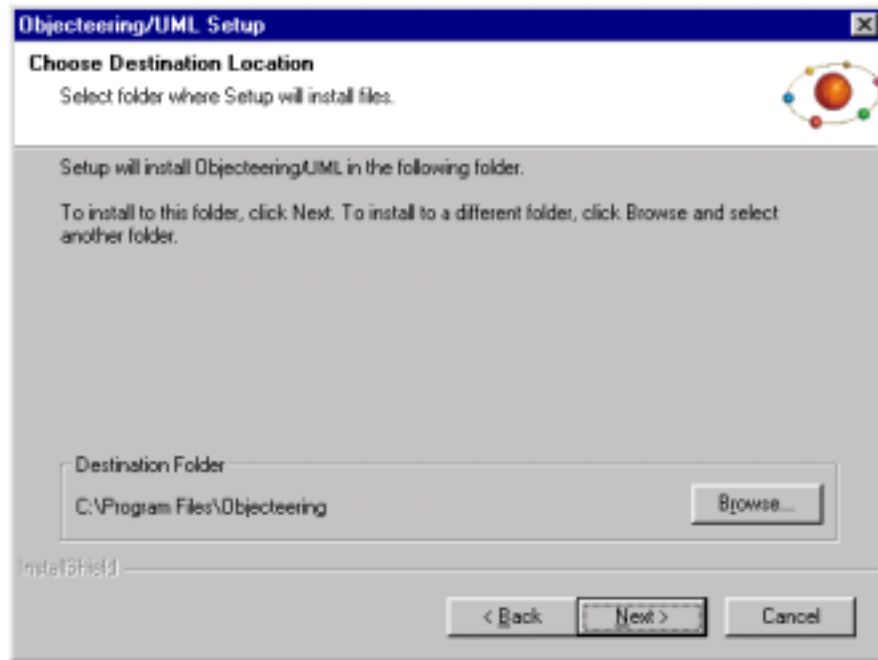


Figure 2-22. Entering the installation location

Simply enter the folder where you wish to install Objecteering/UML. You may use the browse button to make selection even easier.

Note: Please note that the directory in which you choose to install Objecteering/UML must have the same name as the directory you have mapped as network drive, in our example "*Objecteering*".

The next step is to indicate the location of the server, which will be used by all the clients (Figure 2-23).

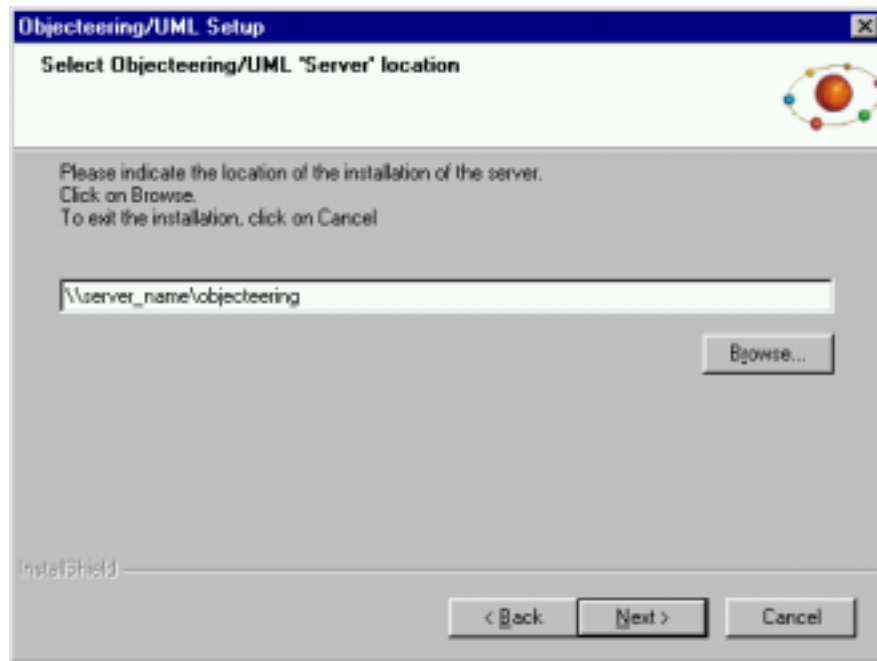


Figure 2-23. Indicating the server which is to be used

Simply indicate the server you have already installed. Make sure you select the site's parent directory.

Note: Please note that the network drive indicated must have the same name as the directory in which you have chosen to install Objecteering/UML (in our example, "Objecteering").

The next step is to enter the system and user environment variables (as shown in Figure 2-24).

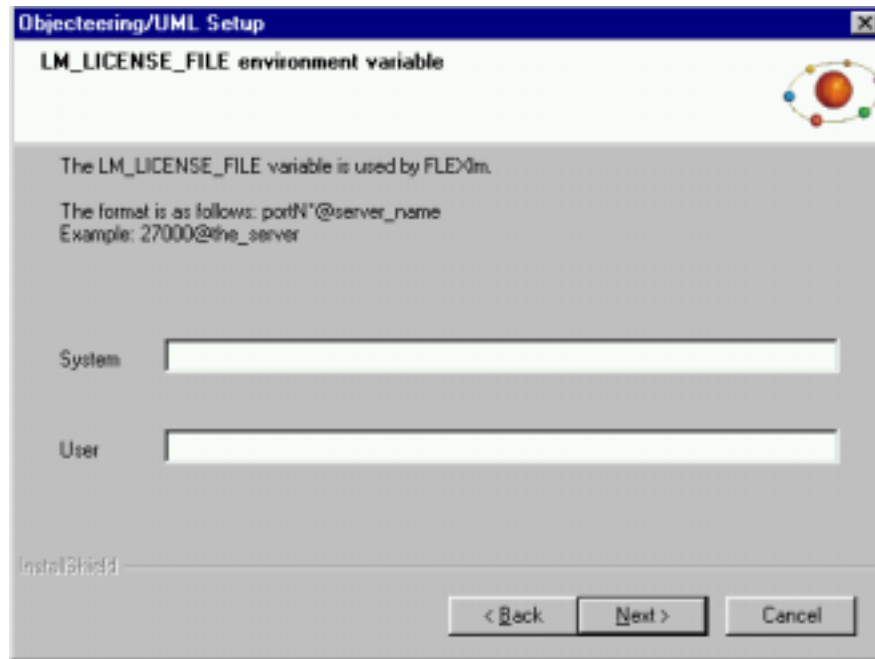


Figure 2-24. Entering the LM_LICENSE_FILE environment variable

In the "System" field, enter the LM_LICENSE_FILE environment variable for your machine.

In the "User" field, enter the LM_LICENSE_FILE environment variable for your particular user ID.

Note: For further information on the LM_LICENSE_FILE environment variable, please refer to the "The LM_LICENSE_FILE environment variable" section in the current chapter of this user guide.

Continue by entering your name and company name (as shown in Figure 2-25).

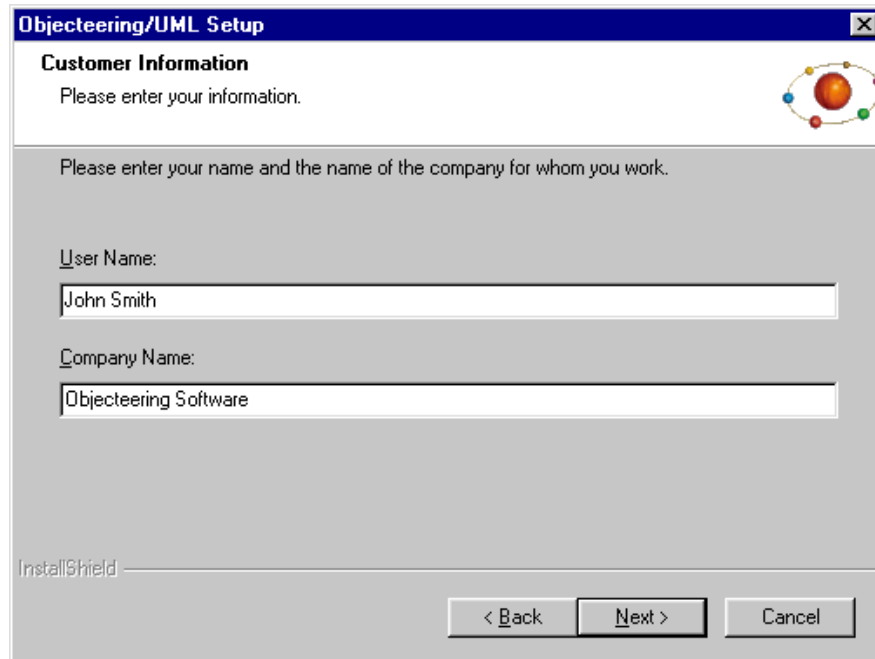


Figure 2-25. Entering your user name and company name

Enter a user name and the name of your company.

From this point on, server mode installation is identical to stand alone installation (for further details, please refer to the "*Stand alone installation in Windows*" section in the current chapter of this user guide.

TCP/IP configuration

In client or server installation mode, the installation requires that the user declare a TCP/IP service, used to run the server, if this does not already exist.

The following dialog box (Figure 2-26) opens and asks the user whether the installation procedure should automatically modify the port/service numbers present in the "services" file.

Note 1: By default, the "Services" file is automatically modified.

Note 2: The option of saving modifications in the "Services_obj" file allows you to manually transfer the modifications into the "Services" file once the installation is complete.

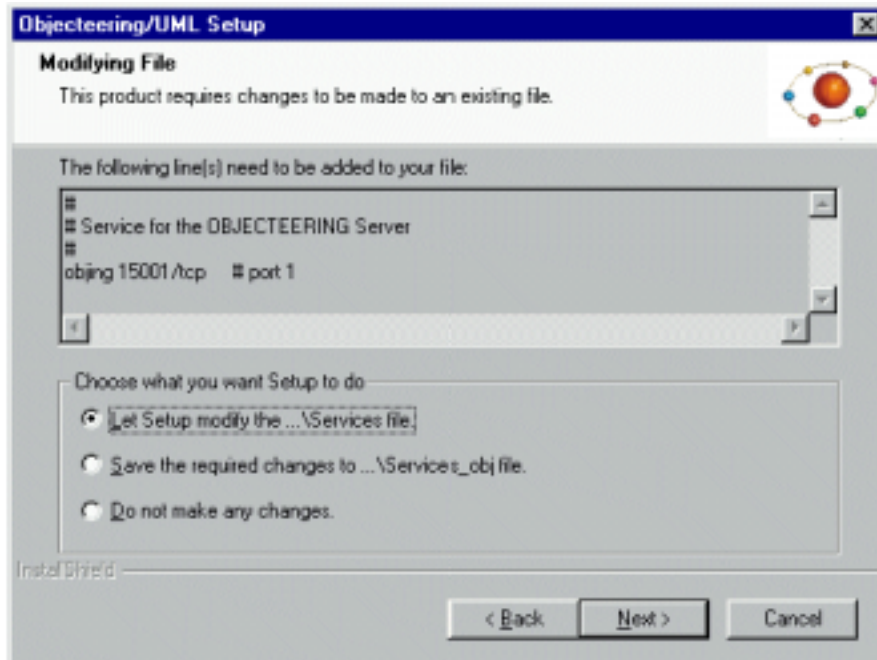


Figure 2-26. Dialog box for modifying the file

Warning! In the event of modifications being made, the blanks at the end of lines must be kept, for example, by adding comments:

```
#  
# Objecteering site server  
#  
objing 15001/tcp # port no 1
```

The numbers chosen are arbitrary, but:

- ◆ they must be identical for all the Objecteering/UML site machines
- ◆ they must not be the same as service ports existing in the file
- ◆ they must not be above 1024

Completing the installation

OBJING_BINPATH specifies access to all the Objecteering/UML binary files.

OBJING_PATH specifies access to the files ensuring the site's administration.

Check that the ports are correctly configured. From the explorer, edit the following file:

```
WINNT\system32\drivers\etc\Services
```

Modules

In "*Server installation*" mode, all the .prof files of the Objecteering/UML modules delivered are installed on the server. To install them, simply double-click on the .prof in question.

Activating the Objectteering/UML site server

The Objectteering/UML site server, which is an NT service, can be activated:

- ◆ in all cases, on the machine declared site server
- ◆ on machines possibly used as the model servers of this site

The Objectteering/UML installation procedure automatically creates a server named "*FpServer*".

At the end of the installation procedure, this service is run automatically. Several administration operations still need to be carried out in order to activate this service. It is possible to change its running mode, so that the service is activated when the machine is started up (by default, this is in manual mode), or so that the user is associated to it.

Activating the FlexIm service

The Objectteering/UML installation procedure automatically creates a license management server procedure called "*FlexIm Objectteering*".

At the end of the installation procedure, this service is automatically launched. By default, the launch mode is set to automatic.

Activating the Objecteering/UML server and the Flexlm service manually

If Objecteering/UML is installed in a directory which is mapped as a network drive, the Objecteering/UML server and the Flexlm service should be activated manually. These services are successfully installed, but are not launched automatically, since they attempt to start as the "LocalSystem", whereas they must start as a user.

In order to activate the Objecteering/UML server and the Flexlm service for an Objecteering/UML server installed on a network drive, go into the control panel, open the "Services" dialog box, double-click on the services in question and modify their startup parameters.

The user account selected must have administrator access rights for the NT work station. We recommend that you use an account provided by the system, in order to avoid any potential problems in the case of account deletion.

Once these steps have been carried out, the Objecteering/UML server and the Flexlm service can be manually started, and will run correctly the next time the machine is started up.

Preparing to install the client station

The client station is installed in a standard way, except that the site's licenses do not need to be provided, since they are centralized in the server. UNC (Unified Naming Convention) names can be used. On the client station, the installation directory of the server must be connected to the same network letter as the server station.

The letter of the connected network must be identical to the one specified during the installation of the Server station. Connection to this network must be realized before a Windows Client installation procedure.

Client installation in Windows

This installation in client mode is used to access an NT or Unix server.

Note 1: The server must previously have been installed.

Note 2: The TCP/IP layer on the client machine must have previously been installed.

UNC paths can be used, or if you wish, you can connect a network drive directed towards the Objecteering/UML installation directory on the server.

Warning! The letter of the network drive on the client must be the same as the server's letter.

Launch "Setup" from the CD ROM and confirm the license dialog box by clicking on the "Next" button. Select one of the "Client" mode (as shown in Figure 2-27).

Note: Heavyweight client and lightweight client modes are similar. The main difference between the two modes is that when a heavyweight client uses a resource, it speaks to the server, obtains the resource, and if it does not exist locally, copies it locally, whereas a lightweight client speaks to the server every time.

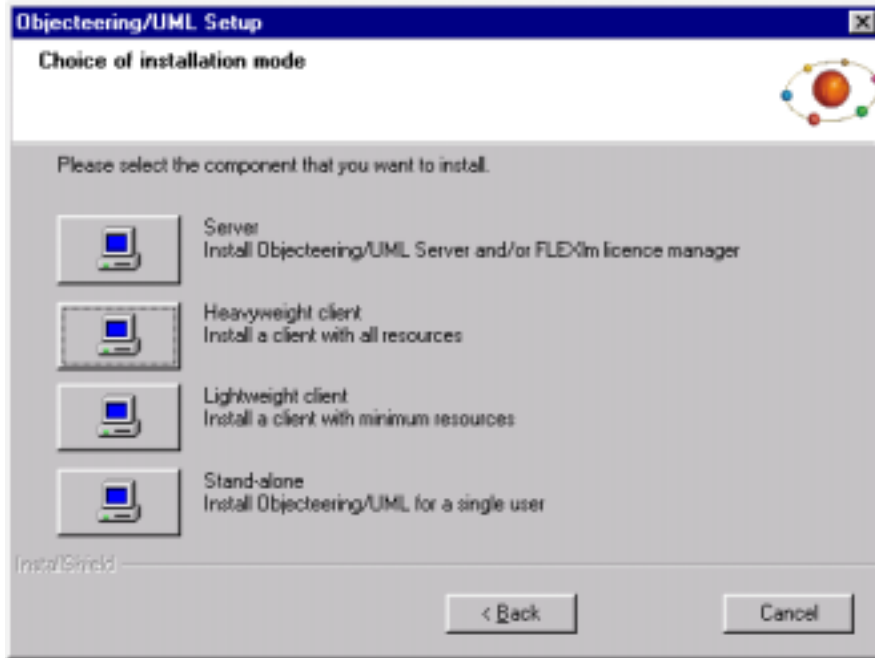


Figure 2-27. Selecting the "Heavyweight client" installation mode

Choose the location of the server installation (Figure 2-28).

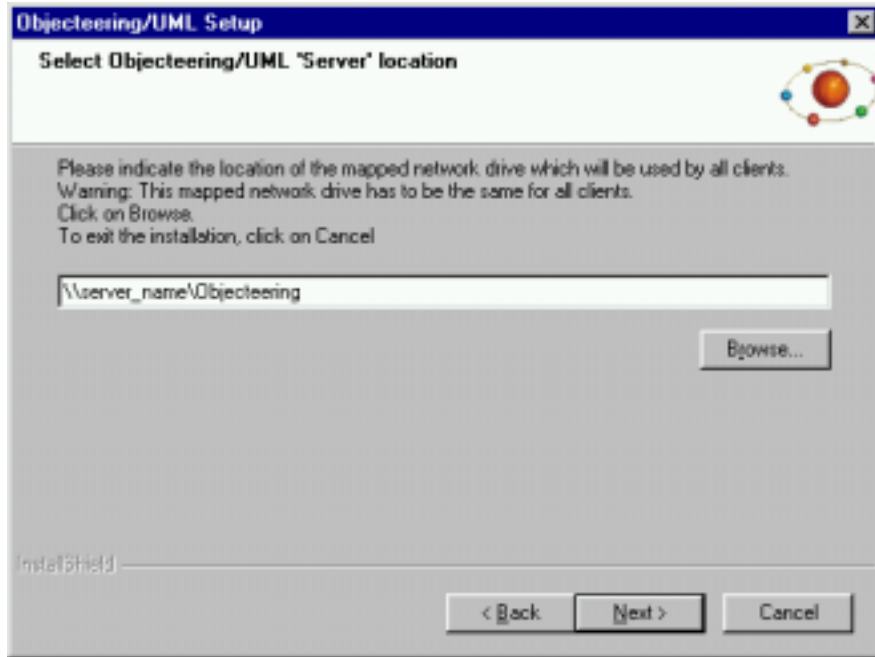


Figure 2-28. Selection of a server directory

If the Objectteering/UML server is a UNIX station, you must then be able to access the UNIX directory from your Client Windows post and to transfer it into this selection dialog box.

If the directory entered does not contain the Objecteering/UML server installation, the following dialog box will appear (Figure 2-29):



Figure 2-29. Objecteering/UML installation directory server incorrect

Confirm the user information dialog box by clicking on the "Next" button.

Confirm the installation directory selection dialog box by clicking on the "Next" button (please see the "Stand alone" section earlier in this chapter).

For heavyweight client installation, the window shown in Figure 2-30 then appears, indicating that modules are being searched for on the site.

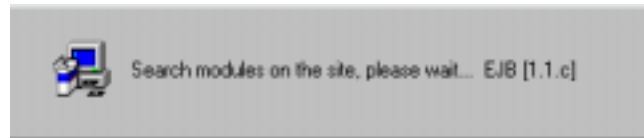


Figure 2-30. Window indicating the progress of module searching for heavyweight client installation

Once modules have been located, continue by selecting the components you wish to install (please see the "Stand alone" section earlier in this chapter).

After selecting components, the window shown in Figure 2-31 then appears.

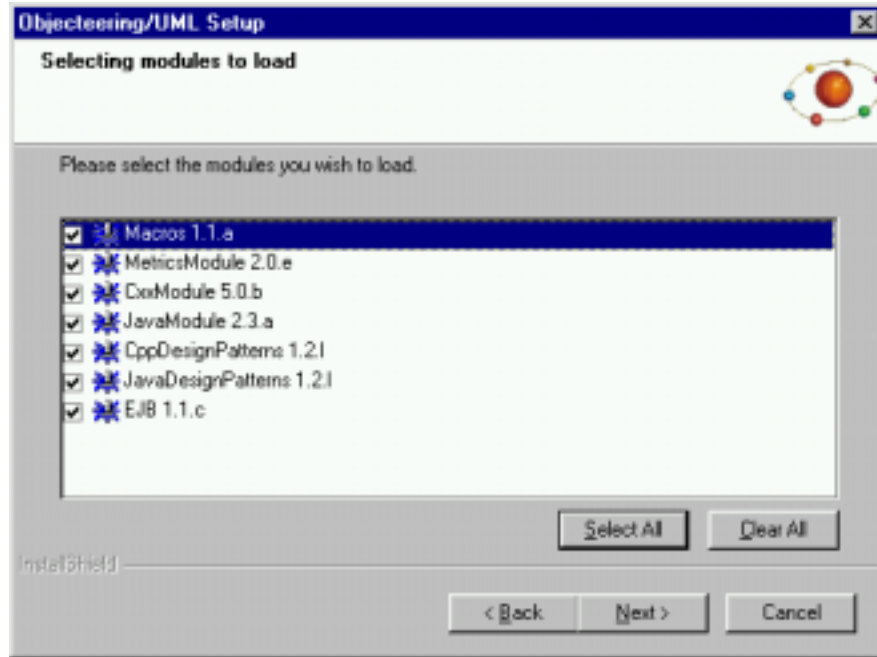


Figure 2-31. Selecting the modules you wish to load from the server

In this window, all the modules found on the server are listed. Simply check the tickboxes of the different modules you wish to load from the server onto your heavyweight client installation, and then click "Next" to continue.

Continue by selecting the program folder desired and then confirm the window by clicking on the "Next" button.

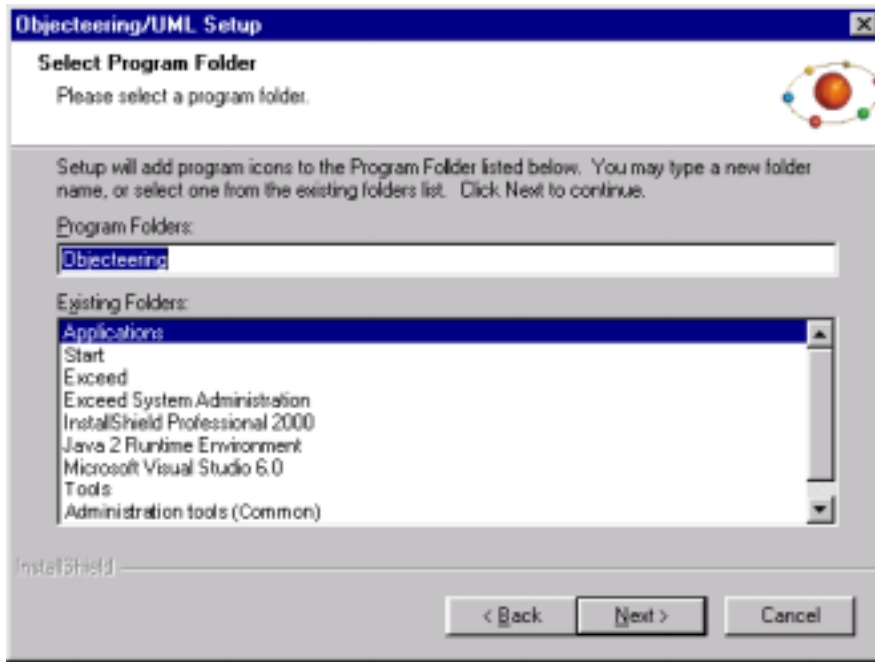


Figure 2-32. Selection of the program file

Chapter 2: Installing Objecteering/UML

Confirm the modification dialog box for the "Services" file (figure 2-33) after consulting the service number proposed by the Objecteering/UML server, by clicking on the "Next" button.

Note 1: By default, the "Services" file is automatically modified.

Note 2: The option of saving the modifications in the "Services_obj" file will allow you to manually carry out modifications in the "Services" file, once installation is complete.

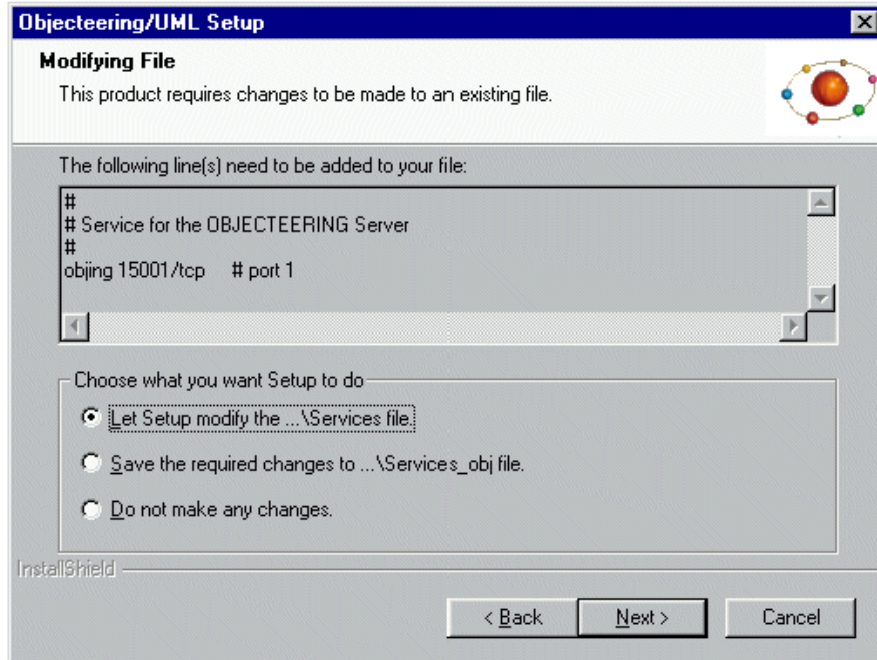


Figure 2-33. Modification of the Services file

Confirm the "Copying program files" dialog box by clicking on the "Next" button, after having checked the different options displayed.

Note 1: If one of the displayed options is not appropriate, it is possible to go back to the last ones by clicking on the "Back" button.

Note 2: By clicking on the "Next" button, the copying of the files will start (as shown in Figure 2-34).

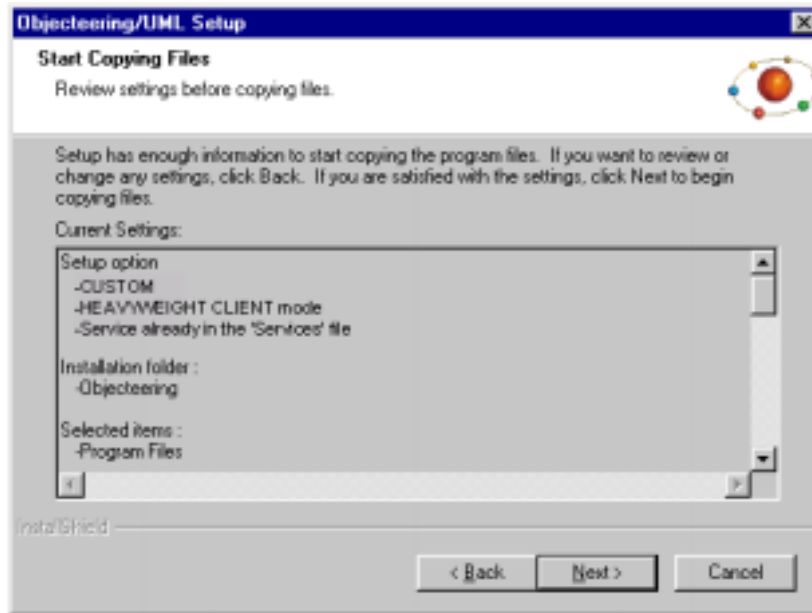


Figure 2-34. Display of installation options

Once all files have been copied onto your disk, the client Windows installation is complete.

Refer to the *Confirmation TCP/IP* section of the server to check the service numbers necessary to the Objecteering/UML server in the "Services" file.

Modules

In "*Client installation*" mode (for both heavyweight and lightweight clients), all the .prof files of the Objecteering/UML modules delivered are installed on the server. To install them, simply double-click on the .prof in question.

Migrating a client installation

When migrating a client installation which predates version 5.1 of Objecteering/UML, the following window (Figure 2-35) will appear. In this window, you should indicate whether or not you wish to migrate to a lightweight client installation or a heavyweight client installation.

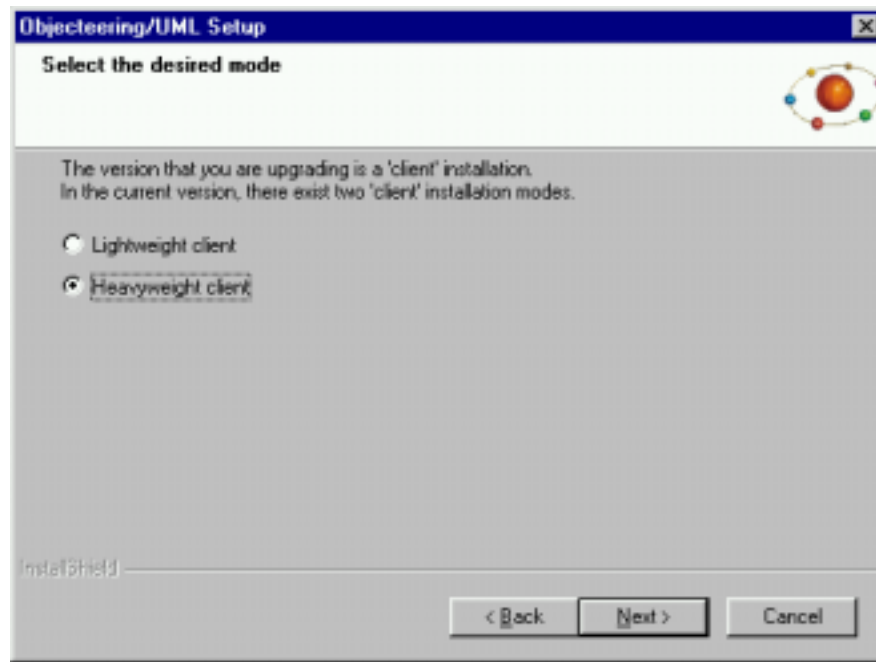


Figure 2-35. Selecting to upgrade a client installation to a lightweight or heavyweight client

Objectteering/UML tools in Windows

The Objectteering/UML tools

The "Start/Programs/Objectteering" menu is used to run the following tools:

- ◆ *Examples*: this is used to launch the demonstration UML modeling project, "VendingMachine".
 - ◆ *Help*: this allows you to access the online Objectteering/UML help, including the *Objectteering/Introduction* that contains *Objectteering's First Steps*. For Windows, the html "browser" is defined by default.
 - ◆ *Uninstall*: this allows you to run the Objectteering/UML uninstallation procedure.
 - ◆ *Administration*: this allows the creation or repair of Objectteering/UML UML modeling, UML profiling and document template projects, as well as the managing of their physical properties. Modules are also managed via this tool.
 - ◆ *Objectteering UML Modeler*: this allows you to run Objectteering/UML Modeler version 5.2.2.
 - ◆ *Objectteering UML Profile Builder*: this allows you to run Objectteering/UML Profile Builder version 5.2.2.
 - ◆ *Reinitialize site number*: this is used to reinitialize your site number, after an event such as a change in license.
 - ◆ *Search engine*: this allows you to run the Objectteering/UML on-line help search engine.
-

Installing in UNIX

Introduction

This section presents the procedure to be followed, when carrying out the installation of Objectteering/UML on a UNIX platform. UNIX platforms supported are Solaris and LINUX, and the installation procedure for these two platforms is the same.

This section is addressed to any person reasonably accustomed with the use and administration of UNIX machines.

This phase includes:

- ◆ the installation of Objectteering/UML
- ◆ the initialization of the information specific to your site

Different installation modes

- ◆ Installation over a previous version
- ◆ Client-server configuration

Prerequisites

To carry out an Objectteering/UML installation, you must have the CD-ROM and the licence file supplied by Objectteering Software.

The product's general characteristics are as follows:

- ◆ disk resources: 150 Mo
- ◆ average installation time: 30 minutes

The licenses required are as follows:

- ◆ a license to possibly add a file in an X11 directory
- ◆ the right to be able to run the "*objingsrv*" server at the start-up of the UNIX machine
- ◆ the "*/etc/services*" file must be modified. You must have the "*root*" user in order to update it.

Procedure

There are two steps involved in the installation of Objecteering/UML:

- ◆ the setting up of the CD-ROM
- ◆ the running of the installation program

Please see the "*Installing licenses*" section of the current chapter of this user guide.

Locating information

It is recommended that you install Objecteering/UML in the `/usr/objecteering` directory. This directory is later named `<installation_directory>`. If the server is being installed, clients must be able to access this directory through NFS.

If you are re-installing, it is strongly recommended that you make a backup copy of the installation directory before re-installation.

Setting up the CD-ROM

The following example applies to an installation in Sun Solaris 2.4 (X11R.5). For LINUX, the installation procedure is the same.

In Solaris, the CD-ROM is generally set up automatically by the system just after the CD-ROM has been inserted in the drive. To check it, make sure the */cdrom/cdrom0* directory contains the *SETUP* file. If the set up has to be carried out manually, use the following command under root:

```
% mkdir /cdrom
% mount -r -F hsfs /dev/sr0 /cdrom
```

If the CD-ROM drive is only accessible through the network, the procedure, under root, is as follows:

- ◆ if the drive is not automatically set up by the station which owns the CD-ROM, use the following command on this station:

```
% mkdir /cdrom
% mount -r -F hsfs /dev/sr0 /cdrom
% share -o ro /cdrom
```

Then on Objecteering/UML's installation station:

```
% mkdir /cdrom
% mount -r <station_name_with_CD>:/cdrom/cdrom
```

- ◆ if the drive is automatically set up by the station which owns the CD-ROM, use the following command on this station after having inserted the CD-ROM :

```
% share -o ro /cdrom/O43_SOL
```

Then on Objecteering/UML's installation station:

```
% mkdir /cdrom
% mount -r <station_name_with_CD>:/cdrom/O43_SOL/cdrom
```

Installing from the CD ROM

In the Objecteering/UML administrator's UNIX account, position yourself in the CD-ROM set directory:

```
% cd /cdrom/cdrom0
```

or

```
% cd /cdrom
```

Run SETUP. This will start the installation procedure.

```
% ./SETUP
```



Figure 2-36. The start of Objecteering/UML's installation program

Click on "Yes" and continue the installation procedure, by answering the questions shown in Figure 2-37.

```

do
Do you accept the terms of the license agreement ?

If you choose No, the installation procedure will stop.
To install Objecteering, you must accept this agreement.
Do you accept ? ([Y]es/[N]o) :
1— Y

-----

Please enter the destination directory for Objecteering/UML '5.2.2
#4900'
( default : [/usr/objecteering522] ) :
/tmp/Objecteering522
**** Warning : the '/tmp/Objecteering522' directory does not exist
-
2— Would you like to create it ? ([Y]es/[N]o) :
Y

-----

Please select the desired configuration mode([L]ightweight_client,
[H]eavyweight_client or [S]erver):
3— S

-----

4— Please enter the server name ( default : [mars] )

-----

To continue this installation, you must enter
the directory and the name of the '.lic' file
given by Objecteering Software to use Objecteering/UML :
5— /integration/lic/licenceMars522.1:c
Please enter the company name (default : [none])
6— Softear[]

```

Figure 2-37. Running the installation procedure

Chapter 2: Installing Objecteering/UML

Steps:

- 1 - Answer "Yes". The installation procedure then continues.
- 2 - Enter the path of the directory where Objecteering/UML is to be installed.
- 3 - Indicate the desired installation mode (lightweight client, heavyweight client or server).
- 4 - Enter the name of the server and press "*Return*" to start the installation procedure.
- 5 - Enter the license file directory (this is only necessary for server mode installations). The license will have previously been delivered to you by Objecteering Software's client support department.
- 6 - Enter the name of your company.

The Objecteering/UML installation procedure is then run.

Re-installing

When a new version of Objecteering/UML is installed over a former installation, the previous version is overwritten, but modeling, profiling and document template projects are retained.

Warning: Accessible projects are automatically upgraded to the new version!

Modules can either be updated or added (as shown in Figure 2-38 below).

```

stera
Please enter the server name ( default : [mars] )

-----
To continue this installation, you must enter
the directory and the name of the '.lic' file
given by Objecteering Software to use Objecteering/UML :
/integration/lic/licenceMars522.lic
Please enter the company name (default : [Softeam])

Search modules on server...
.....

-----
Please select the modules that you wish to update or add.
- Update: Deletes earlier versions and installs the latest version
- Add: Migrates earlier versions and installs the latest version

-----
- Module : ProcessManager
  Versions present on the site are: [1.1.g]
  Version present in the packaging: [1.2]
  Do you wish to [U]pdate or [A]dd the latest version ([[A]dd)?
  [=] to add everything  [-] to update everything
A

-----
- Module : AnalysisProfileModule
  Versions present on the site are: [1.1.g]
  Version present in the packaging: [1.2]
  Do you wish to [U]pdate or [A]dd the latest version ([[A]dd)?
  [=] to add everything  [-] to update everything

```

Figure 2-38. Selecting modules to add or update

Updating a module version signifies that the new version of the module will replace the former version(s), which means that former versions will no longer be available.

Adding a module version signifies that the new version is installed and former versions remain available.

Other necessary operations

Objecteering/UML installation is now complete. However, a few operations still remain:

- ◆ configure the Objecteering/UML user accounts, by including the content of the `<installation_directory>/user/env.csh` file in the `.cshrc` of the accounts that use `'csh'` and the content of the `<installation_directory>/user/env.sh` file in the `.profile` of the accounts using `'sh'` or `'ksh'`. Furthermore, in Solaris, the `'LD_LIBRARY_PATH'` variable must contain `'usr/openwin/lib'`.

Environment variables

The following environment variables are installed automatically during the installation phase:

- ◆ `OBJING_BINPATH`: access path to the Objecteering/UML installation directory
- ◆ `OBJING_PATH`: access path to the file directory which ensures the management of the site

These variables are defined in:

```
<installation_directory>/user/env.csh
```

Note: the modification of these variables is not advised.

- ◆ `OBJING_LANG`: this should be set to `"us"`

Note: The variable must be initialized before launching the tool. According to which shell script you use, run either the `env.sh` or the `env.csh` script.

Objecteering/UML's online help in UNIX

The Objecteering/UML online help is a set of HTML documents. To read them, you need an HTML drive (NSCA mosaic or netscape navigator). The choice of drive is indicated in the `O_HELP_BROWSER` environment variable. By default, mosaic is launched.

For example, to access the documentation using Netscape, set the variable in csh shell as follows:

```
setenv O_HELP_BROWSER netscape
```


Client-Server installation in UNIX

Client-server configuration

Several users can connect to the same project. In this configuration, UML modeling, UML profiling and document template projects are accessed directly by the Objectteering/UML site server.

Installation is carried out on the server in a directory that can be accessed by client stations through NFS. Furthermore, installation requires an update of the TCP/IP configuration of the client and server stations.

TCP/IP Configuration

Before going on to anything else, it is necessary to declare a TCP/IP service, as follows:

```
objing 15001/tcp
```

This service must be declared in the system's "services" file:

```
%>/etc/services
```

Warning! The spaces at the end of lines must be kept, for example, by adding comments.

```
#  
# Objectteering site server  
#  
objing 15001/tcp # port no 1
```

The port numbers chosen are arbitrary but:

- ◆ they must be identical for all the machines of the Objectteering/UML site
- ◆ they must not be the same as port numbers existing in the file
- ◆ they must be greater than 1024

Activating the Objecteering/UML site server

The Objecteering/UML site server must be activated in all cases on the machine declared as site server.

The Objecteering/UML site server is run using the "*objingsrv*" command.

- ◆ It must benefit from the shell environment defined after installation (<installation_directory>/user/ directory).
- ◆ It must be activated with the appropriate licenses in order to write in the declared UML modeling, UML profiling and document template projects on the site.
- ◆ If the entry prompt does not come back, run it in background mode ("*objingsrv*&").
- ◆ If it displays possible diagnosis, redirect the standard exit and the error exit to a "log" file. By default, the "*objingsrv.log*" file is written in the \$OBJING_PATH/site directory.

Activating the FlexIm service

To manually launch the flexIm server, run the following command:

```
lmgrd -c <Objecteering_installation_path>/site
```

Locating the installed software

The Objecteering/UML installation directory must be visible through NFS on all the user machines. For an improved performance, it is recommended that the installation disk be local to the site server.

Installing the client station

The installation of the client station is carried out in a standard way, except that the site's licenses do not need to be given. These are centralized in the server.

To install a UNIX client, the server installation directory must be accessible from the UNIX platform (UNIX directories are often accessible from Windows, whilst Windows directories are not accessible from UNIX). It must, therefore, be possible to install the Objecteering/UML server on a UNIX disk, which is visible from Windows work stations.

To do this, map a network drive to the UNIX disk which is to receive Objecteering/UML files. Objecteering/UML server installation can then be run, specifying the network drive as the installation directory.

Special cases: directories which are not shared

For many technical reasons, the client station and server software directory may be distinct. This is especially the case:

- ◆ if the client and the server are two machines with distinct architectures and systems, for example a PC client and a UNIX server
- ◆ if you want to benefit from an installation on a disk local to a client machine, and not to load the runnables of the application through the network

Updating the client station configuration

The various client stations do not share the server's configuration. It is, therefore, necessary to update the execution environment of the client stations upon each creation, suppression or modification of a modeling, profiling or document template project (these operations are carried out using the "*Administration*" tool).

This is done by manually copying the following file below from the site server machine onto the client station:

```
<installation_directory>/site/site.ifo
```

The LM_LICENSE_FILE environment variable

Introduction

The LM_LICENSE_FILE environment variable is used where a FLEXlm license server is used independently of the Objectteering/UML application server.

This environment variable must be positioned on the application server and on the Objectteering/UML client workstations, with the following value:

```
portnum@FlexLMMachineName
```

"*portnum*" corresponds to the port number used by FLEXlm. By default, this is 27000.

License contents on the license server

The license server contains a license which includes all the features of Objectteering/UML, except the Objjing_Site feature, as shown in the following example:

```
SERVER FlexLMMachineName hostidmachineFlexLM
VENDOR objlicd
FEATURE Objjing_ModelerModule objlicd 5.2.2 31-jul-2003 xxxxxxxxxxxxxx \
    VENDOR_STRING=ENTERP_E HOSTID=ANY SIGN=xxxxxxxxxxxxxxxx
FEATURE Objjing_HModule objlicd 5.2.2 31-jul-2003 xxxxxxxxxxxxxx \
    VENDOR_STRING=ENTERP_E HOSTID=ANY SIGN=xxxxxxxxxxxxxxxx
FEATURE Objjing_CxxModule objlicd 5.2.2 31-jul-2003 xxxxxxxxxxxxxx \
    VENDOR_STRING=ENTERP_E HOSTID=ANY SIGN=xxxxxxxxxxxxxxxx
```

License contents on the Objecteering/UML application server

The Objecteering/UML application server contains a license which includes only the Objing_Site feature, as shown in the following example:

```
FEATURE Objing_Site objlicd 5.2.2 31-jul-2003 uncounted xxxxxxxxxxxxxxxx \  
  VENDOR_STRING=<SiteNumber>-ENTERP_E HOSTID=ANY
```

Installing licenses

Entering license information

Objectteering Software provides a license with the *.lic* extension. This license takes the following form:

- ◆ SERVER idfix 17003456 21987
 - ◆ VENDOR demo
 - ◆ FEATURE Objjing_Site demo 1.0 10-jan-2000 uncounted 426B3A9472569 \
 - ◆ VENDOR_STRING=9999-ENTERP_E HOSTID=ANY
 - ◆ FEATURE Objjing_modeler demo 1.0 10-jan-2000 5 3F873B783D0C \
 - ◆ VENDOR_STRING= « 7002 »
 - ◆ FEATURE Objjing_genericity demo 1.0 10-jan-2000 5 C9ABF27B80B2
 - ◆ FEATURE Objjing_cplus demo 1.0 10-jan-2000 5 EE2C82DF8B7F
 - ◆ FEATURE Objjing_java demo 1.0 10-jan-2000 5 11FBB11E730E
-

Using Imtools

Overview of Imtools (Windows)

Imtools is a flexlm tool delivered as standard with Objecteering/UML. It groups together a set of tools used to manage the use of licenses on the network.



Figure 2-39. The *Imtools* window

| The ... field | is used to ... |
|----------------------|---|
| Checksum | carry out a check of the license file. |
| Diagnostics | diagnose problems when a license file cannot be used or where a feature is not correct. |
| Shutdown | close a daemon. |
| Hostid | return information on the system and the hostid of machine. |
| Reread | force the daemon to reread the license file. |
| Status | provide information on the state of license activity on the server. |
| Switchr | switch the report write log file. |
| Time | print out the system's internal time settings, intended to diagnose any time zone problems. |
| Version | indicate the flexlm binary version. |

Note: All the information given out by lmttools can be saved in a text file. To do this, click on "Save *text*" and choose the name of the desired file.

Modifying your Objecteering/UML installation

Overview of installation modification

To modify your Objecteering/UML installation, first activate the "Start/Programs/Objecteering/UnInstall/UnInstall Objecteering" menu. Once this menu has been activated, the "Objecteering Setup Maintenance" window appears (as shown in Figure 2-40).

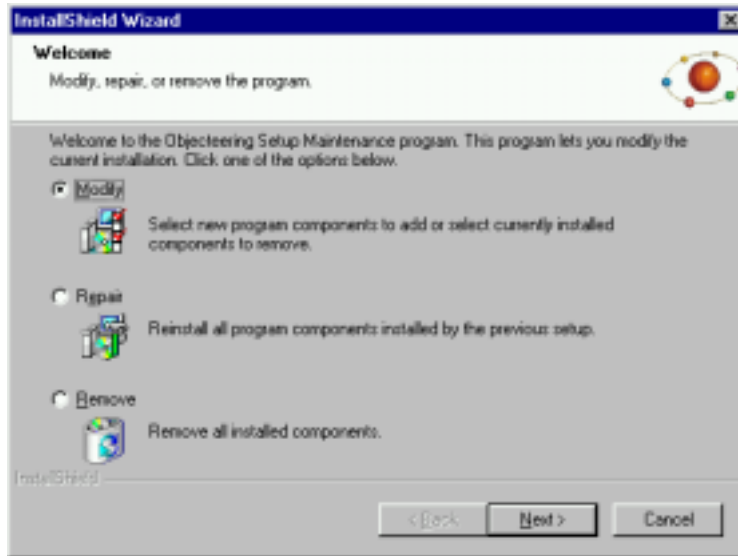


Figure 2-40. The "Objecteering Setup Maintenance" window

Three installation modification options are available:

- ◆ "Modify", which is used to add new program components and/or remove components which have already been installed
- ◆ "Repair", which is used to reinstall all program components installed during the last setup
- ◆ "Remove", which is used to remove all installed program components

"Modify" mode

After selecting the "Modify" mode, click on the "Next" button. The following screen (Figure 2-41) then appears.

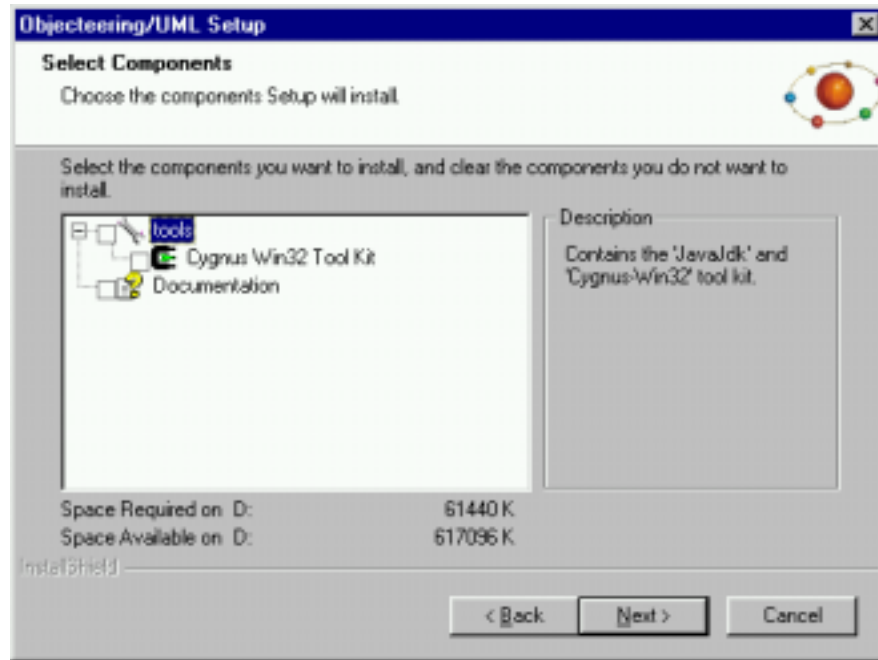


Figure 2-41. "Modify" mode

If you wish to add new program components to your setup, or to remove some of those already installed, simply check and/or uncheck the tick boxes next to the relevant program components.

Once you are happy with your selection, click on the "Next" button to proceed. The progress of your modifications can be monitored on the screen which then appears. When the modifications are complete, the following screen (Figure 2-42) appears.

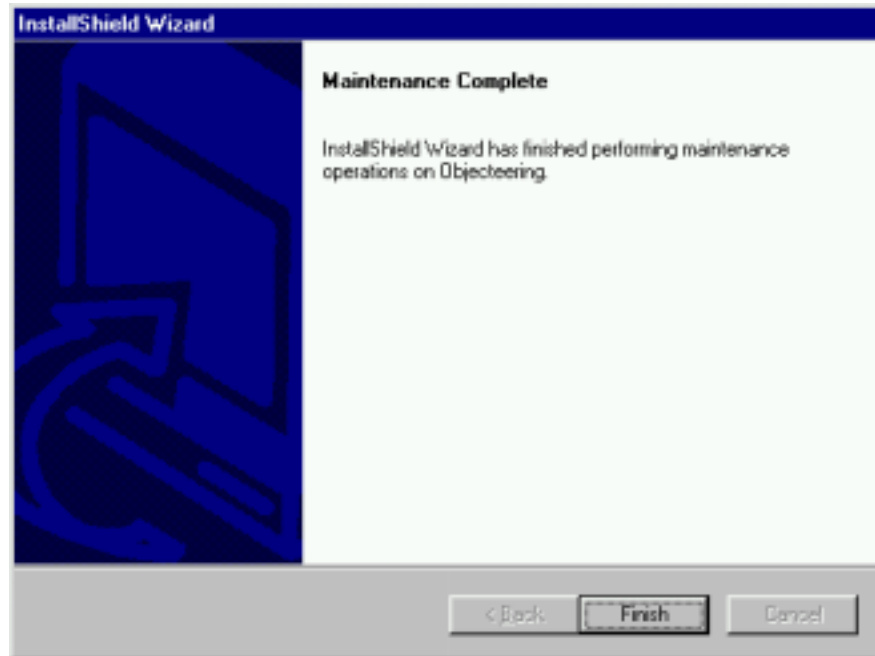


Figure 2-42. Screen informing you that modifications are complete

"Repair" mode

After selecting the "Repair" mode, click on the "Next" button. Objectteering/UML then carries out an installation which is identical to the last setup run. You can monitor the progress of the installation on the "Setup Status" screen which is displayed.

At the end of this installation, the screen shown in Figure 2-42 appears, to confirm that installation modifications are complete.

"Remove" mode

After selecting the "Remove" mode, click on the "Next" button. The following screen (Figure 2-43) appears.

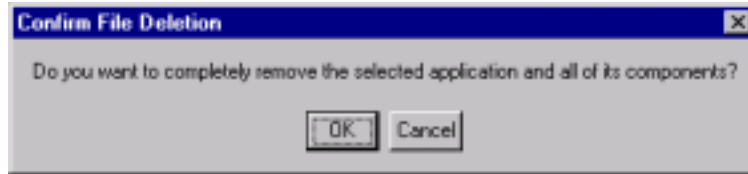


Figure 2-43. Removal confirmation window

Click on the "OK" button. The uninstallation procedure then starts.

At the end of this operation, the screen shown in Figure 2-42 appears, to confirm that installation modifications are complete.

Read only and locked files

If read only files are detected during uninstallation or installation modification operations, the following screen (Figure 2-44) appears.

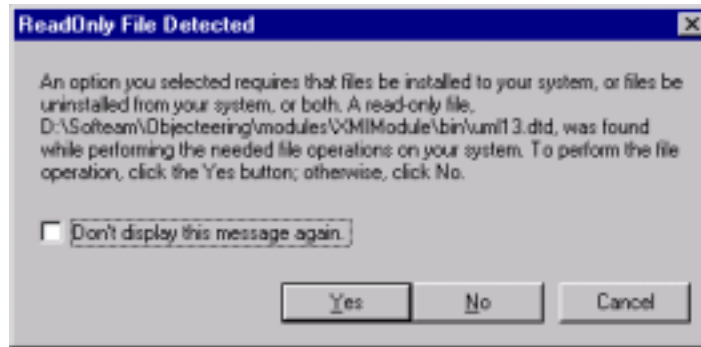


Figure 2-44. Window indicating that read only files have been detected

If you wish to proceed with the uninstallation or installation modification procedure, click on the "Yes" button. Checking the "*Don't display this message again*" tickbox means that read only files will automatically be uninstalled, without this confirmation question being asked again.

We recommend that you proceed with the uninstallation of read only files, since it is advisable to delete all files before reinstalling Objecteering/UML.

Chapter 2: Installing Objecteering/UML

If locked files are detected during uninstallation or installation modification operations, the following screen (Figure 2-45) appears.

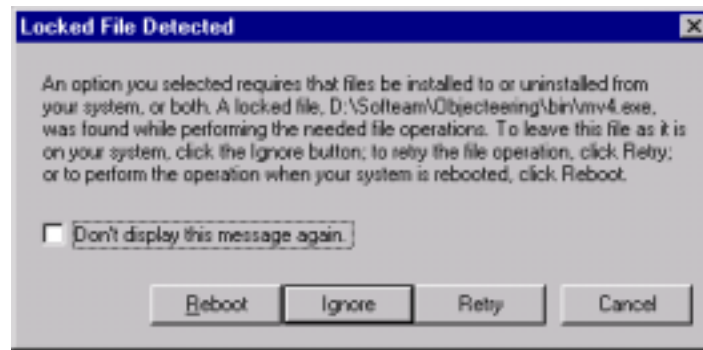


Figure 2-45. Window indicating that locked files have been detected

Locked files are files which are in the process of being used, for example, if Objecteering/UML is running.

Chapter 3: First Steps

Preparation

Introduction

Welcome to our *Objecteering/UML First Steps!*

We recommend that every new user follow these first steps, in order to get a general idea of the numerous complex and powerful functions of the Objecteering/UML CASE tool.

The aim is to help you become familiar with the Objecteering/UML CASE tool, notably with regard to the following operations:

- ◆ creating a UML modeling project
- ◆ working in the explorer
- ◆ creating different kinds of diagrams
- ◆ using Objecteering/UML mechanisms: assisted data entry, undo/redo, consistency checks, copy/paste
- ◆ using modules
- ◆ entering and producing documentation
- ◆ producing and revising Java code

Our on-line help is always available, and can constantly be accessed from the Objecteering/UML tool itself, by using the ever-present "*Help*" or "?" buttons.

These First Steps last, on average, 90 minutes for an inexperienced user.

Creating a new UML modeling project

Creating the "TrainingSystem" UML modeling project

In Objectteering/UML, UML modeling projects are the work spaces used to build and work with UML models.

The first operation we are going to carry out in these Objectteering/UML First Steps is the creation of a new UML modeling project (as shown in Figure 3-1).

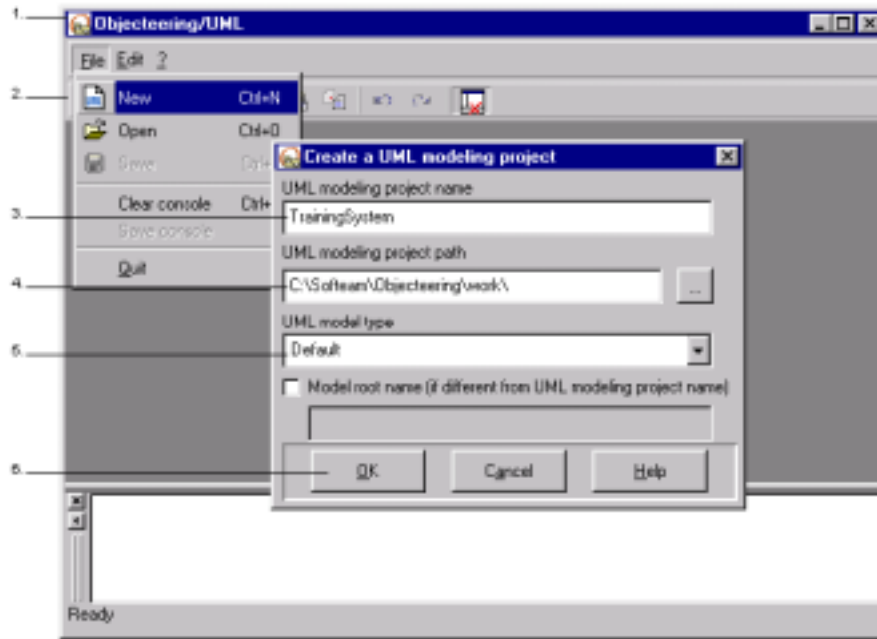




Figure 3-1. Creating the "TrainingSystem" UML modeling project

Steps:



- 1 - Click on the  *Objectteering/UML Modeler* icon in your desktop. The window shown in Figure 3-1 will then appear.
- 2 - Click on the "File/New" menu. The "Create a UML modeling project" window will then open.
- 3 - In the "UML modeling project name" field, enter the "TrainingSystem" name.
- 4 - In the "UML modeling project path" field, enter the path of the directory where the new UML modeling project is to be created. You may also use the  icon to open a file browser through which you can select your UML modeling project path.
- 5 - In the "UML model type", select a type for the modeling project which is to be created. For example, by selecting the "DefaultJava" type, your Objectteering/UML working environment will be automatically configured for Java development with the *Objectteering/Java* and *Objectteering/Design Patterns for Java* modules. Here, we are going to select the "Default" UML model type.
- 6 - Confirm by clicking on the "OK" button.

Note: By default, the name of the UML model root which appears in the explorer is the same as the name of the UML modeling project itself (in our example, "TrainingSystem"). If you should wish to give the UML model root another name, you should simply check the "Model root name (if different from UML modeling project)" tickbox, which will then allow you to enter a different name directly in the field below. It is not possible to change the UML model root name once the UML modeling project has been created.

Result

Objectteering/UML Modeler will then open on your newly created UML modeling project (as shown in Figure 3-2).

Note: UML modeling projects can only be created or opened using the *UML Modeler* tool.

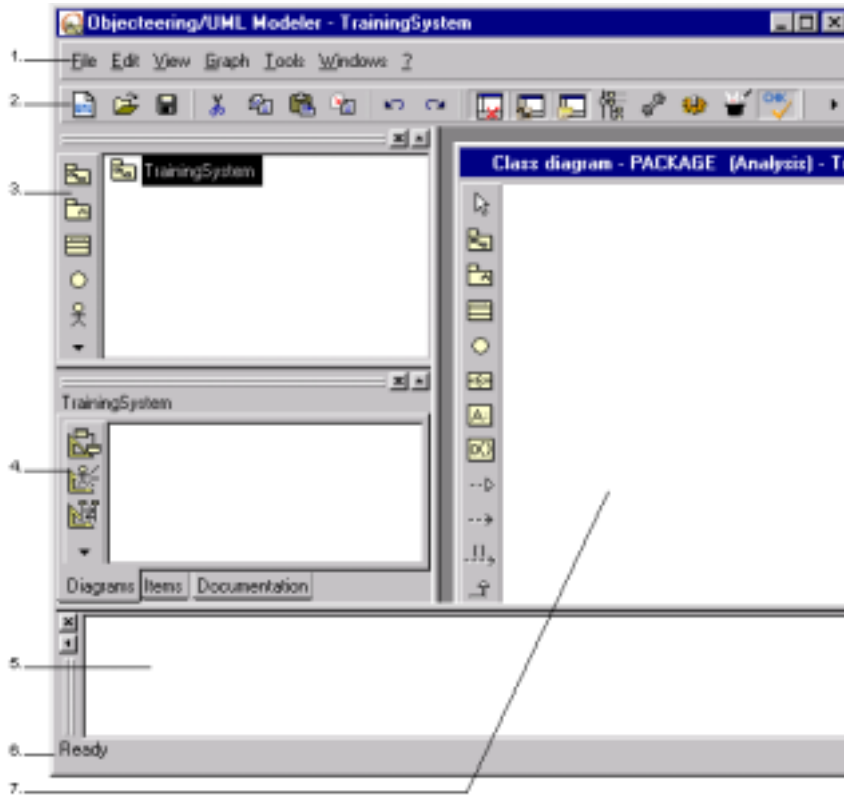


Figure 3-2. The first window you will see - *Objectteering/UML Modeler* on your newly created UML modeling project

Key:

- 1 - Menu bar: This contains the "File", "Edit", "View", "Graph", "Tools", "Windows" and "?" menus.
 - 2 - Menu shortcut bar: This provides icons associated with certain functions present in the menu bar.
 - 3 - Explorer: The explorer is one of the main tools used to work with model elements within a UML modeling project. As you can see in your UML modeling project, the name of the UML modeling project and the name of the UML model itself are identical (this is not the case if you decide to give the model a different name when creating your UML modeling project, by checking the "*UML model*" tickbox and entering a different name).
 - 4 - Properties editor: This box contains a number of tabs, each containing information specific to a certain domain. For example, the "*Diagrams*" tab provides the icons you will need to create diagrams in Objecteering/UML, as well as information on those diagrams which already exist for the selected modeling element.
 - 5 - Console: This contains operation traces, error messages and warnings.
 - 6 - Status bar: This provides information complementary to that displayed by the bubbles which appear over the tool's icons.
 - 7 - Class diagram: When a new UML modeling project is created and opened, a class diagram is automatically created and opened on the root element of the new UML modeling project.
-

Changing UML modeling project

If you wish to change from the UML modeling project you are working on to another modeling project, simply follow the steps illustrated in Figure 3-3 below.

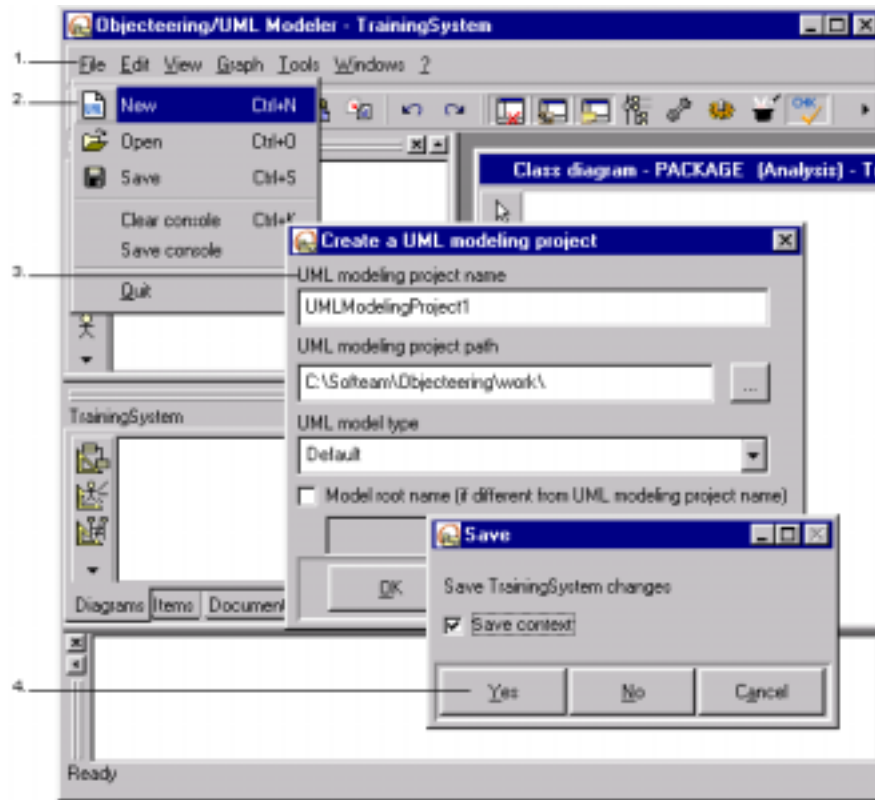


Figure 3-3. Changing UML modeling project

Steps:

- 1 - Click on the "File" menu.
- 2 - Select the "New" or "Open" menu items (depending on whether you wish to create a new modeling project or open an existing one). The "Create a UML modeling project" or "Open an existing UML modeling project" window then appears.
- 3 - Enter the relevant information, and confirm by clicking on the "OK" button.
- 4 - A dialog box appears, asking you if you wish to save the work carried out on your original UML modeling project before closing it and creating or opening another. Click on the "Yes" button to confirm. Your original modeling project is then closed and the new one opened.


Note: The "Save context" tickbox is used to save your model context (size and position of open graphic editors, selected elements remembered in explorers other than the main explorer). For further details on saving your model context, please refer to the "Saving your model context" section in chapter 3 of the *Objectteering/UML Modeler* user guide.

For further information on creating or opening of UML modeling projects, please refer to the "Creating or opening a UML modeling project" in chapter 3 of the *Objectteering/UML Modeler* user guide.

For information on receiving and upgrading UML modeling projects created on other sites or using earlier version of Objectteering/UML, please refer to the "Receiving and upgrading UML modeling projects" section in chapter 3 of the *Objectteering/UML Modeler* user guide.

Selecting modules in the current UML modeling project

Presentation

The  "UML modeling project modules" icon

The Objecteering/UML CASE tool contains modules, which provide particular services on constructed models. For example, the following modules are available:

- ◆ *Objecteering/Documentation*
- ◆ *Objecteering/UML Profile Builder*
- ◆ *Objecteering/C++*
- ◆ *Objecteering/Java*
- ◆ *Objecteering/Design Patterns for C++/Java*

Selecting modules

Objecteering Software modules are delivered as .prof files and are located in the \$OBJING_PATH/modules, which is created during installation.

During the installation procedure, the user can select which modules he wishes to have installed on his site, simply by electing to carry out a "Custom" installation (please see Figure 2-8, 2-9 and 2-10 in the "Stand alone installation in Windows" section in chapter 2 of the current user guide) and by checking the tickboxes of the relevant modules. Once Objecteering/UML installation is complete, he simply has to select the module for the current UML modeling project in the "Modules" window (shown in Figure 3-4).

Module licenses must be present where necessary.



The "UML modeling project modules" icon is used to select those modules which are to be used.


Here we are going to select the *Objecteering/Java* module.



Figure 3-4. Module selection window

Chapter 3: First Steps

Steps:

- 1 - Click on the  "UML modeling project modules" button.
- 2 - Select the *JavaModule* module.
- 3 - Click on "Add".
- 4 - Confirm.

Note: The *Objectteering/Macros* module can be selected to provide you with a number of standard Objectteering macros, which can be run on your model elements. This module can also be used to create your own macros. For further information, please refer to the "Macros" section in chapter 3 of the *Objectteering/UML Modeler* user guide.

Module selection and the properties editor

Certain modules, when selected in Objectteering/UML, add tabs to the properties editor. This is the case for the *Objectteering/C++*, *Objectteering/Java*, *Objectteering/Visual Basic* and *Objectteering/Documentation* modules.

It should be noted that where a version of a module which adds a tab to the properties editor is selected in place of an earlier version of the module in question, which did not provide this service, you should quit and restart Objectteering/UML, in order for the properties editor to be correctly displayed.

Working in the explorer


Presentation



The explorer is one of the central Objecteering/UML editing tools. It is used to create and visualize structural elements of a model (classes, operations, attributes, etc.), and gives a hierarchical view of the model's organization. Here, we are going to create several packages, classes and operations.

When the explorer is launched, a package (called the UML model root) associated with the UML modeling project, and with the same name, appears (here, the "*TrainingSystem*" package). This modeling project is the root of the explorer's model element hierarchy.

Creating a package in the explorer

 We are now going to create the "Training" package in the "TrainingSystem" package (as shown in Figure 3-5).

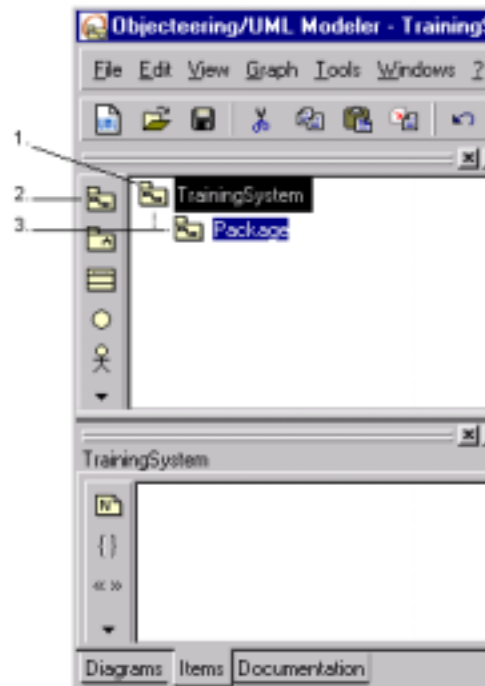




Figure 3-5. Creating the "Training" package


Steps:

- 1 - Select the "TrainingSystem" package.
- 2 - Click on the  "Create a package" icon.
- 3 - Enter the name of your new package ("Training") over the highlighted text (Package), and press "Return".

The name of your new package has been confirmed. You will have noticed that by pressing "*Return*", another package has been created. This is the continuous entry creation mode mechanism, which is present on most elements in the explorer. Click elsewhere in the explorer or press the "*Escape*" key to stop this mechanism.

Click on the  "*Undo*" icon to make the additional package created by pressing return disappear. A line appears in the console, indicating that the "*undo*" operation has been successfully carried out.

Creating classes

The  "Create a class or interface" icon

We are now going to create the "ResponsibleForTraining" and "TrainingManager" classes in the "Training" package (as shown in Figure 3-6). The continuous entry creation mode mechanism is used to rapidly enter several classes one after the other.

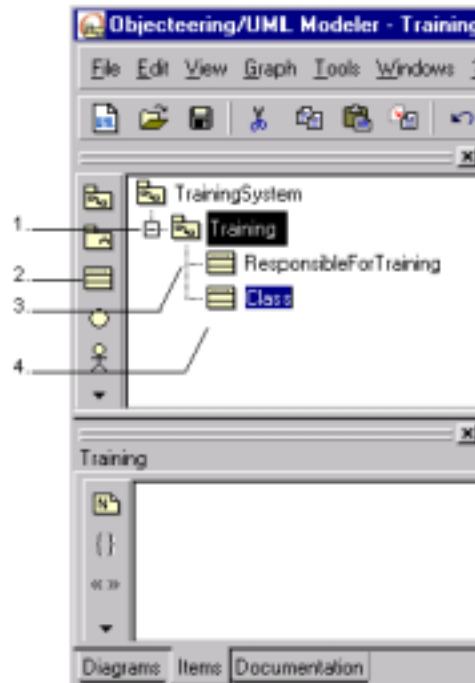



Figure 3-6. Creating the "ResponsibleForTraining" and "TrainingManager" classes

Steps:

- 1 - Select the "*Training*" package.
 - 2 - Click on the  "Create a class" icon.
 - 3 - Enter the name of the "*ResponsibleForTraining*" class over the highlighted text (Class) and press "*Return*".
 - 4 - Enter the "*TrainingManager*" name over the highlighted class text which has just been created by pressing "*Return*", and then click elsewhere in the explorer to stop the continuous entry creation mode.
-

Creating an operation

The  "Operation" icon

The continuous entry creation mechanism will allow us to create several operations in a class. We are going to create the "*manageTraining*" and "*allocatePupil*" operations in the "*ResponsibleForTraining*" class (as shown in Figure 3-7).

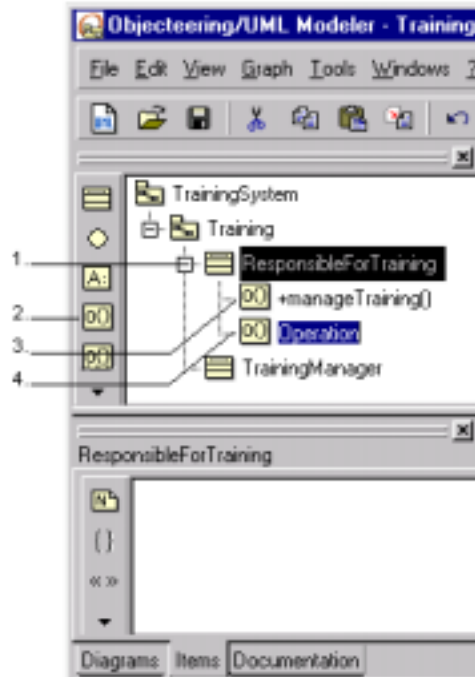



Figure 3-7. Creating the "*manageTraining*" and "*allocatePupil*" operations

Steps:

- 1 - Select the "*ResponsibleForTraining*" class.
- 2 - Click on the  "Create an operation" icon.
- 3 - Enter the "*manageTraining*" name over the highlighted text ("*Operation*") and then press "*Return*" on your keyboard, which will create a second operation.
- 4 - Enter the "*allocatePupli*" name over the highlighted text on the operation which has just been created. Stop the continuous entry creation mechanism, by left-clicking elsewhere in the explorer.

Note: Operations are displayed differently depending on their visibility. Operations with public visibility are shown with a "+" before their names. Operations with protected visibility are displayed with a "#" before their names. Operations with private visibility are shown with a "-" before their names. By default, operations are created with public visibility.

Adding a parameter to an operation

The  "Parameter" icon

We are now going to affect the "pupil" parameter to the "allocatePupil" operation (Figure 3-8).

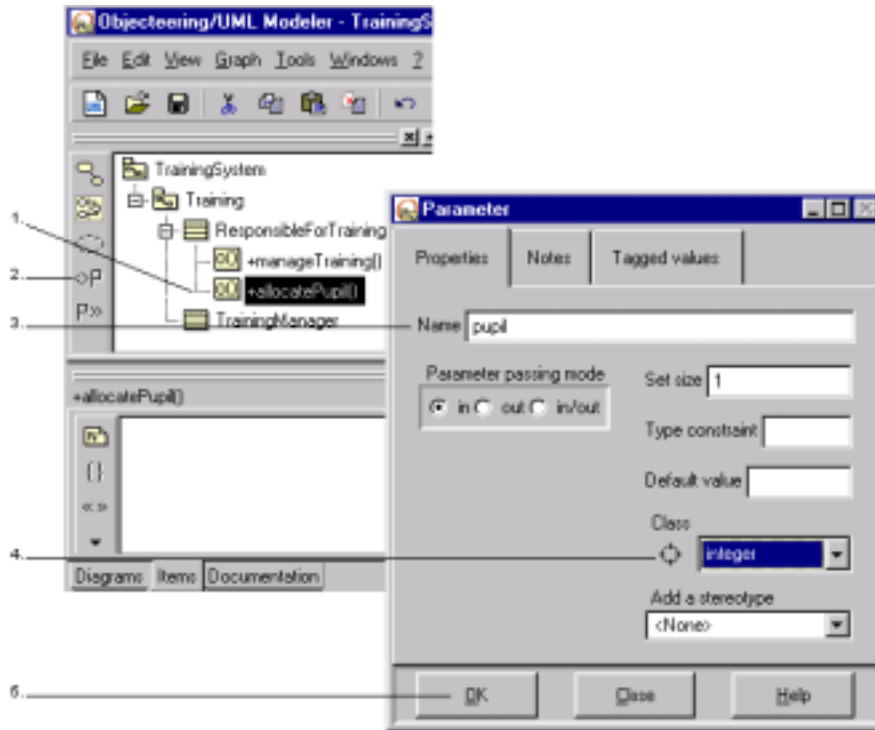




Figure 3-8. Affectation of the "pupil" parameter on the "allocatePupil" operation

Steps:

- 1 - Select the "*allocatePupil*" operation.
- 2 - Click on the  "Add a parameter" icon. A dialog box appears.
- 3 - Enter the "*pupil*" name over the highlighted text.
- 4 - Choose "*integer*" in the scrolling list. This scrolling list allows you to select the parameter class. This is the assisted data entry mechanism provided by the Objectteering/UML consistency mechanisms.
- 5 - Confirm.
- 6 - Create the  "*doTrainingReview*" operation in the "*TrainingManager*" class.

Result

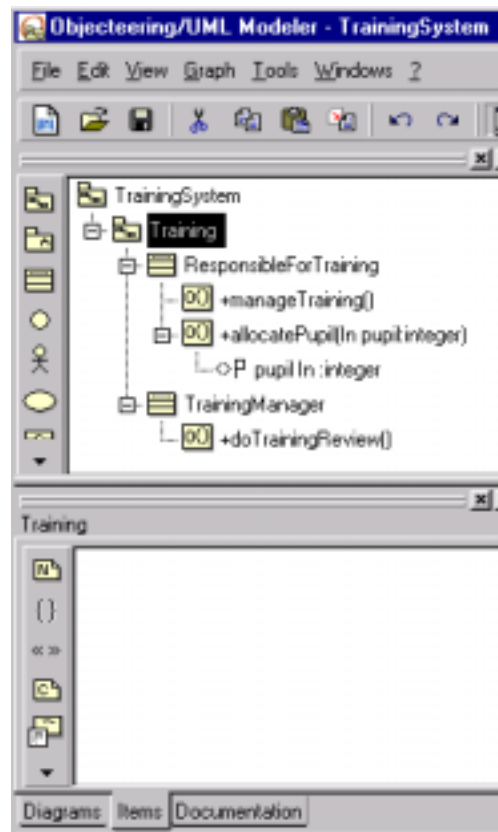



Figure 3-9. Model entered in the explorer

Note: Objecteering/UML displays parameter passing modes in the syntax of the operations (In/Out)

Creating a diagram

Creating a class diagram

The  "Create a class diagram" icon

Various elements can have diagrams (packages, classes, scenarios, etc.).

Diagrams are created using the icons which appear in the "Diagrams" tab of the properties editor. In this tab, diagrams which already exist on model elements are also displayed.

We are now going to create a class diagram for the "Training" package (as shown in Figure 3-10).

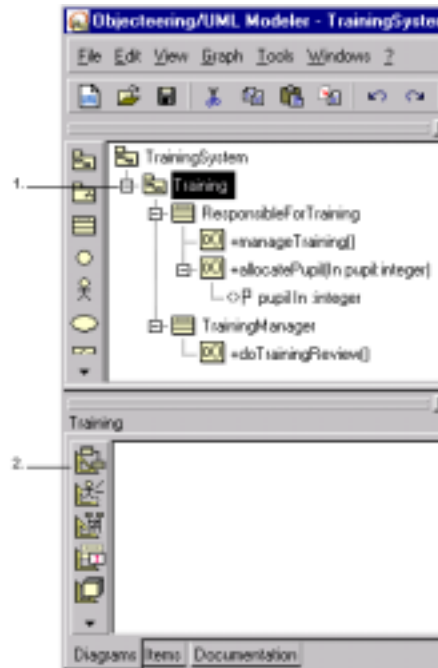



Figure 3-10. Creating a class diagram in the "Training" package

Chapter 3: First Steps


Steps:

- 1 - Select the "*Training*" package in the explorer.
- 2 - Click on the  "Create a class diagram" icon in the "*Diagrams*" tab of the properties editor.

The class diagram is then immediately displayed (Figure 3-11).

Note: When a new UML modeling project is created, a class diagram is automatically created and opened for the root element.

The "Show contents" option on elements in a diagram

The  "Show contents" icon

In the "Training" class diagram editor, not all graphic elements appear when the diagram is displayed. The "Show contents" operation is used to make elements which exist in the model appear in an editor.

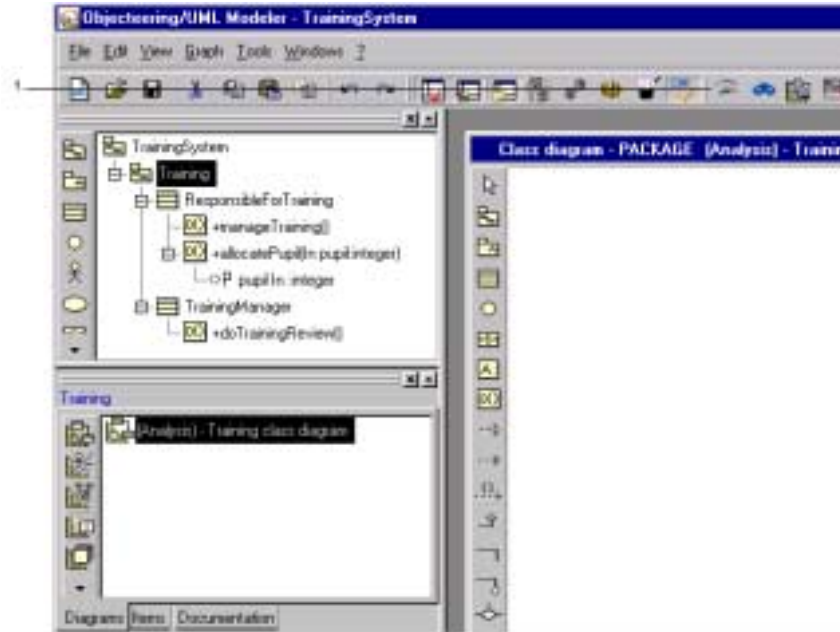


Figure 3-11. Class diagram of the "Training" package - Running the "Show contents" operation

Steps:

- 1 - Click on the  "Show contents" icon.

Result

The two "*ResponsibleForTraining*" and "*TrainingManager*" classes appear in the diagram (as shown in Figure 3-12). You can move them using the left mouse button.

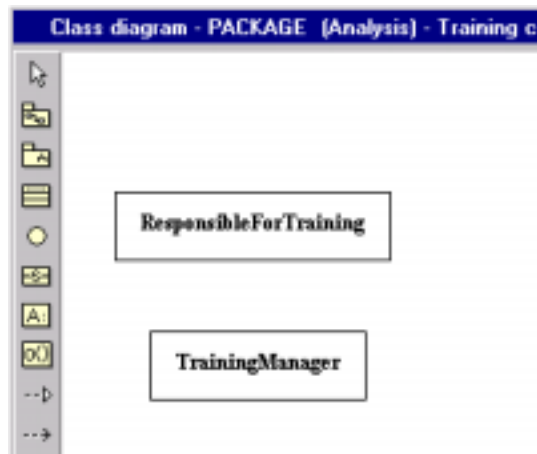



Figure 3-12. Unmasked classes in the class diagram

The "*Show contents*" operation visualizes package elements, but not their contents. To show the contents of an element, you must:

- 1 - Select the classes in the class diagram.
- 2 - Click on the  "*Show contents*" icon.

You can also display the contents of a single class, by activating the context menu on the class in question (using the right mouse button) and then selecting the "*Show contents*" option (Figure 3-13). The *Ctrl+E* keyboard shortcut activates this option.

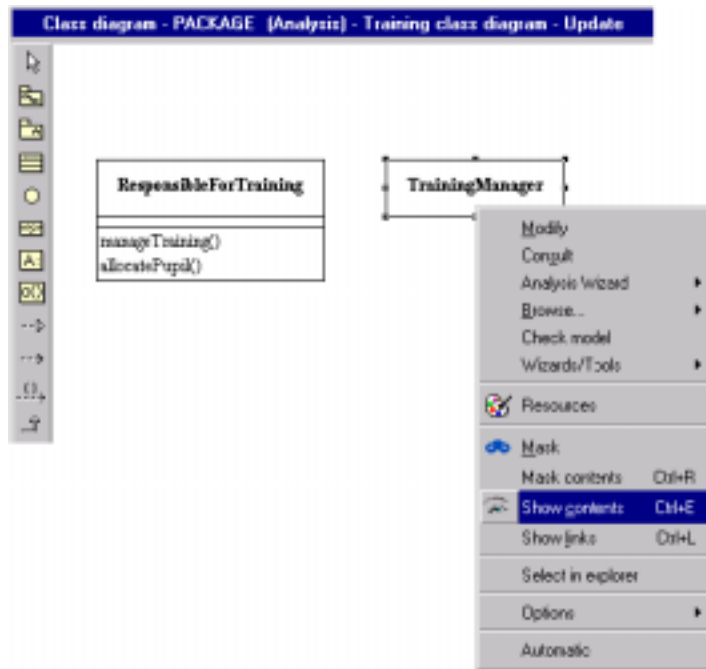


Figure 3-13. Running the "*Show contents*" option from the context menu available on the "*TrainingManager*" class

Masking an element

The  "Mask" icon

The mask function allows you to "hide" an element in a diagram. You can mask one or several elements (classes, links, operations, etc.).

In the example below (Figure 3-14), we are going to mask the "allocatePupil" operation of the "ResponsibleForTraining" class.

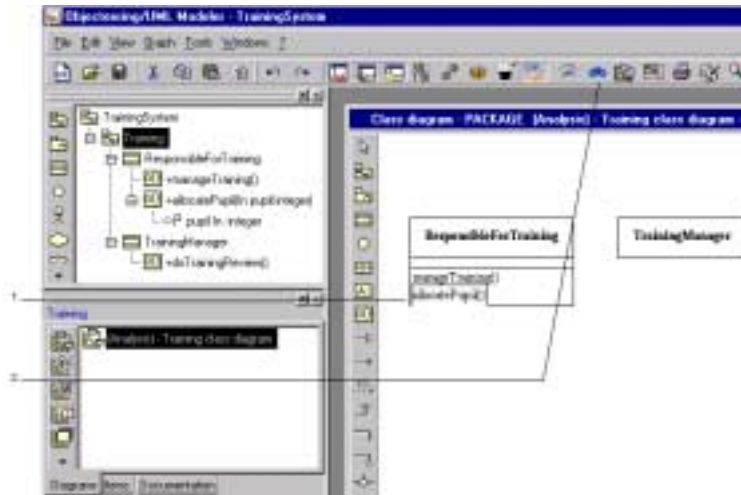



Figure 3-14. Masking the "allocatePupil" operation in the "ResponsibleForTraining" class

Steps:

- 1 - Select the "allocatePupil" operation.
- 2 - Click on the  "Mask" icon.

Note: Model elements appearing in a diagram can be masked using the context menu options. A keyboard shortcut, holding down the *Ctrl* and *R* keys, can also be used to mask the contents of an element.

Consistency and help mechanisms

Permanent consistency

All Objectteering/UML graphic editors ensure that model elements which appear are always consistent. For example, the name of an operation can appear in a class diagram, in an object diagram, in a sequence diagram and in the explorer. It will always be consistent.

We are now going to change the name of an operation in an editor. This name change will be automatically taken into account in the explorer.

- 1 - Click on the "*manageTraining*" operation using the right mouse button.
- 2 - Choose the "*Modify*" option from the context menu (right-clicking), or by double-click on the operation or indeed by activating the "*Edit/Modify*" menu.
- 3 - Enter the name in CAPITAL LETTERS.
- 4 - Confirm.
- 5 - Observe the explorer.

You will observe that the name of the operation in the explorer is in CAPITAL LETTERS.

The undo/redo mechanism

The  "*Undo*" icon

The  "*Redo*" icon

Objectteering/UML provides an unlimited undo/redo mechanism. For example,

let's carry out an "*undo*" operation by clicking on the  icon. As you can see, we get back the initial form of our "*manageTraining*" operation.

Copy/Paste

The  "Copy"  "Cut"  "Paste"  "Move" icons

The copy/paste mechanism is very practical when editing. In the explorer, we are now going to move the "doTrainingReview" operation into the "ResponsibleForTraining" class (as shown in Figure 3-15).

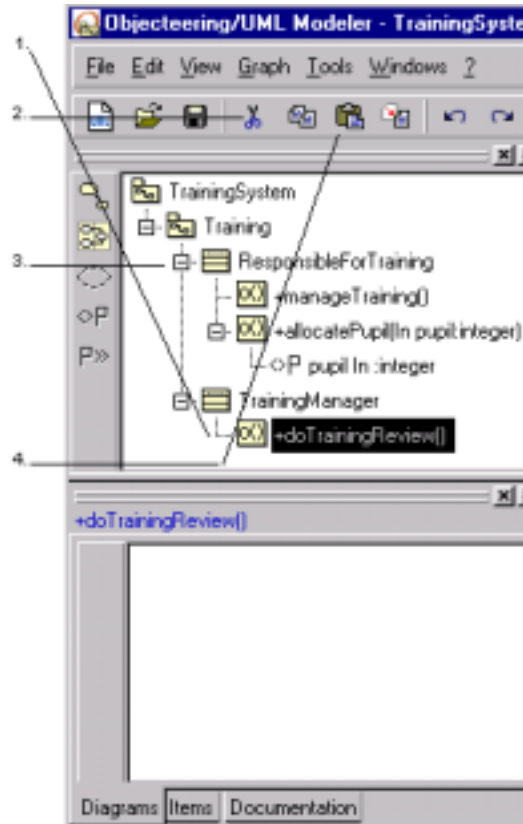





Figure 3-15. Moving the "doTrainingReview" operation of the "TrainingManagement" class to the "ResponsibleForTraining" class

Steps:


- 1 - Select the "*doTrainingReview*" operation.
- 2 - Click on the  "Cut" icon.
- 3 - Select the "*ResponsibleForTraining*" class.
- 4 - Click on the  "Paste" icon.

Cancel this action by carrying out two "*undo*" operations using the  icon in the explorer.

Note: The drag and drop function is used to do this more efficiently. Simply click on the "*doTrainingReview*" operation and drag it into the "*ResponsibleForTraining*" class.

Assisted data entry: Example of link creation

Creating a dependency

The  "Create a dependency" icon

Objecteering/UML has an assisted data entry mechanism for all operations. In this way, errors during data entry can be avoided. In the following example (shown in Figure 3-16), let's take a look at how a dependency can be created by using the assisted data entry mechanism.

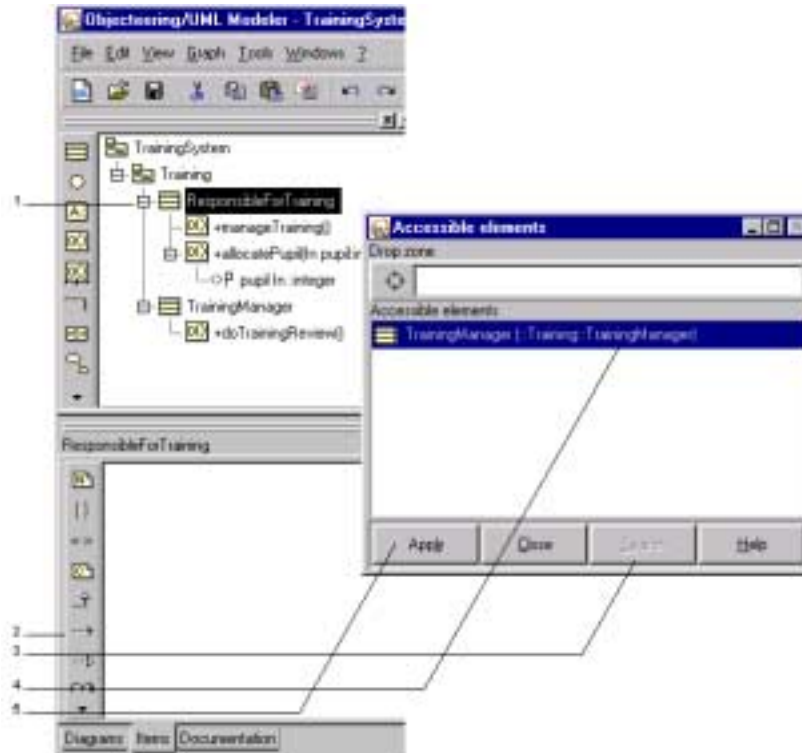




Figure 3-16. Creating a dependency between two classes

Steps:

- 1 - Select the "*ResponsibleForTraining*" class.
- 2 - Click on the  "Create a dependency" icon.
- 3 - Click on the "Search" button. The list of accessible elements appears in the window.
- 4 - Double-click on the accessible element.
- 5 - Click on the "Apply" button to apply this operation.

Result: The dependency is displayed in the "*Items*" tab of the properties editor.

It is possible to use drag and drop for this operation. After having selected the "ResponsibleForTraining" class and clicked on the  "Create a dependency" icon, proceed as follows (as shown in Figure 3-17):

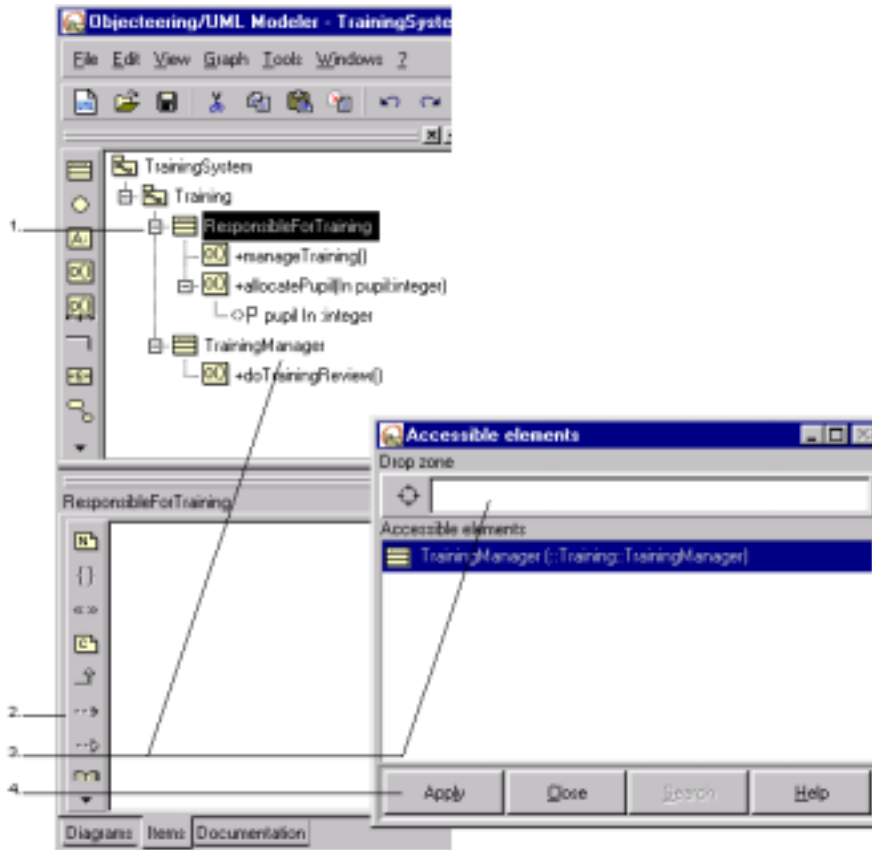



Figure 3-17. Creating a dependency using the drag and drop function

Steps:

- 1 - Select the "*ResponsibleForTraining*" class.
- 2 - Click on the  "Create a dependency" icon.
- 3 - Then select the "*TrainingManager*" class by left-clicking in the explorer. Hold down the left mouse button and drag the "*TrainingManager*" class into the drop zone in the accessible elements window.
- 4 - Release the "*TrainingManager*" class in the drop zone, and then confirm by clicking on the "*Apply*" button.

Showing a link through the context menu

The dependency does not appear in the graphic editor because it is masked. We are now going to show it, by using the "Show links" function in the context menu on the class which contains this link (as shown in Figure 3-18):

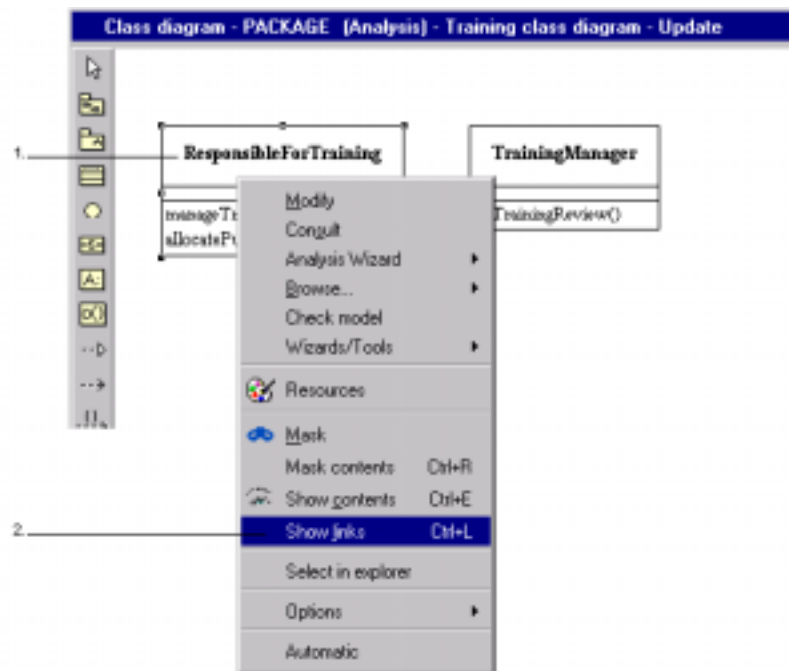


Figure 3-18. Showing the dependency on the "ResponsibleForTraining" class

Steps:

- 1 - In the class diagram, click on the "ResponsibleForTraining" class using the right mouse button.
- 2 - Choose the "Show links" option from the context menu.

Note: Links can be shown using the "Ctrl + L" keyboard shortcut.

Drawing links

In the graphic editor, you can draw links using the "*Redraw link*" option from the context menu. We are now going to redraw the dependency. Proceed as follows (shown in Figure 3-19):

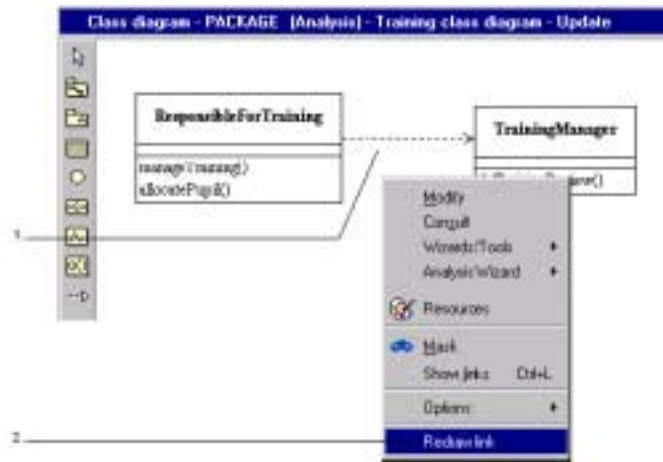


Figure 3-19. Redrawing the dependency


Steps:

- 1 - Select the dependency by right-clicking.
- 2 - Choose the "*Redraw link*" option from the context menu.
- 3 - Redraw the dependency from the origin class towards the destination class (in several points, for example).

Note 1: For further information on redefining links, please refer to chapter 5 of the *Objecteering/UML Modeler* user guide.

Note 2: If your mouse has a middle button, you can also select the dependency by middle-clicking. You may then redraw the link.

Consistency checks

The  "Error message"

Objecteering/UML carries out more than 200 consistency checks in real time on your model (an example is shown in Figure 3-20). This guarantees the highly professional quality of models built in the Objecteering/UML CASE tool.

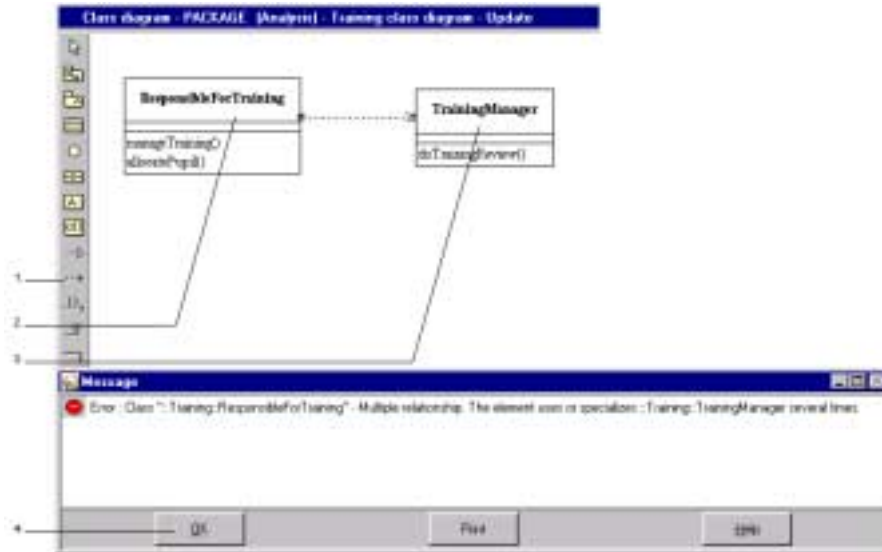



Figure 3-20. Example of consistency checking

Steps:

- 1 - Click on the  "Create a dependency" icon.
- 2 - Select the "ResponsibleForTraining" class.
- 3 - Select the "TrainingManager" class.
- 4 - Error message details are given in the console, and a beep is heard. Click on the "OK" button to confirm.

This error also occurs if you try to carry out the dependency creation operation using the drag and drop function. In this case, the "*TrainingManager*" class does not appear in the "*Accessible elements*" window when you click on the "*Search*" button.

Note: If you wish to print a description of the error, simply click on the "*Print*" button. For help regarding the error, click on the "*Help*" button.

Completing our first model

Creating a class in a diagram

The  "Create a class" icon

Just like in the explorer, classes may be created in the graphic editors. The classes created appear at exactly the selected position (as shown in Figure 3-21).

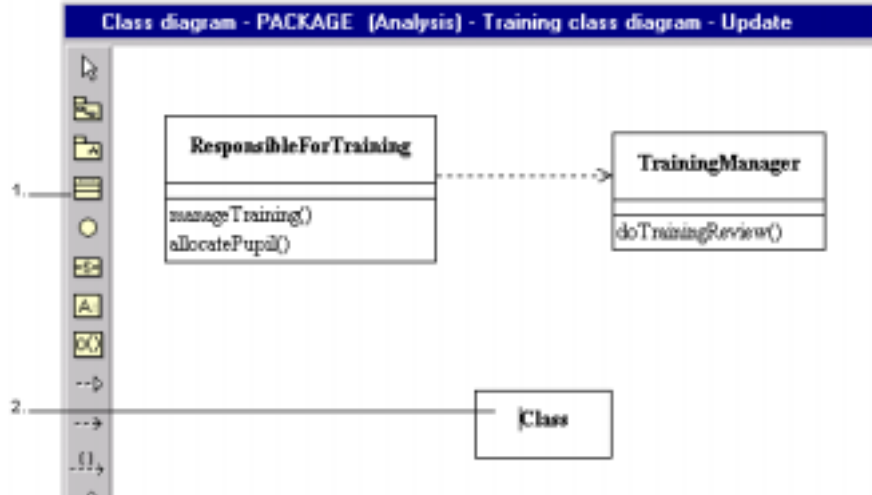




Figure 3-21. Creating a new class ("TrainingSession") in the graphic editor

Steps:

- 1 - Click on the  "Create a class" icon in the graphic editor.
- 2 - Click where you want to place the class, and enter the "TrainingSession" name over the highlighted text.
- 3 - Click outside the class to confirm.

Note: Please note that the "TrainingSession" class also appears in the explorer.

Creating an association between two classes

The  "Create an association" icon

We are now going to create an association between the "TrainingManager" class and the "TrainingSession" class (as shown in Figure 3-22):

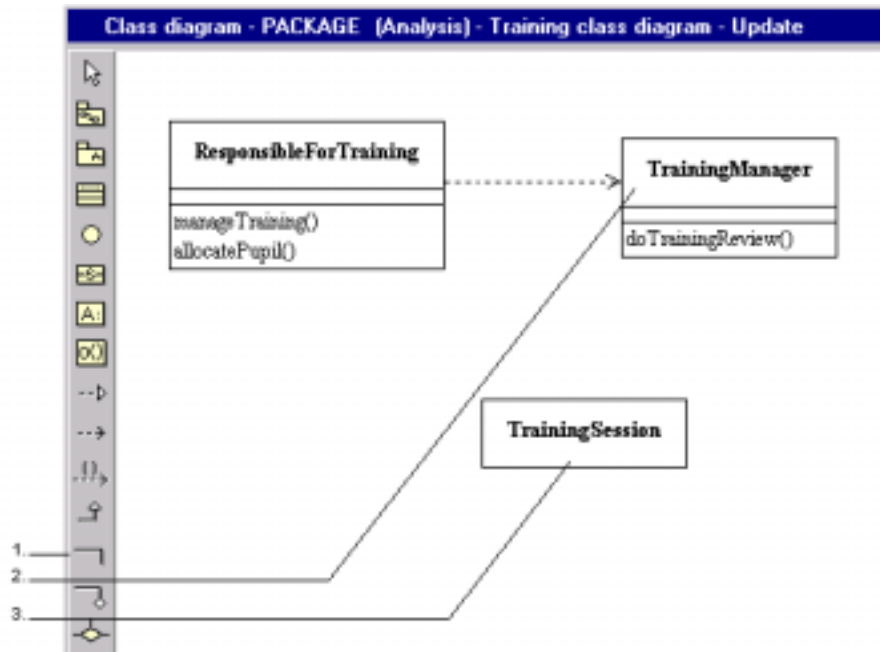



Figure 3-22. Creating an association between two classes

Steps:

- 1 - Click on the  "Create an association" icon.
- 2 - Click on the "TrainingManager" class.
- 3 - Click on the "TrainingSession" class.


Chapter 3: First Steps

To redraw the association, select it using the right-mouse button, and choose the "*Redraw link*" option from the context menu.

Note 1: When the association is created, you can directly control the drawing of the link by holding down the shift key on your keyboard.

Note 2: The shift key allows you to draw links containing right angles during link redefinition or creation.

Creating a note in a class diagram

The  "Add a note" icon

Notes can be created:

- ◆ in the "Items" tab of the properties editor, in the same way as tagged values and constraints, amongst other things
- ◆ directly in graphic editors

For our example, we are now going to create a note in our class diagram (Figure 3-23), which will add information to our model.

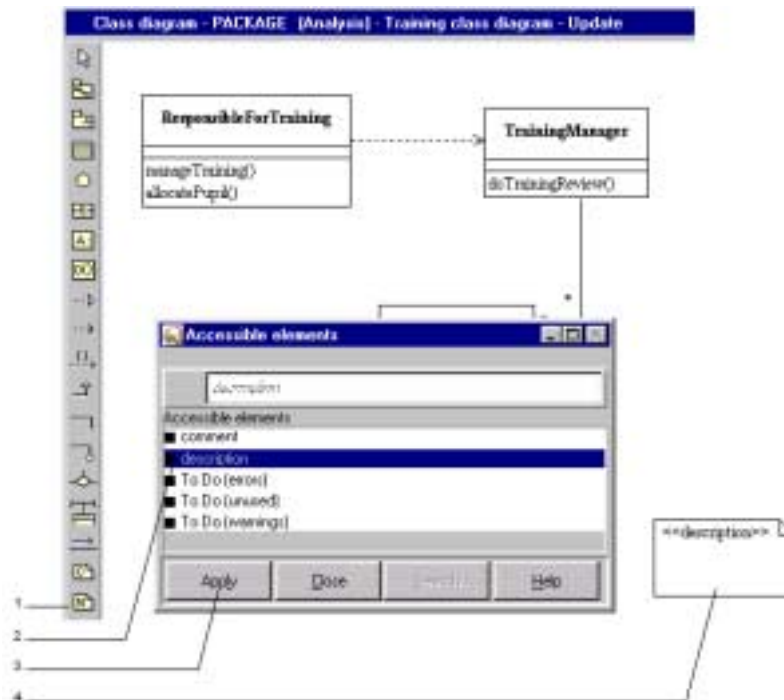


Figure 3-23. Creating a note in a class diagram

Chapter 3: First Steps

Steps:


- 1 - Click on the  "Create a note" icon and click in the diagram, at the point where you wish the note to appear.
- 2 - The "Accessible elements" window then appears. Choose the *description* type (this type is only available when the *Objectteering/Documentation* module has been installed).
- 3 - Confirm.
- 4 - Enter the following text directly in the graphic element: "*Diagram which presents the organization of the UML Training*".

Diagram notes are not visible from the explorer. To make them visible, you must select the diagram in the "Diagrams" tab of the properties editor using the right-mouse button, and select the "Modify" option from the context menu.

The "Note" tab of the displayed window will indicate the presence of the description note. (Figure 3-24)

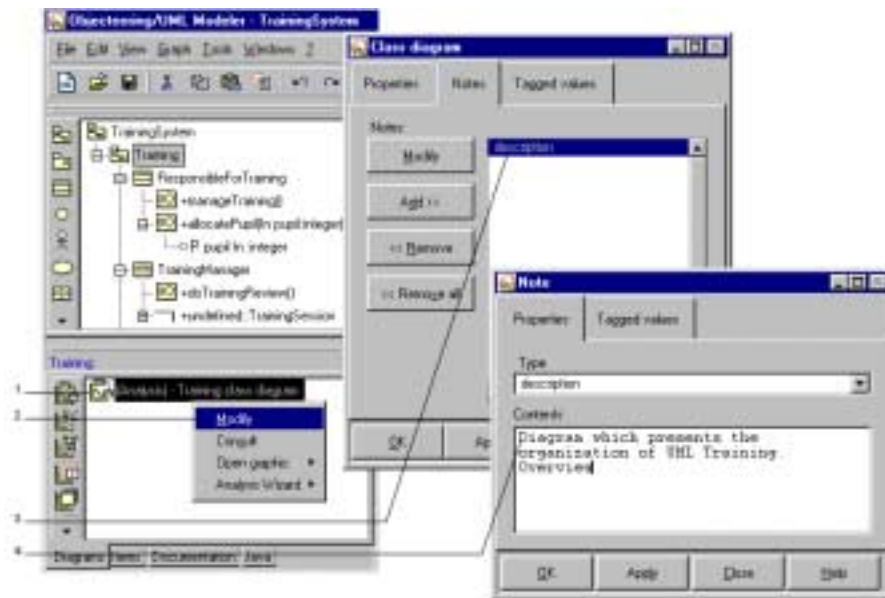


Figure 3-24. Editing the class diagram "Note" dialog box from the explorer

Steps:

- 1 - In the "Diagrams" tab of the properties editor, select the "Training class diagram" diagram.
- 2 - Choose the "Modify" option from the context menu.
- 3 - In the "Notes" tab of the "Class diagram" dialog box which appears, double-click on description.
- 4 - Add the word "Overview" in the "Content" zone.

Note: You will observe that the text of the description note has been automatically modified in the diagram (consistency mechanism).

Entering association values

Association values may be entered either in the edited dialog box or directly in the diagram (Figure 3-25).

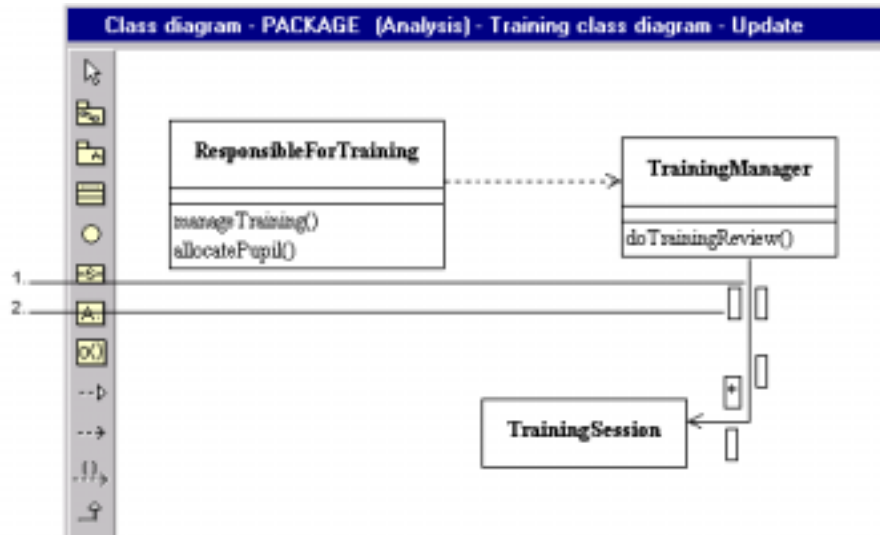


Figure 3-25. Direct entry of association values in a diagram

Steps:

- 1 - Click once, using the left mouse button, and then a second time. The entry fields will appear.
- 2 - Now modify the association name, roles and multiplicities as follows:
 - ◆ Association name: "Management"
 - ◆ Role of the "TrainingManager" class: "manager"
 - ◆ Role of the "TrainingSession" class: "managed"
 - ◆ Multiplicity of the "TrainingManager" class: 0..1
 - ◆ Multiplicity of the "TrainingSession" class: 1

Editing the binary association dialog box

If you wish to enter association attributes in the entry dialog box, display this window by double-clicking or by activating the context menu on the association (right-clicking on the mouse) and selecting the "Modify" option (Figure 3-26).

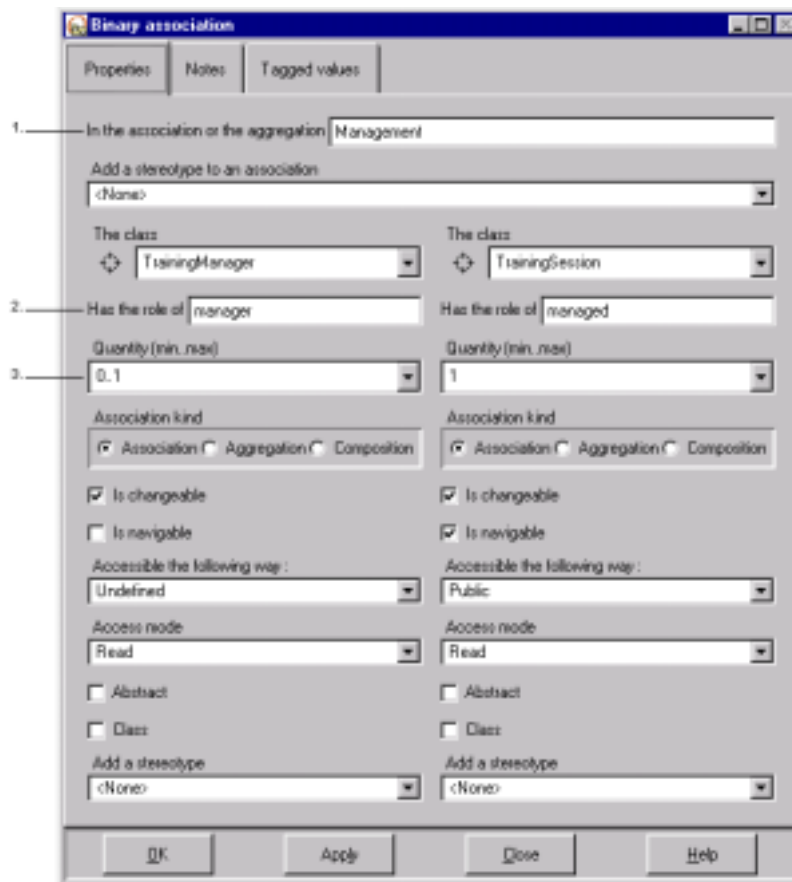


Figure 3-26. Editing the binary association dialog box

Chapter 3: First Steps

Key:

- 1 - Entry field for the name of the association
- 2 - Entry field for association roles
- 3 - Entry field for multiplicities

Note: For further information on this dialog box, please refer to the *Objecteering/Model Dialog Boxes* user guide.

Result: model created

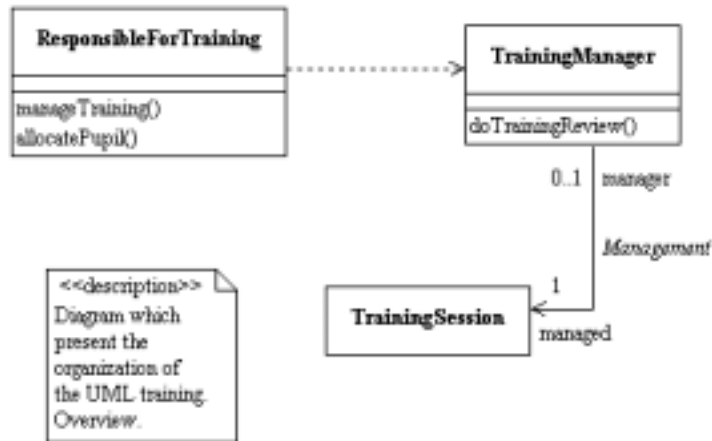



Figure 3-27. Graphic result of our model

Exit the graphic editor and save your work.

Organization

Creating a package

The  "Create a package" icon

The explorer allows us to organize and structure our model, by arranging classes into different packages (as shown in Figure 3-28).

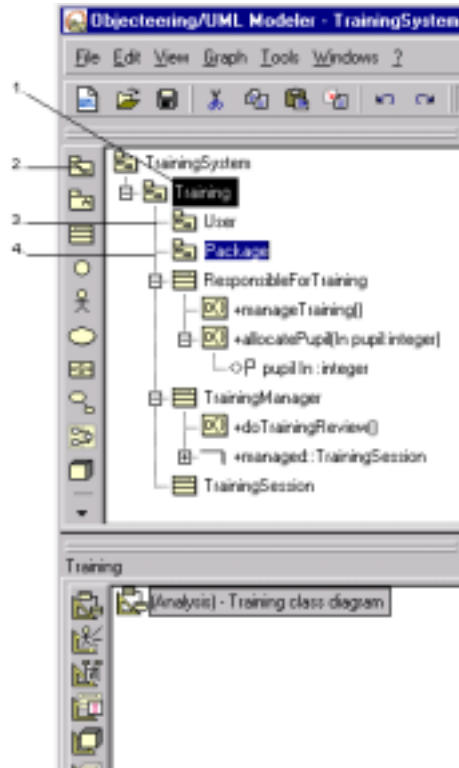



Figure 3-28. Creating the "User" and "TrainingManagement" packages in the "Training" package

Chapter 3: First Steps

Steps:

- 1 - Select the "*Training*" package.
- 2 - Click on the  "*Create a package*" icon.
- 3 - Enter the name of the "*User*" package over the highlighted text, and then press "*Return*" on the keyboard.
- 4 - Enter the name of the second package ("*TrainingManagement*"). Left-clicking on the mouse stops the continuous entry system.

Organizing classes

The  "Copy" icon

The  "Move" icon

It is possible to modify model organization by re-allocating packages and classes to other packages, using the "copy" and "move" actions.

The drag and drop mechanism allows you to carry out this type of operation (as shown in Figure 3-29) even more efficiently.

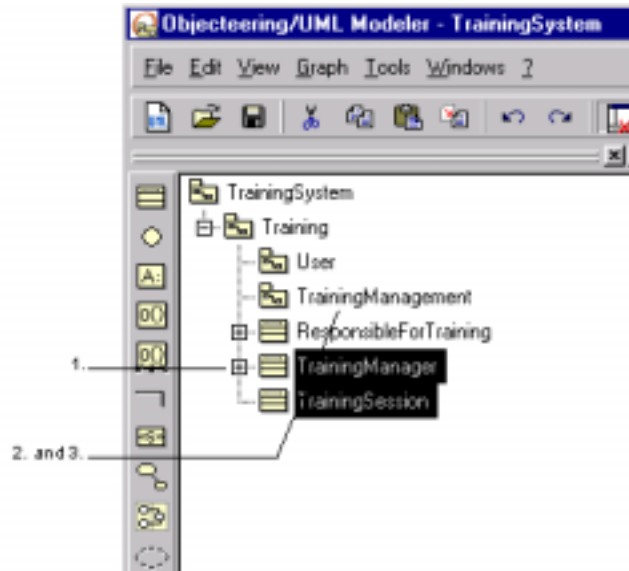


Figure 3-29. Moving the "TrainingManager" and "TrainingSession" classes to the "TrainingManagement" package


Chapter 3: First Steps

Steps:

- 1 - Select the "*TrainingManager*" and "*TrainingSession*" classes. (The "*Shift*" button can be used to make multiple selections).
- 2 - Drag the classes into the "*TrainingManagement*" package, by selecting these two classes and moving them, whilst holding down the left mouse button.
- 3 - Release the mouse button.

You are now going to move the "*ResponsibleForTraining*" class into the "*User*" package.

An error message appears. In effect, the "*User*" package has to use the "*TrainingManagement*" package.

You are therefore going to create the  dependency from "User" to "TrainingManagement" (Figure 3-30).

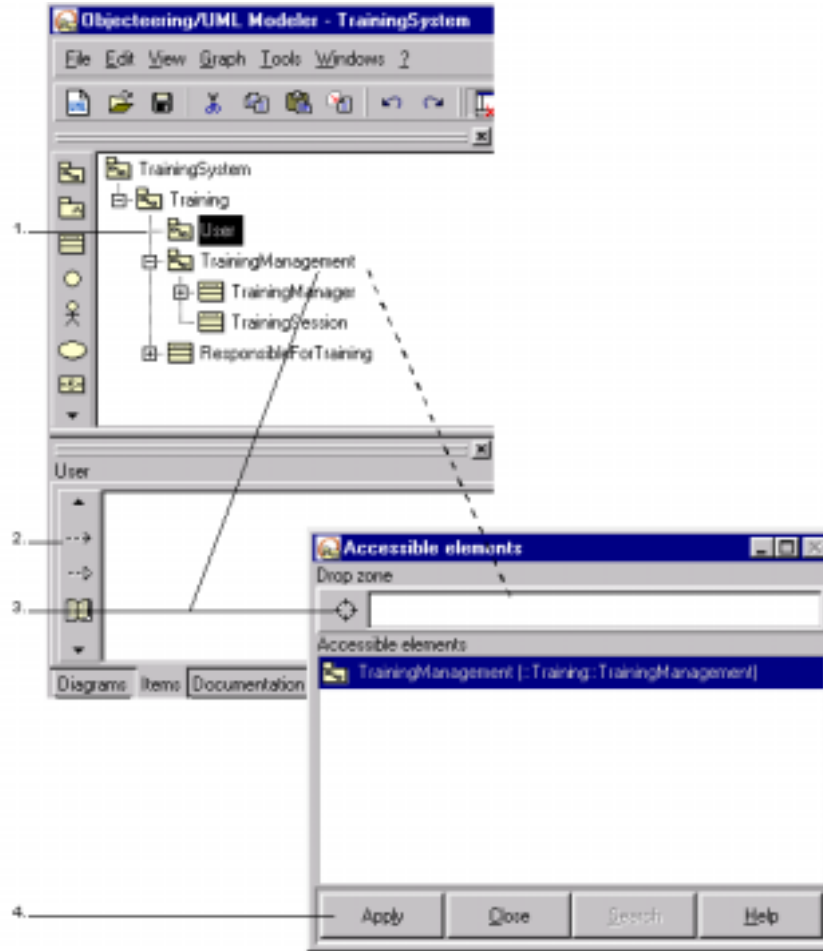



Figure 3-30. Creating the dependency between "User" and "TrainingManagement"

Chapter 3: First Steps

Steps:

- 1 - Select the "User" package.
- 2 - Click on the  icon.
- 3 - Drag the "TrainingManagement" package into the drop zone of the accessible element window.
- 4 - Confirm.

Now move the "ResponsibleForTraining" class into the "User" package.

Result of the operation in the class diagram

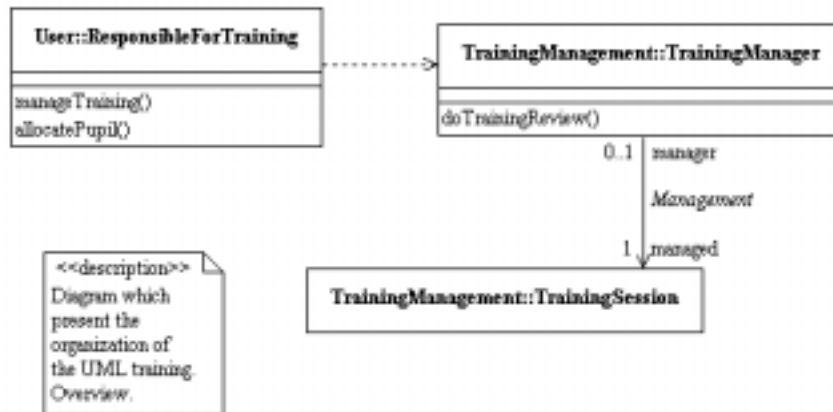





Figure 3-31. Result of the reorganization of the classes

To open the class diagram once again from the explorer, carry out the following steps:

- 1 - Select the  "Training" package.
- 2 - Double-click on the  "Training class diagram" class diagram in the "Diagrams" tab of the properties editor.

Note: In the diagram, classes are prefixed with the name of their owner packages through the selection of the "detailed visibility" mode. To activate or deactivate this mode, simply right-click on a diagram element and then select "Options/Detailed visibility" from the context menu.

Entering operations on the "TrainingSession" class

The  "Create an operation" icon

We are now going to create the "create", "confirm", "start" and "end" operations in the "TrainingSession" class, by using the repeated entry system in a diagram (Figure 3-32):

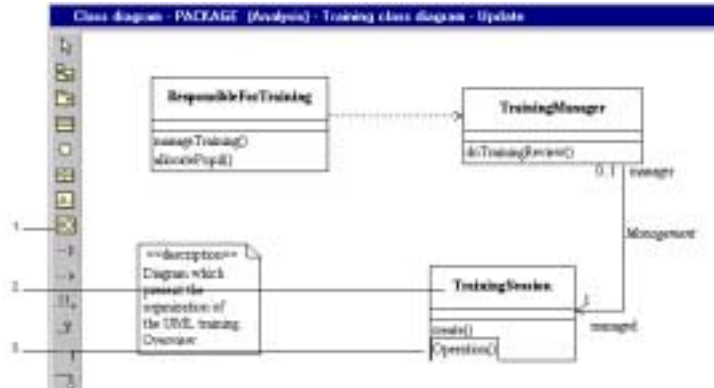




Figure 3-32. Entering the "create", "confirm", "start" and "end" operations in the "TrainingSession" class in a class diagram


Steps:

- 1 - Click on the  "Create an operation" icon, whilst holding down the "Ctrl" key on your keyboard or double-click on the icon, in order to activate the repeated entry system.
- 2 - Click in the "TrainingSession" class.
- 3 - Enter the name of the "create" operation over the highlighted text ("Operation"), and then click on the left mouse button over the "TrainingSession" class. A second operation is created. Enter the name of the "confirm" operation.
- 4 - Continue in this way by creating the "start" and "end" operations.
- 5 - Click on the  "Curser" icon to stop the repeated entry action.


Showing elements in a diagram

Showing elements in a diagram

There are three ways of showing elements in a diagram:

- 1 - The first way is to use the  "Show contents" icon (already used here).
- 2 - The second method is to use the context menu available on elements and to use the "show" functions proposed.
- 3 - The third option is to use the drag and drop function, by dragging elements from the explorer to the diagram.

Showing elements using the "Show contents" icon

The  "Show contents" icon

First of all, we are going to illustrate how elements can be shown, by using the "TrainingSystem" package, which was automatically created on the "TrainingSystem" UML model root when the UML modeling project was first created.

- 1 - Select the "TrainingSystem" package in the explorer.
- 2 - Double-click on the "TrainingSystem class diagram" diagram in the "Diagrams" tab of the properties editor.

When this diagram is opened, no elements of the "TrainingSystem" package are visible, therefore we are going to "show" all its elements (Figure 3-33).

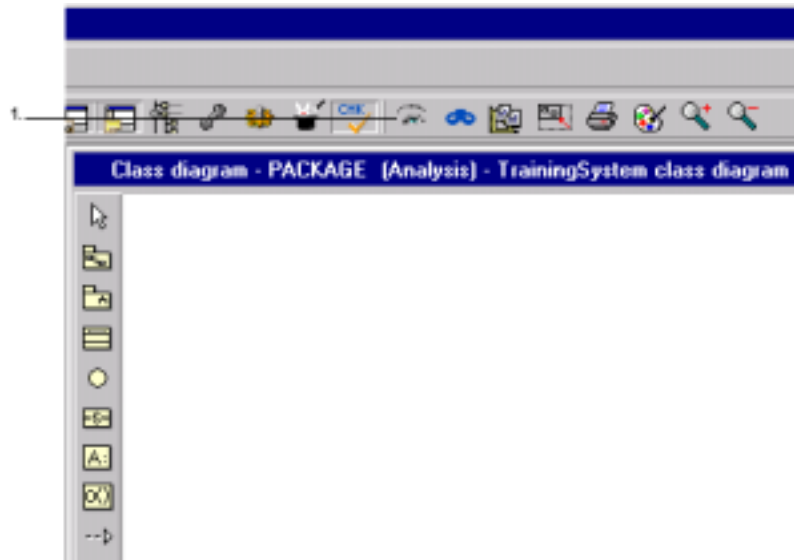


Figure 3-33. Showing the contents of the elements of the "TrainingSystem" package

Steps:

- 1 - Click on the  "Show contents" icon.

You will notice that this action has only revealed the "Training" package. Our aim is to see the whole model for this package. We are, therefore, going to show the contents of the "Training" package (Figure 3-34).

Note: If you do not see the "Training" package, and see only a note, this is because the note is hiding the package. Simply drag the note to a different part of the window to see the package.

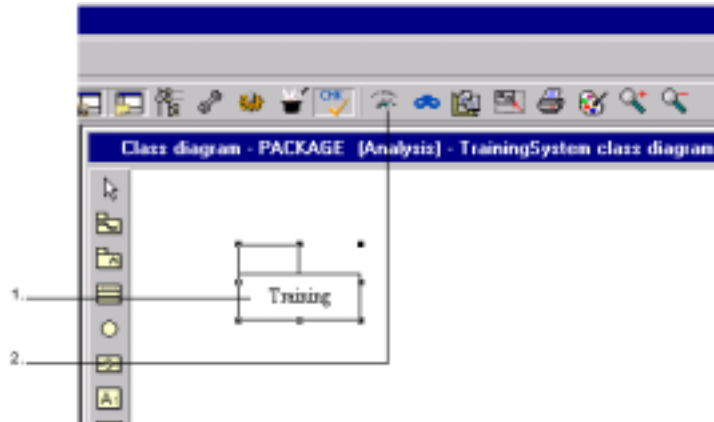



Figure 3-34. Showing the "Training" package

Steps:

- 1 - Select the "Training" package, by left-clicking.
- 2 - Click on the  "Show contents" icon.

The last action presents the contents of the "Training" package, which are the "User" and "TrainingManagement" packages.

Showing a link through the context menu

It is possible to show the contents of an element in a diagram, by activating the context menu on the element in question (by right-clicking over the element).

In our example, we are now going to show the link between the "User" and "TrainingManagement" packages (Figure 3-35).

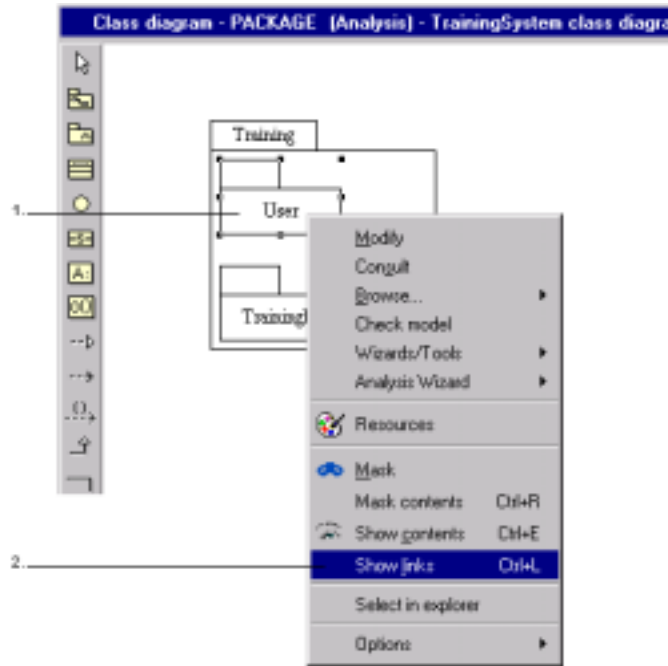


Figure 3-35. Showing the links on the "User" package


Steps:

- 1 - Click on the "User" package using the right mouse button.
- 2 - Choose the "Show links" option.

Note: The *Ctrl + L* keyboard shortcut can be used to show an element's links.

Showing a link through the drag and drop function

We shall first create another class diagram in the "TrainingSystem" package.

- 1 - Select the "TrainingSystem" package in the explorer.
- 2 - Click on the  "Create a class diagram" icon in the "Diagrams" tab of the properties editor.

In this new diagram, we are going to show the dependency between the "User" and "TrainingManagement" packages, by using the drag and drop function (Figure 3-36).

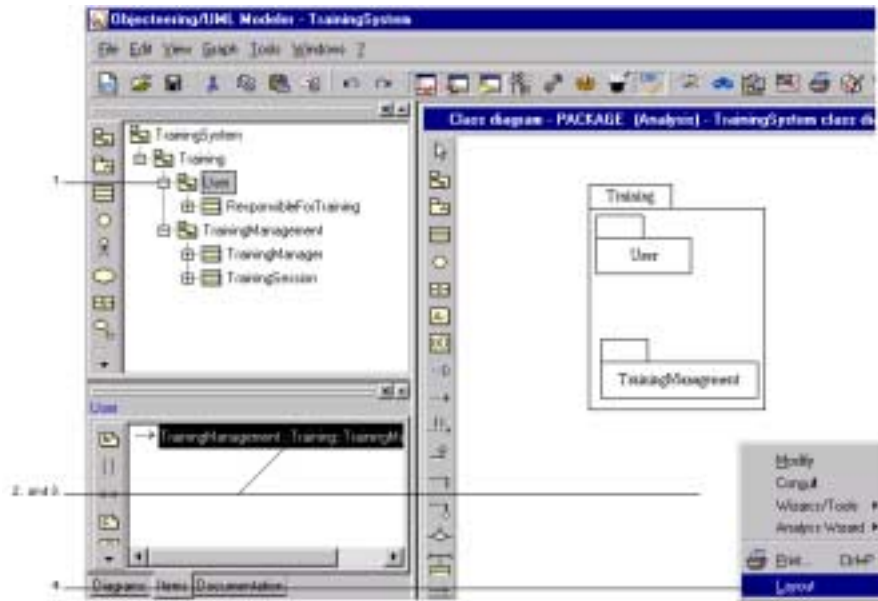



Figure 3-36. Showing a dependency using drag and drop between two packages

Chapter 3: First Steps

Steps:

- 1 - In the explorer, select the "User" package.
- 2 - Select the  dependency situated in the "Items" tab of the properties editor by clicking on the left mouse button, and move this link from the properties editor to the diagram.
- 3 - Release the link in the diagram. The link between the "User" and "TrainingManagement" packages is now shown.
- 4 - Use the general "Graph/Layout" menu, or select the "Layout" option from the pop-up menu (right-click in a diagram zone). The diagram is automatically positioned.

Result

Figure 3-37 presents two different ways of showing a dependency.

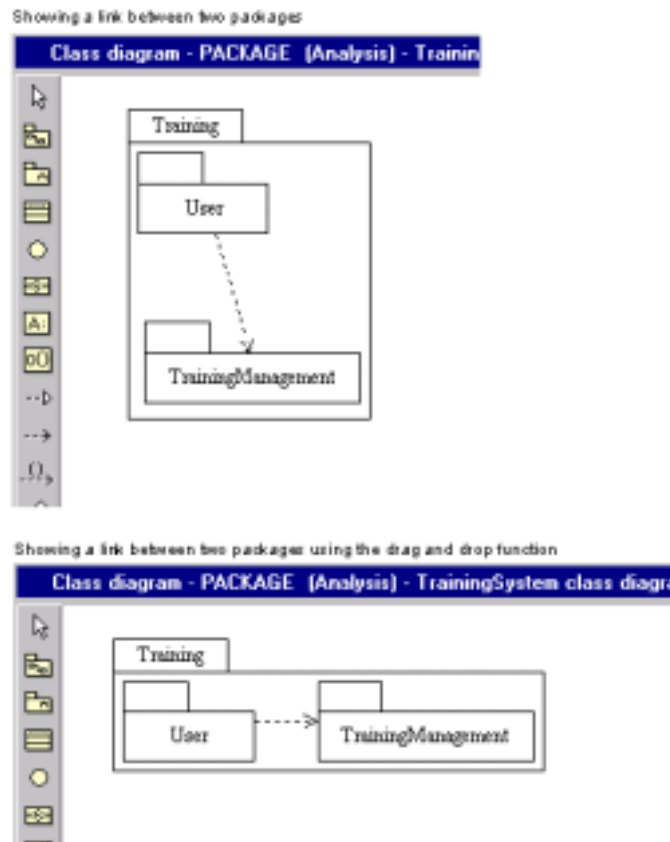



Figure 3-37. General views of model organization: two ways of showing links

Creating a sequence diagram

Creating from the explorer

The  "Create a sequence diagram" icon

A sequence diagram can be created for a package, a class or a collaboration.

We are now going to create a sequence diagram for the "User" package (Figure 3-38).

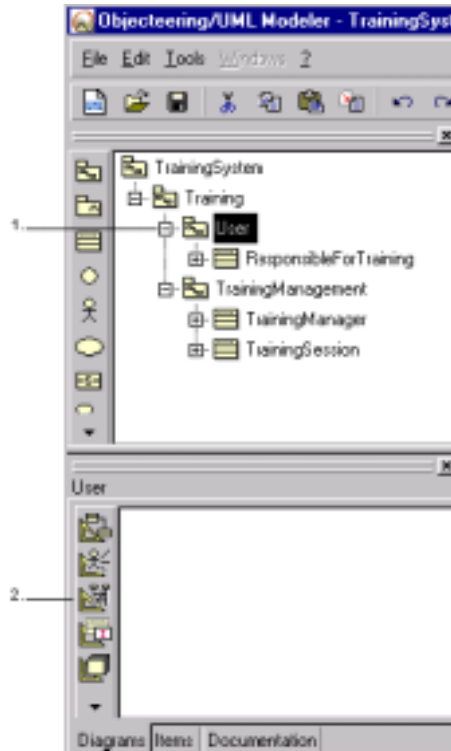




Figure 3-38. Creating a sequence diagram in the "User" package

Steps:

- 1 - Select the "User" package in the explorer.
- 2 - Click on the  "Create a sequence diagram" icon in the "Diagrams" tab of the properties editor. The newly created sequence diagram then opens automatically.

Creating an instance in the sequence diagram graphic editor

The  "Create an instance" icon

Sequence objects can have a name and represent a class (as shown in Figure 3-39).

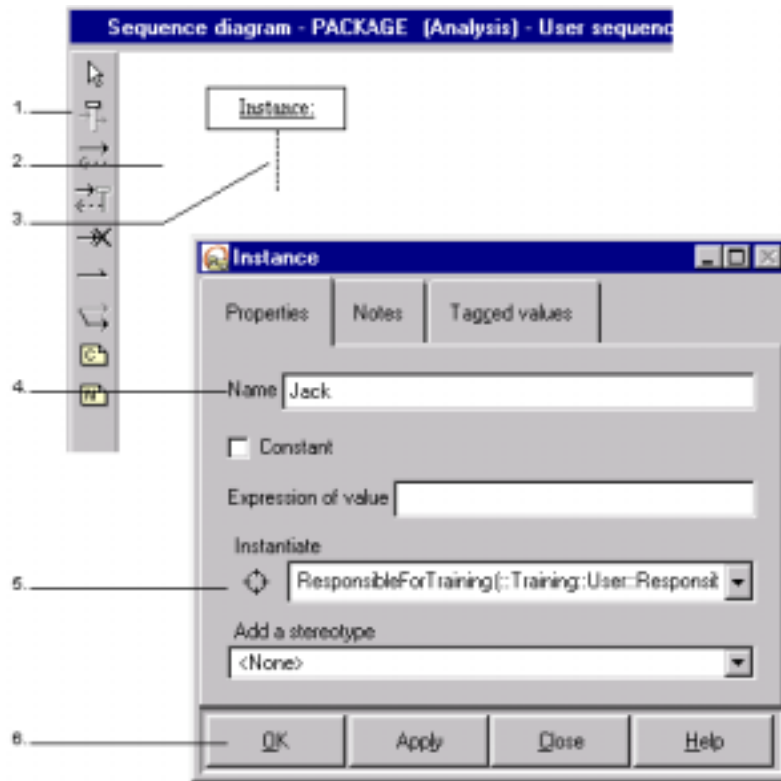



Figure 3-39. Creating the "Jack" instance

Steps:

- 1 - Click on the  "Create an instance " icon.
- 2 - Click in the diagram.
- 3 - Double-click on the object.
- 4 - Enter the name of the "Jack" object.
- 5 - Select "User:ResponsibleForTraining" from the "Instantiate" list.
- 6 - Confirm.


Note: You can enter or modify the name of the object without opening the dialog box, simply by clicking over its name.

In the same way, create another object ("Session3") with "TrainingSession" as the referenced class.

Note: If you wish to move your objects, you will notice that you can only move them horizontally.

Note: It can be impractical to display package names which prefix class names. If this is the case, display the diagram dialog box (by right-clicking in the graphic to display the context menu, and then selecting the "modify" option), and de-activate the "Detailed visibility" option.

Creating a reflexive message from an instance

The  "Create a sequence message" icon

Messages can have a name or be associated with one of the receiving object class operations (as shown in Figure 3-40).

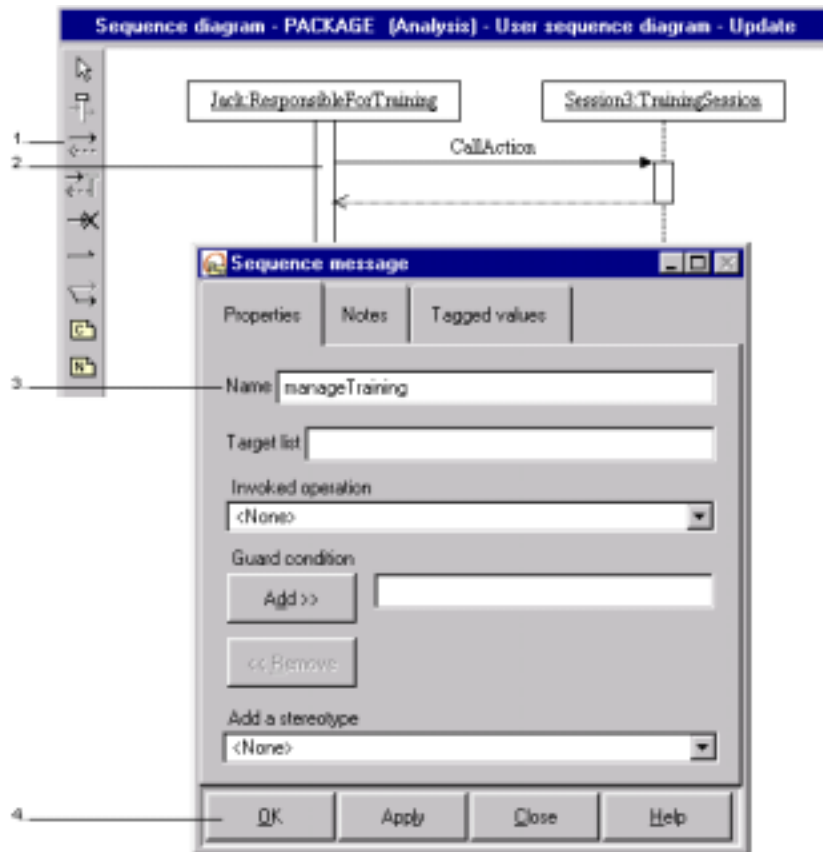



Figure 3-40. Creating the "manageTraining" message

Steps:

- 1 - Click on the  "Create a sequence message" icon, and then click first on the "Session3" object and then on the "Jack" object.
- 2 - Double-click on the message (link), and a dialog box then appears.
- 3 - Enter "manageTraining" in the "Name" field.
- 4 - Confirm.

In the same way, create another message ("allocatePupil") from the same object, and then a "confirm" message between from "Jack" to "Session3", as well as an "end" message, also from "Jack" to "Session3".

Result




Figure 3-41. Sequence diagram for the "User" sequence object

Note 1: Here, we have an example of assisted data entry, which allows you to reference a message's operation.


Note 2: It is not necessary to enter a name where an associated operation exists.

Creating a state diagram

Creating a state in from the explorer

The  "Create a state diagram" icon

Before creating a state diagram, you must first create a state machine for a class, as shown in Figure 3-42.

The  "Associate a state machine" icon

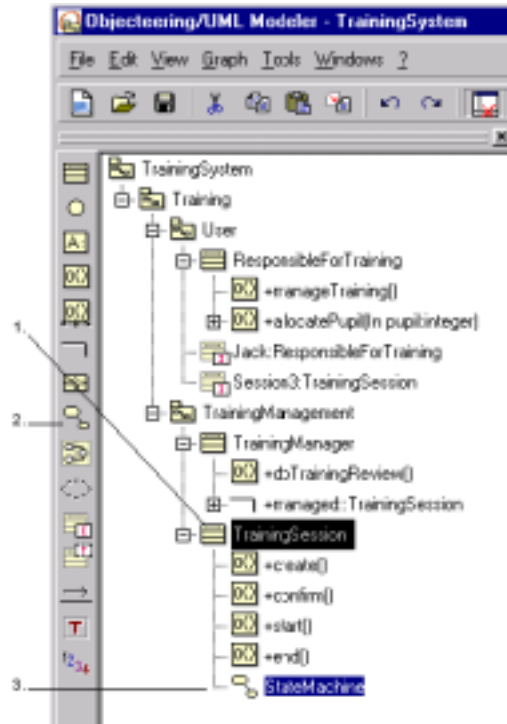



Figure 3-42. Creating the "Training" state machine

Chapter 3: First Steps

Steps:

- 1 - Select the "*TrainingSession*" class.
- 2 - Click on the  "Associate a state machine" icon.
- 3 - Enter the "*Training*" name over the highlighted text.

Note: The procedure for the creation of an activity diagram is similar, in that an activity graph must first be created for the chosen element, before proceeding with the creation of the diagram itself.

Creating a state diagram from the properties editor

We are now going to create a state diagram for the "Training" state machine (Figure 3-43).

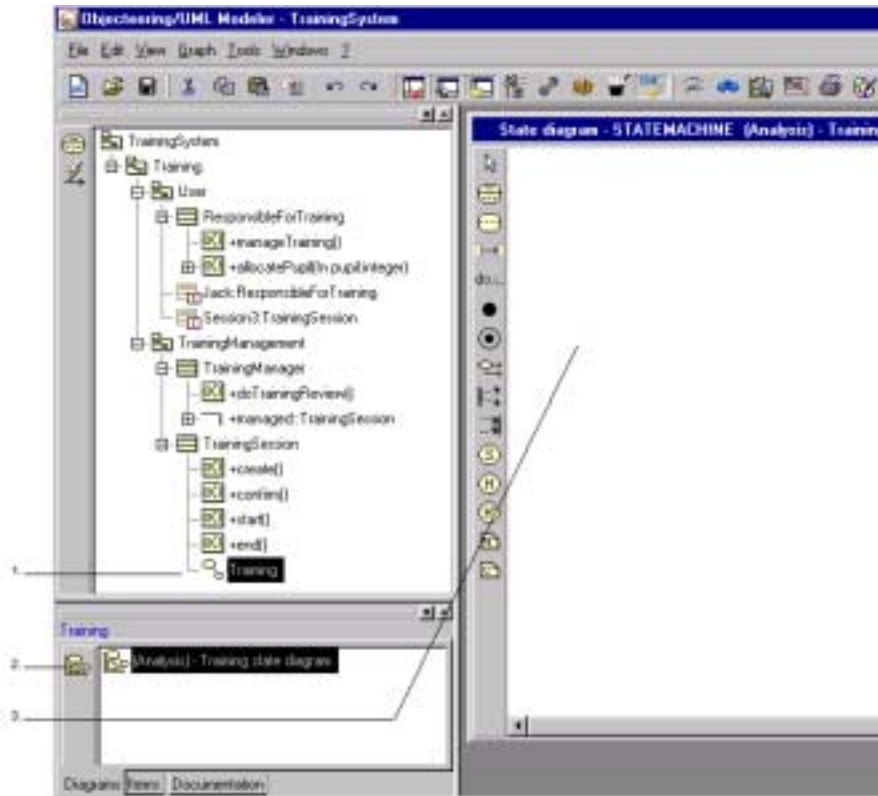



Figure 3-43. Creating a state diagram

Chapter 3: First Steps

Steps:

- 1 - Select the "*Training*" state machine in the explorer.
- 2 - Click on the  "Create a state diagram" icon in the "Diagrams" tab of the properties editor.
- 3 - The newly created state diagram is then immediately opened.

Creating a state

The  "Create a state" icon

States can be created in diagrams or in the explorer. We are going to create for our example the "defined", "confirm", "inProgress" and "done" states in our state diagram (Figure 3-44).

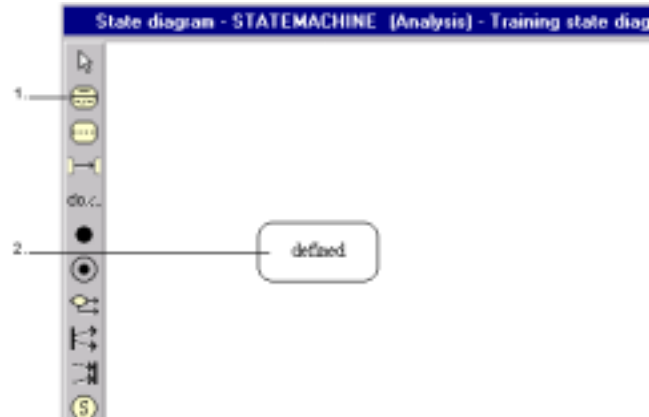




Figure 3-44. Creating the "defined" state


Steps:

- 1 - Click on the  "Create a state" icon.
- 2 - Click in the diagram and enter the name of the "defined" state over the highlighted text.

In the same way, create the "confirm", "inProgress" and "done" states.

Note: By clicking on the "Ctrl" key on your keyboard during the selection of the  "Create a state" icon, or by double-clicking on the same icon, you will be able to create states using the repeated entry mechanism.

Creating transitions

The  "Create a transition" icon

Transitions can be associated with one of the operations of the current class.

We are going to create the first transition from an initial state that we will have to create (as shown in Figure 3-45). It will be used as the starting point for our model.

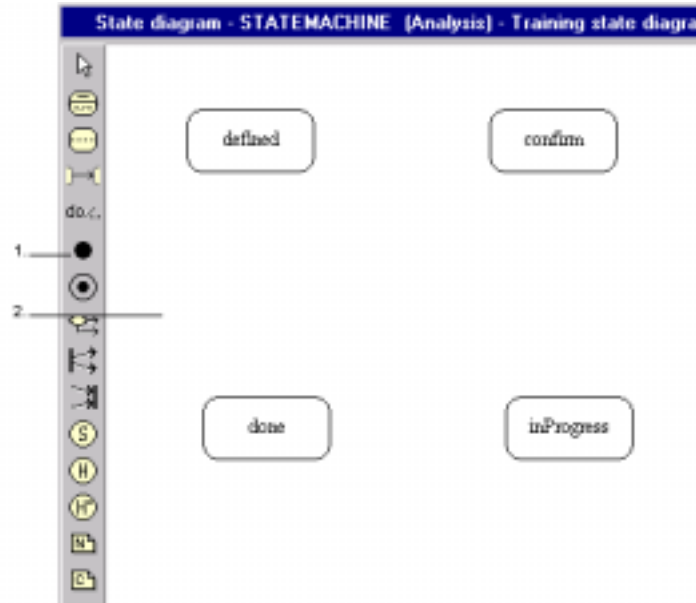



Figure 3-45. Creating an initial state

Steps:

- 1 - Click on the  "Create an initial state" icon.
- 2 - Click on the spot where you wish to place this state.

Now create your first transition starting from this point (as shown in Figure 3-46), towards the "defined" object.

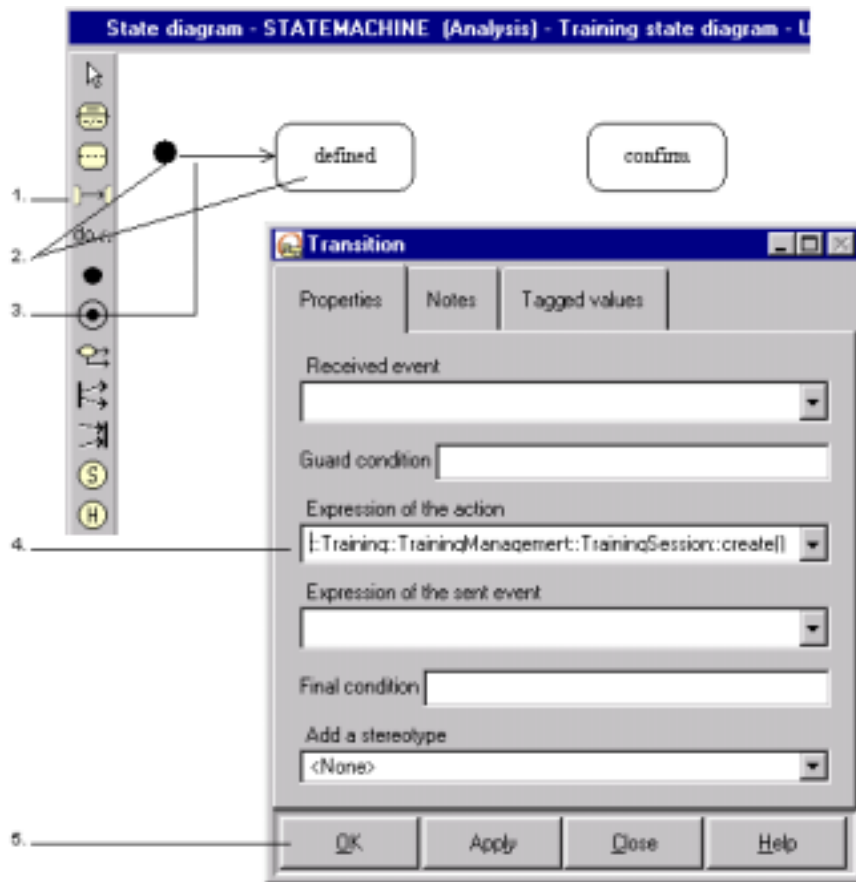



Figure 3-46. Creating a simple transition

Chapter 3: First Steps

Steps:

- 1 - Click on the  "Create a transition" icon.
- 2 - Click on the initial state and then on the "defined" state.
- 3 - Double-click on the transition to display its dialog box.
- 4 - Select the expression of the "create" action in the scrolling list ("*Expression of the action*").
- 5 - Confirm.

In the same way, create the following transitions:

- ◆ "confirm" between the "defined" and "confirm" states
- ◆ "start" between the "confirm" and "inProgress" states
- ◆ "end" between the "inProgress" and done states

View of the "Training" state diagram

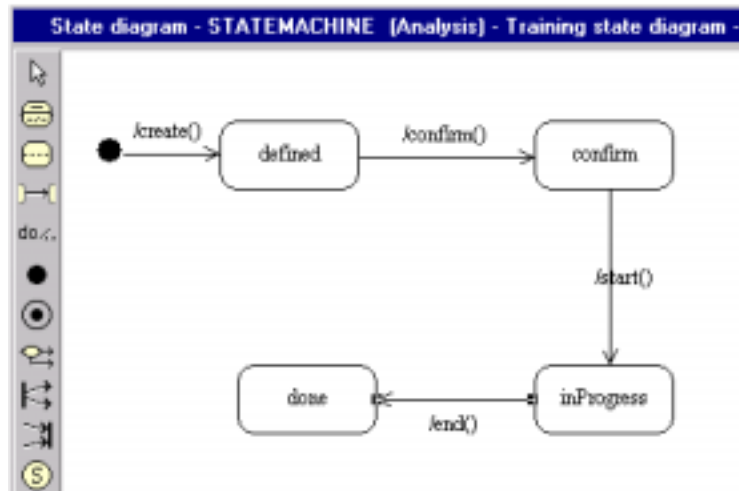



Figure 3-47. View of the "Training" state diagram

Note: To obtain the view represented in the figure above, simply move the states using the mouse and modify the drawing of the link.

Notes and tagged values

Documenting an operation

The  "Add a note" icon

Textual descriptions or notes can be associated to any model element. These notes are assigned a type, designed to describe their purpose (Java, description).

Editing from the explorer

Textual descriptions appear in the "Items" tab of the properties editor, with their type displayed. They can be directly created or edited (as shown in Figure 3-48).

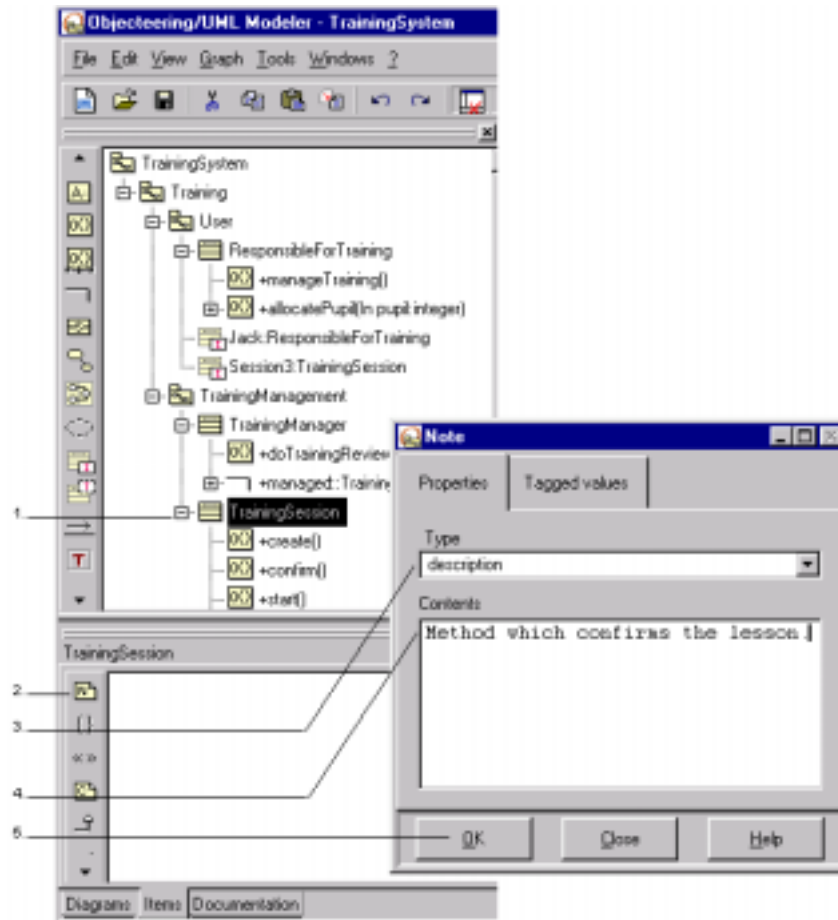



Figure 3-48. Creating a note for the "TrainingSession" class

Chapter 3: First Steps

Steps:

- 1 - Select the "*TrainingSession*" class in the explorer.
- 2 - Click on the  "Add a note" icon in the "Items" tab of the properties editor.
- 3 - Select description.
- 4 - Enter your text in the dialog box.
- 5 - Confirm.

In the same way, create notes for an association, a class and a parameter.

Editing from dialog boxes

The "Notes" tab (shown in Figure 3-49) is used to edit textual descriptions in element dialog boxes (see the *Objecteering/Model Dialog Boxes* user guide).

Note: These are the same as those visible in the "Items" tab of the properties editor.

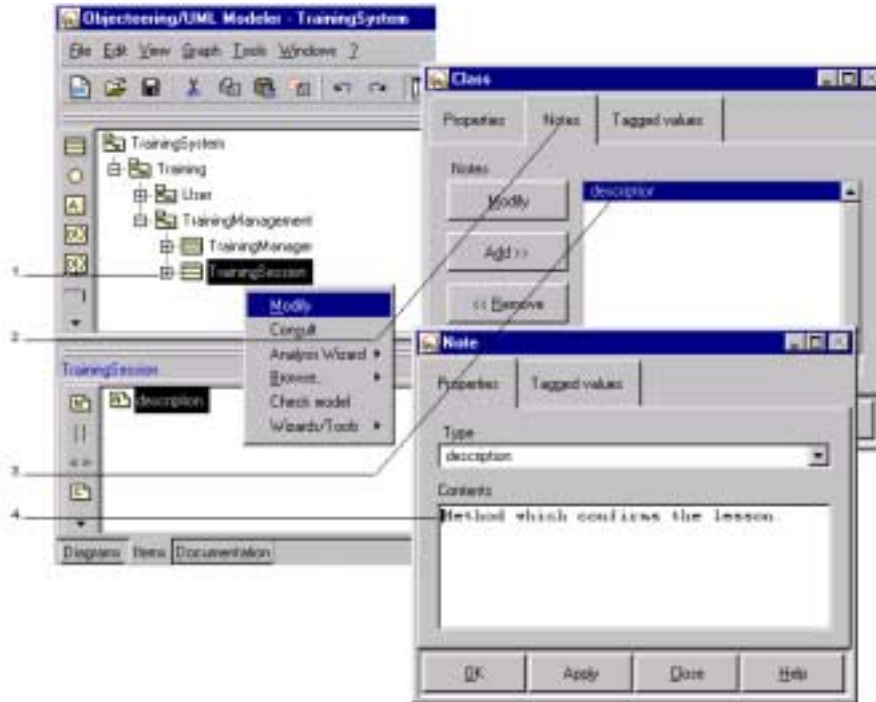



Figure 3-49. Adding a note to the "TrainingSession" class

Chapter 3: First Steps

Steps:

- 1 - Select the "*TrainingSession*" class by clicking on the right mouse button, and select the "*Modify*" option from the context menu which appears, or double-click on the "*TrainingSession*" class itself.
- 2 - Select the "*Notes*" tab.
- 3 - Double-click on description.
- 4 - Modify the text in the dialog box and confirm.

Annotating a class

The  "Associate a tagged value" icon

Tagged values allow the user to annotate model elements, in order to give them a special interpretation (persistent, identifier, etc.). Here, the "{nocode}" tagged value expresses that code will not be generated for the "ResponsibleForTraining" class. The tagged values available vary according to the modules installed. As with notes, tagged values can be edited (Figure 3-50) in the "Items" tab of the properties editor of in the dialog boxes (through the "Tagged values" tab).

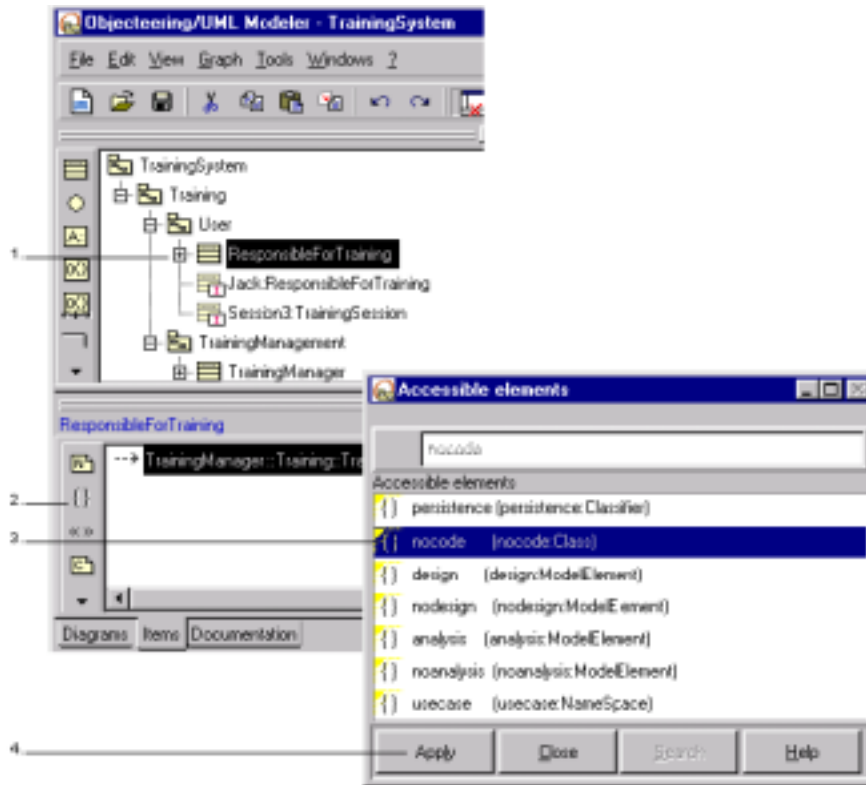



Figure 3-50. Annotation of the "ResponsibleForTraining" class

Chapter 3: First Steps

Steps:

- 1 - Select the "*ResponsibleForTraining*" class in the explorer.
- 2 - Click on the  "Associate a tagged value" icon in the "Items" tab of the properties editor.
- 3 - Select the elements to be referenced from the list of accessible elements.
- 4 - Confirm by clicking on the "Apply" button. The name of the tagged value appears in the field of elements entered in the accessible elements window.

Note: To modify a tagged value, you must delete it and create a new one for the class in question. This is done by selecting the class, using the "Modify" command from the context menu and selecting the "Tagged values" tab, or by directly deleting the tagged value in the "Items" tab of the properties editor.

Result

This is the result as seen in the explorer and the properties editor (as shown in Figure 3-51).

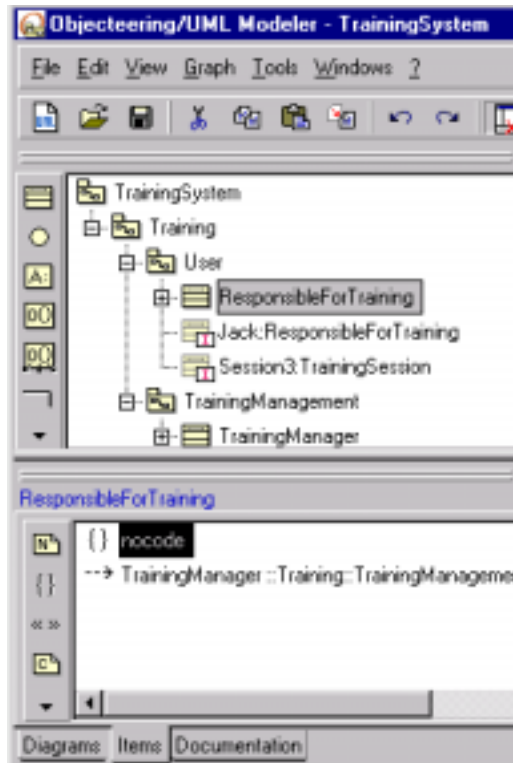


Figure 3-51. Result of the annotation of the "ResponsibleForTraining" class

Chapter 3: First Steps

You can choose whether model elements are displayed with or without their tagged values in graphic editors. For this, simply carry out the following steps:

- 1 - Click on the "Training" package in the explorer, and select the "Training class diagram" diagram from the "Diagrams" tab of the properties editor.
- 2 - Click on the right mouse button and choose the "Modify" option from the context menu which appears.
- 3 - Check the "Tagged value visibility" tickbox (as shown in Figure 3-52).

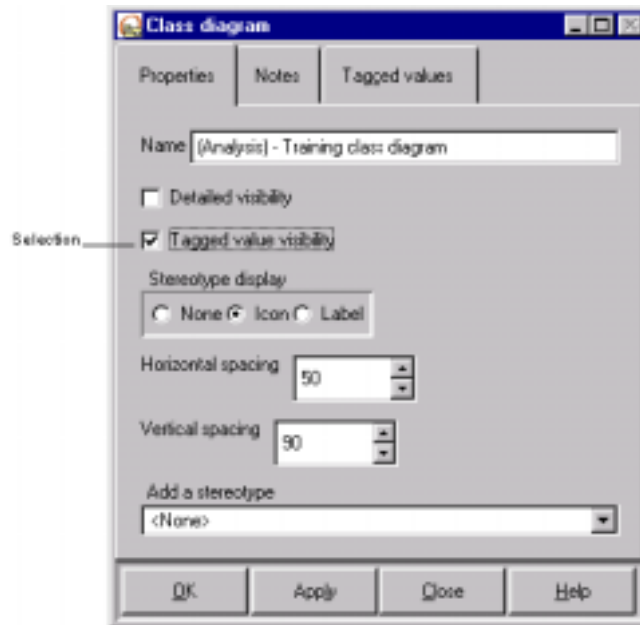



Figure 3-52. The "Training class diagram" diagram dialog box

The result of this operation is that elements created in the explorer which have tagged values are displayed with their tagged values in the graphic editors.

Result

Once this option has been selected, elements annotated with tagged values are displayed with their tagged values visible in graphic editors (Figure 3-53). We are going to check this by editing the diagram created on the "Training" package class.

In the explorer, select the "Training" package. In the "Diagrams" tab of the properties editor, double-click on the  "Training class diagram" class diagram.

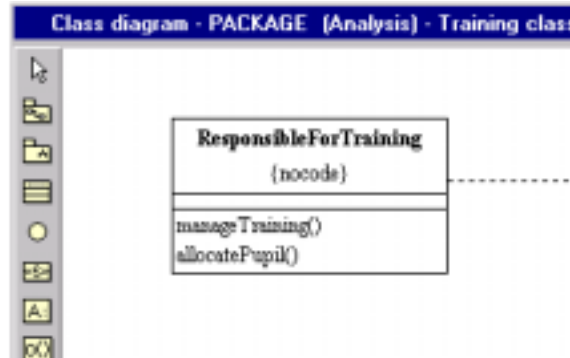


Figure 3-53. The `{nocode}` tagged value is now visible in the "ResponsibleForTraining" class

Note: Please note that tagged values which have parameters are displayed in the "Item" tab of the properties editor with their parameters also displayed.





Implementing modules

Presentation

Up until now, we have only used *Objectteering/UML Modeler*. Now, *Objectteering/Documentation* (which is selected by default) and *Objectteering/Java* (which you selected at the beginning of these first steps) will be implemented.

The purpose of modules

A module provides (as shown in Figure 3-54):

- ◆   generation work products, which represent the generated element (documentation, Java code, C++ code, Makefiles, etc.), and which appear in the "Items" tab of the properties editor.
- ◆  notes, specific to the module, which contain documentation or code, for example, and which appear in the "Items" tab of the properties editor.
- ◆  annotations or tagged values, used to give specific module indications to the model, and which appear in the "Items" tab of the properties editor.
- ◆ commands, which are "popup menus" or "context menus" specific to the module, and which appear on elements in the explorer and the properties editor, notably work products.

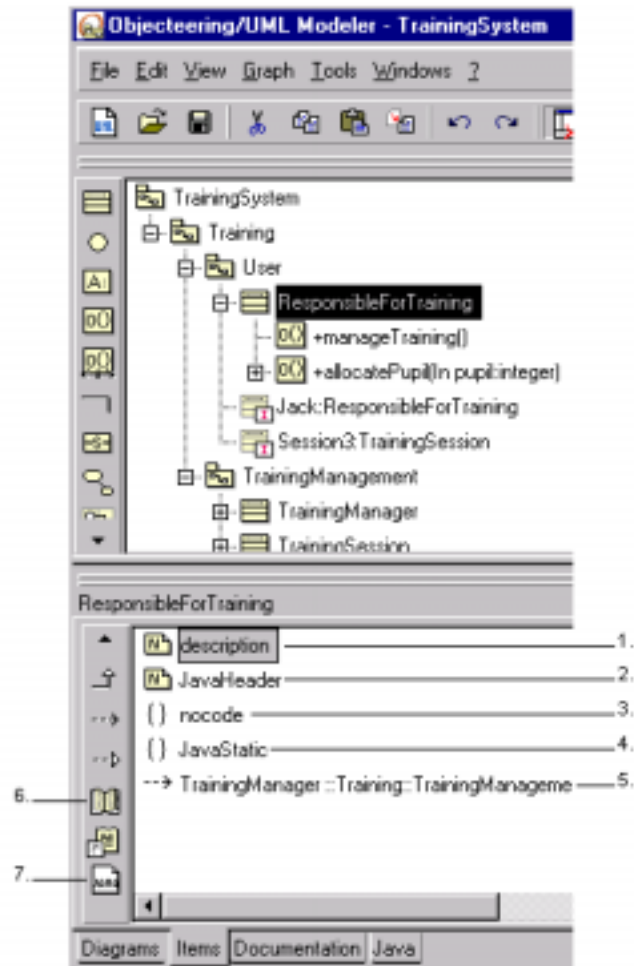



Figure 3-54. Elements available when modules are implemented

Key:


- 1 - Documentation note
 - 2 - Java note
 - 3 - Tagged value specific to documentation
 - 4 - Tagged value specific to Java
 - 5 - Dependency
 - 6 - Document work product icon
 - 7 - Java generation work product icon
-

Setting module parameters

Editing configuration

The  "Modify module parameter configuration" icon is used to launch the module parameterization window (as shown in Figure 3-55). For example:

- ◆ The "directories" sub-section in the "UML Modeler" section is used to position the root directory for generation (GenRoot).
- ◆ The "Documentation" section is used to configure documentation parameters.
- ◆ The "Java" section is used, amongst other things, to specify the root directory for code generation and the usual text editor.

For parameters which require directories to be specified, the  "Browse" icon is used to display a file browser, through which you can select the correct directory.

In the module configuration window, it is possible to enter \$(...) to indicate a directory path, for example, the Objectteering/UML installation path or a generation directory path. This is then transformed in the module configuration window into the path indicated. Parameters entered in the form of \$\$(...) are not expanded, but are transformed into \$(...) form.

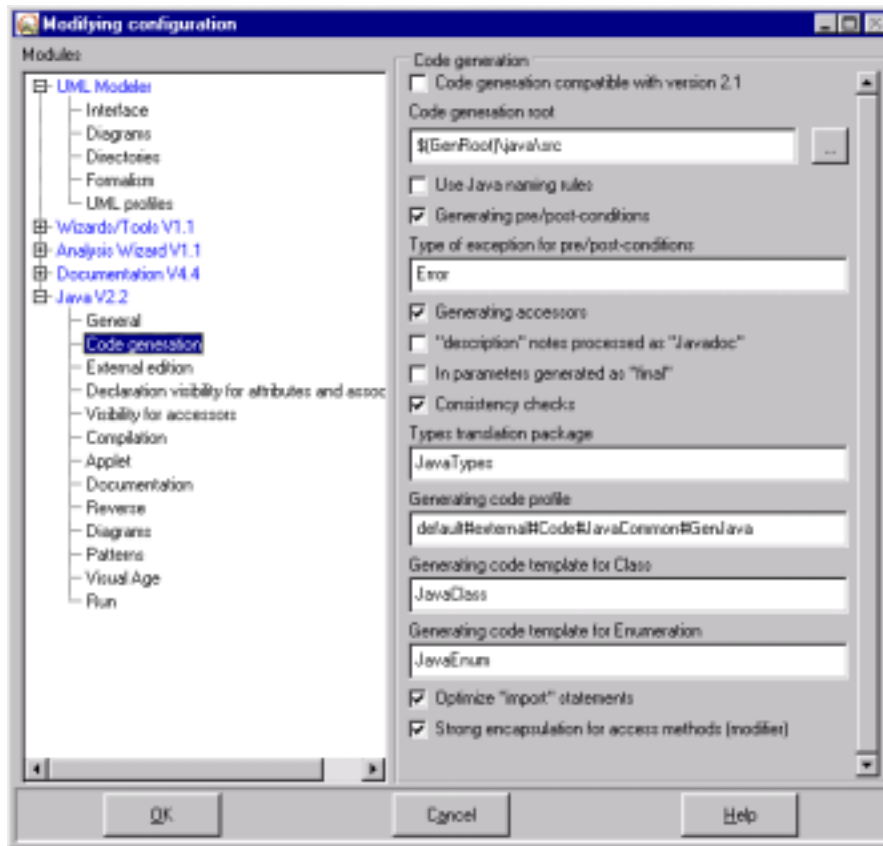


Figure 3-55. The "Code generation" set in the "Java" section of the "Modifying configuration" window


Choosing generation directories

You must define the directories of the different generations. Certain parameters are specially designed for this.

| The ... parameter | in the ... section | Default value ... | is used to ... |
|--|--------------------|---|---|
| Root directory for generation [GenRoot] | UML Modeler | on PC: c:\Projects on UNIX: /true/Projects | define a general directory, called \$(GenRoot) in which the other directories will be placed. |
| Generation root directory | Documentation | \$(GenRoot)\doc | define the root directory of the documentation generation. |
| Code generation root | Java | \$(GenRoot)\java\src | define the generation directory for Java source files. |


Generating documentation


Presentation

The  "Document work product" icon

We are now going to carry out the first documentation generation. This is explained in more detail in the "First Steps" chapter of the *Objecteering/Documentation* user guide.

To create documentation, you must:

 enter textual descriptions

 create document work products

Entering document text



"*Summary*" notes present a one-line element résumé, and "*description*" notes provide a detailed description of these.

We have already created a "*description*" note on the "*TrainingSession*" element (the creation procedure is re-illustrated in Figure 3-56). Now enter some "*summary*" and class and package "*description*" notes.

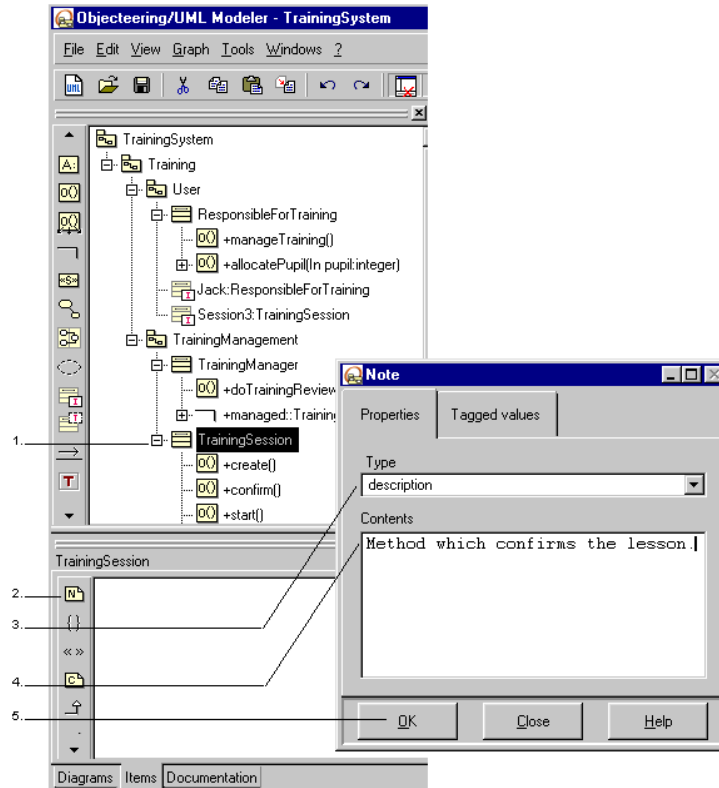




Figure 3-56. Example of entering document text

Steps:

- 1 - Select the "*TrainingSession*" class.
 - 2 - Click on the  "Add a note" icon.
 - 3 - Select description from the list.
 - 4 - Enter the text.
 - 5 - Confirm.
-

Creating a document work product

Creating a document work product

The  "Create a document work product" icon

The document work product (whose dialog box is shown in Figure 3-57) represents the documentation generated. Its destruction brings about that of the generated documentation file. By handling the document work product, you can edit documentation or change generation parameters.

A document work product is associated to a modeling element (class, etc).

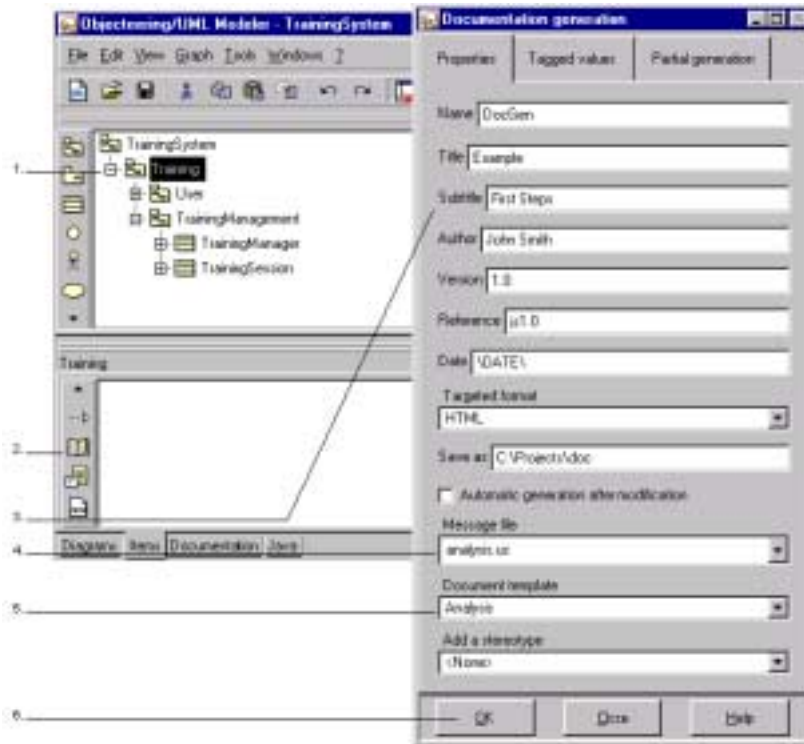



Figure 3-57. Creating a document work product

Steps:

- 1 - Select the "*Training*" package in the explorer.
- 2 - Click on the  "*Create a document*" icon in the "*Items*" tab of the properties editor.
- 3 - Define the entry fields as shown in this example.
- 4 - Select "*analysis.us*".
- 5 - Select "*Analysis*".
- 6 - Confirm.

Generating documentation

The document work product provides a documentation generation service (as shown in Figure 3-58).

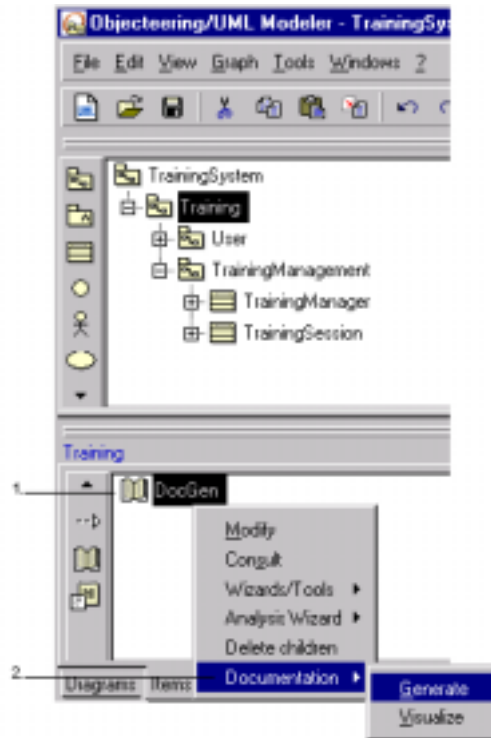


Figure 3-58. Generating the created document

Steps:

- 1 - Select the document by clicking on the right mouse button.
- 2 - Select the "Documentation/Generate" options from the context menu which appears.

Observe the progress of the generation in the console (Figure 3-59).

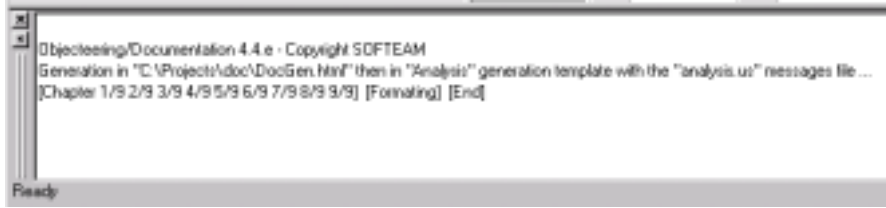


Figure 3-59. Visible progress of the generation in the console

Result

By clicking on the right mouse button, activate the "Documentation/Visualize" options in the context menu on the document work product (Figure 3-60).

Note: In the documentation directory, a file named "DocumentWorkProductName.htm" is generated, as well as all the HTML diagrams and files necessary. These can be directly visualized using internet navigators.



Figure 3-60. Documentation generated

Note: The Word or PostScript formats can also be used.


Generating code

Java generator

All generators (Java, C++, etc.) are based on the same principles.

We are now going to generate Java code (assuming that you have a license), and visualize the result. The Java code generation first steps provide further details on this subject. Here, we are simply going to produce binaries.


To generate code, you must:

- ◆ enter the Java code in the model
- ◆  create a Java generation work product associated with the elements which produce the code (classes, packages).

Note: The creation of a generation work product on a package reflects this action on its components (packages, classes).

Objectteering/UML completely generates code sources, makefiles and the binary. The permanent consistency check mechanism is then used to make the code evolve, either in generated sources or in Objectteering/UML, by guaranteeing that the code remains permanently consistent with the model.

Entering code

The  "Add a note" icon

We are going to enter the contents of an operation. Proceed as shown below (Figure 3-61).

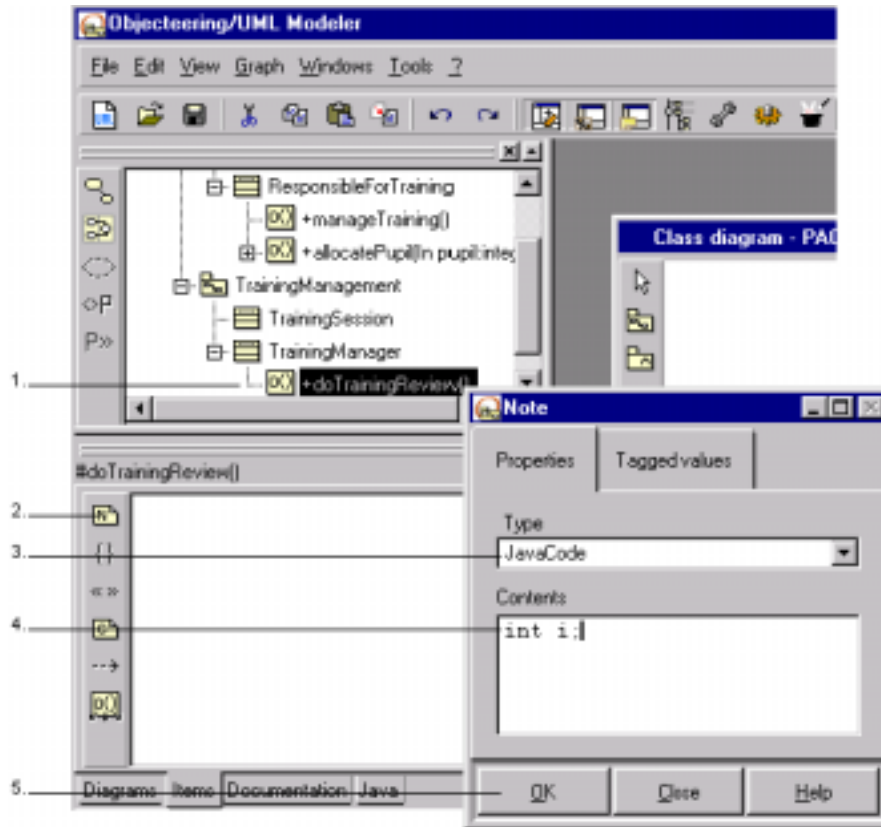




Figure 3-61. Entering code

Steps:

- 1 - Select the "*doTrainingReview()*" operation in the explorer.
- 2 - Click on the  "Add a note" icon in the "Items" tab of the properties editor.
- 3 - Select the "JavaCode" note type.
- 4 - Enter "*int i;*".
- 5 - Confirm.

Note: The *Objectteering/C++* and *Objectteering/Java* modules can also be used to enter code directly in the final source.

Creating a Java generation work product

The  "Create a Java generation work product" icon

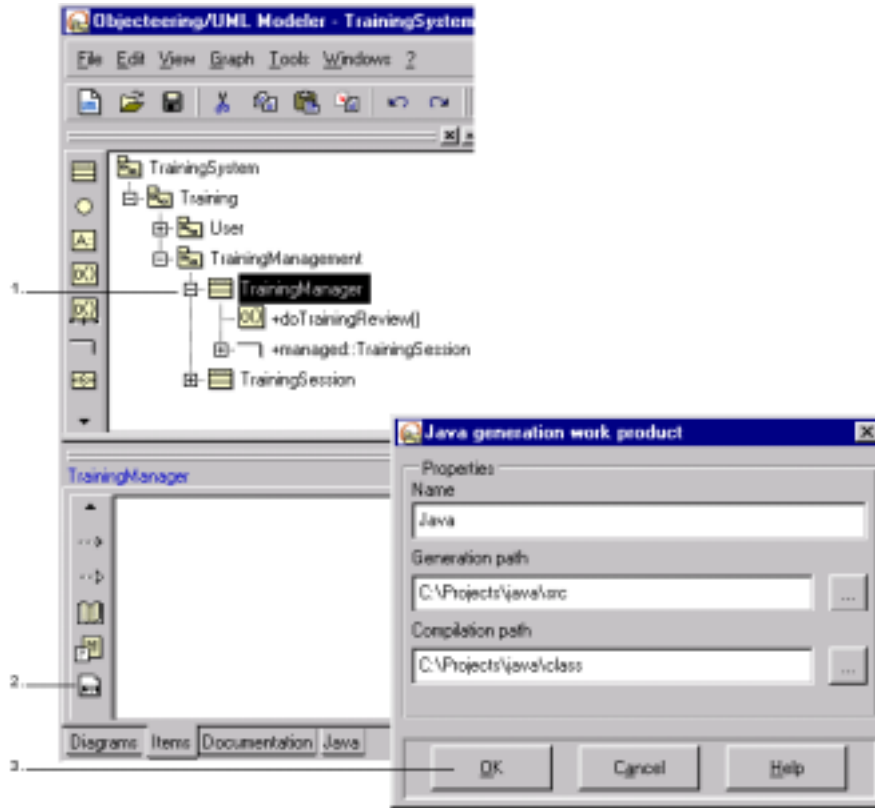



Figure 3-62. Creating a Java generation work product

Steps:

- 1 - Select the "*TrainingManager*" class in the explorer.
- 2 - Click on the  "Create a Java generation work product" icon in the "Items" tab of the properties editor.
- 3 - Confirm.

Note: If the class or package for which you have created a Java generation work product contains other lower-level packages and classes, you can propagate the work product by running the "*Propagate*" command, available in the context menu on the generation work product itself.

Generating code

To generate your generation work product, select the Java work product by right-clicking on the mouse, and then select the "*Java/Generate*" options from the context menu. (Double-clicking on the work product is a generation short cut). Observe the code messages in the console.

Visualizing code

We can now visualize the code directly in an adapted display window (as shown in Figure 3-63). Double-clicking on the textual zones in this window brings you back to the dialog boxes corresponding to the equivalent parts of the model.

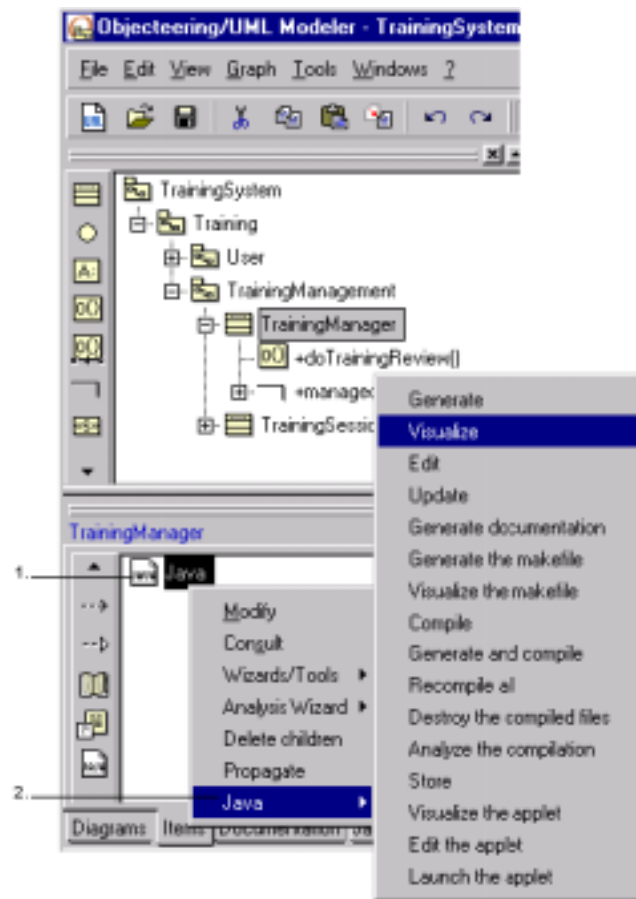


Figure 3-63. Visualizing code

Chapter 3: First Steps

Steps:

- 1 - Select the generation work product in the "Items" tab of the properties editor by clicking on the right mouse button.
- 2 - Choose the "Java/Visualize" options from the context menu which appears.
- 3 - Carry out the steps shown in figure 3-64.

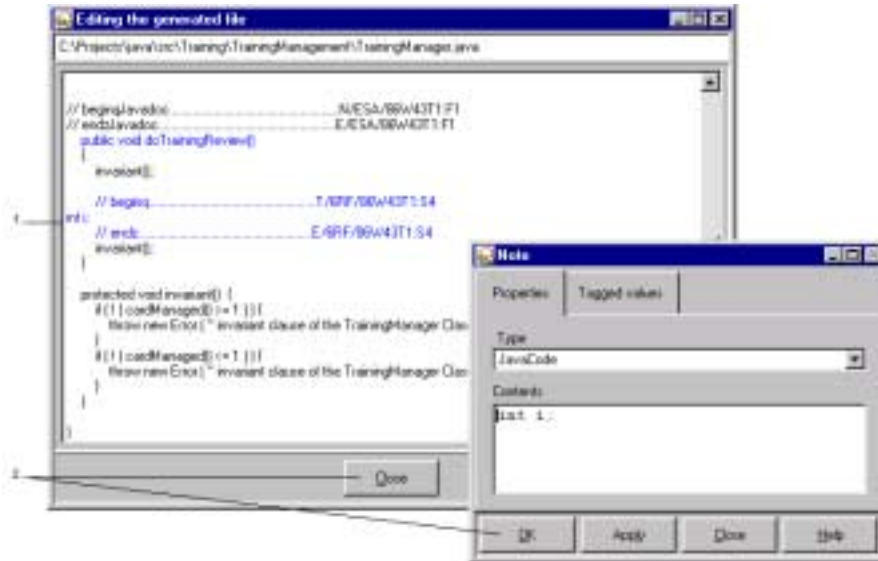


Figure 3-64. Direct return on the model

Steps:

- 1 - Double-click on the "int i;" code line.
- 2 - Close.

Editing code

The source can be edited directly (as shown in Figure 3-65). For example, we will add a second code line to our method, and check that the Objectteering/UML repository has been updated, by consulting the textual zone.

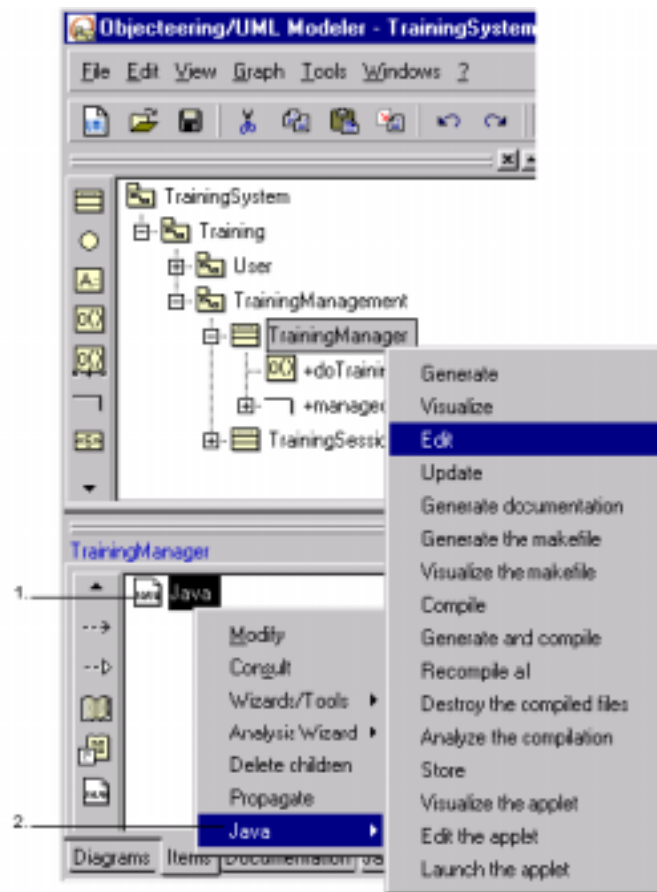
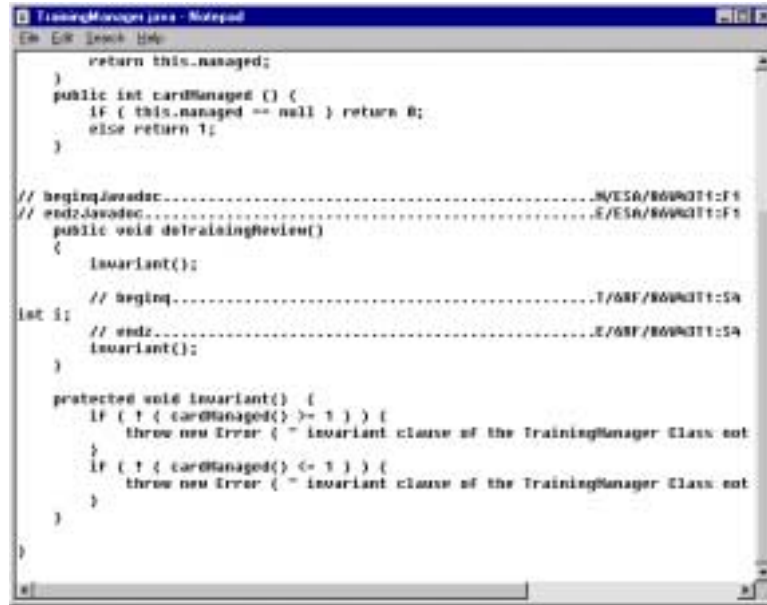


Figure 3-65. Adding a line of code

Chapter 3: First Steps

Steps:

- 1 - Select the generation work product by clicking on the right mouse button.
- 2 - Choose the "Java/Edit" options from the context menu which appears.
- 3 - Proceed with the operations shown in Figure 3-66.



```
TrainingManager.java - Notepad
File Edit Insert Help

return this.managed;
}
public int cardManaged () {
    IF ( this.managed == null ) return 0;
    else return 1;
}

// beginqJavadoc.....N/ESA/R040311:F1
// endJavadoc.....E/ESA/R040311:F1
public void doTrainingReview()
{
    invariant();

// beginq.....T/ASF/R040311:5A
int i;
// endz.....E/ASF/R040311:5A
    invariant();
}

protected void invariant() {
    IF ( ! ( cardManaged() >= 1 ) ) {
        throw new Error ( " invariant clause of the TrainingManager class not
    }
    IF ( ! ( cardManaged() <= 1 ) ) {
        throw new Error ( " invariant clause of the TrainingManager class not
    }
}
}
```

Figure 3-66. Inserting a line of code in a file

Steps:

- 1 - Insert the "int i;" line of code. Save and exit the editor.
- 2 - Edit the "Java" textual zone of the operation ("Java/Edit"), and observe that the repository has been updated.

Conclusion

What you haven't yet seen:

- ◆ *UML Profile Builder*: tool used to parameterize the workshop, only available with a specific license in the Enterprise Edition or the Professional Edition. Capable of defining model processing (requests, model transformations, etc.), new development work products, stereotypes and notes.
- ◆ *Objectteering/UML teamwork services*: group work management ("*Enterprise Edition*").
- ◆ *Objectteering/Administrating Objectteering Sites*
- ◆ Generator parameterization: adaptation of each generator, using the *UML Profile Builder* tool ("*Enterprise Edition*")

Please refer to the following user guides in order to carry out the First Steps modeling projects specific to the module in question:

- ◆ *Objectteering/UML Modeler*
- ◆ *Objectteering/Documentation*
- ◆ *Objectteering/Document Template Editor*
- ◆ *Objectteering/C++*
- ◆ *Objectteering/UML Profile Builder*
- ◆ *Objectteering/Metamodel User Guide*
- ◆ *Objectteering/Java*

You are now ready to use the Objectteering/UML tool. Please refer to the "*Objectteering/UML user guides*" page in chapter 1 of this user guide.

Index

.prof file 2-22
 .prof files 2-37, 2-48, 3-11
 {nocode} tagged value 3-85
 Adding a parameter to an operation 3-20
 Activity diagram 3-72
 Activity graph 3-72
 Administration tool 2-61
 Annotating a class 3-85
 Assisted data entry 3-3, 3-32, 3-70
 Assisted data entry mechanism 3-21
 Association 3-41
 Attribute 3-13
 Before starting server installation 2-26
 C++ code 1-5
 Changing UML modeling project 3-8
 Choosing generation directories 3-96
 Class 3-13, 3-18, 3-23, 3-28, 3-49, 3-64, 3-82, 3-100, 3-105
 Class diagram 3-23, 3-29
 Client installation
 Modules 2-48
 Client installation in Windows
 Procedure 2-40
 Client/server installation mode 2-4
 Code generation 1-3
 Collaboration 3-64
 Consistency checking rules 1-4
 Consistency checks 3-3, 3-38, 3-45, 3-105
 Consistency mechanisms 3-21, 3-29
 Console 3-7, 3-15, 3-103, 3-110
 Constraints 3-43
 Context menu 3-42, 3-44
 Context menu commands 1-4
 Continuous entry creation mode 3-15, 3-18, 3-50
 Copy/Paste mechanism 3-3, 3-30
 Creating a class 3-16
 Creating a class diagram 3-23
 Creating a class in a diagram 3-40
 Creating a dependency 3-32
 Creating a document work product 3-100
 Creating a Java generation work product 3-108
 Creating a note in a class diagram 3-43
 Creating a package 3-14, 3-49
 Creating a reflexive message from an instance 3-68
 Creating a sequence diagram from the explorer 3-64
 Creating a state 3-75
 Creating a state diagram from the properties editor 3-73
 Creating a state machine in the explorer 3-71
 Creating a UML modeling project 3-4, 3-9
 Creating an association 3-41
 Creating an association between two classes 3-41
 Creating an instance in the sequence diagram graphic editor 3-66
 Creating an operation 3-18
 Creating transitions 3-76
 Dependency 3-32, 3-53, 3-93
 Detailed visibility 3-55, 3-67
 Diagram notes 3-44
 Diagrams 3-3, 3-7, 3-23

- Dockable windows 1-7
- Document templates 1-4
- Document work product 3-97, 3-100, 3-102
- Documentation 1-5
- Documentation formats 3-104
- Documenting an operation 3-80
- Drag and drop mechanism 3-31, 3-34, 3-51, 3-57, 3-61
- Drawing links 3-37
- Editing a binary association 3-47
- Editing a class diagram 3-25
- Editing code 3-113
- Editing from dialog boxes 3-83
- Editing from the explorer 3-81
- Editing the configuration of module parameters 3-94
- Entering association values 3-46
- Entering code 3-106
- Entering document text 3-98
- Entering operations 3-56
- Enterprise Edition 2-4, 2-10, 3-115
- Error messages 3-52
- Explorer 2-37, 3-3, 3-7, 3-13, 3-19, 3-29, 3-33, 3-40, 3-57, 3-58, 3-71, 3-83
- Explorer hierarchy 3-13
- File browser 3-94
- First Steps
 - Adding a parameter to an operation 3-20
 - Changing UML modeling project 3-8
 - Creating a class 3-16
 - Creating a class diagram 3-23
 - Creating a new UML modeling project 3-4
 - Creating an operation 3-18
 - Operations demonstrated 3-3
 - Selecting modules 3-10
 - Working in the explorer 3-13
- FlexIm tool 2-4
- Generating code 3-110
- Generating documentation 3-97, 3-102
- Generation directory parameters 3-96
- Generation parameters 3-100
- Generation work product 3-105, 3-110
- Graphic editors 3-40, 3-88
- Handling model elements in graphic editors 3-26
- Help mechanisms 3-29
- Implementing modules 3-90
- Installation
 - Heavyweight client 2-3
 - Lightweight client 2-3
- Installation modes 2-3
 - Client configuration 2-3
 - Server configuration 2-3
 - Single station configuration 2-3
- Installing documentation 2-15
- Installing in UNIX
 - Activating the FlexIm server 2-60
 - Activating the Objectteering/UML site server 2-60
 - CD-ROM 2-52
 - Client-server configuration 2-50, 2-59
 - Different installation modes 2-50
 - Directories which are not shared 2-61
 - Environment variables 2-58
 - Installation over a previous version 2-50

- Installing from the CD-ROM 2-53
- Installing the client station 2-61
- Locating information 2-51
- Locating the installed software 2-60
- Online help 2-58
- Other necessary operations 2-58
- Prerequisites 2-50
- Procedure 2-51
- Re-installing 2-57
- TCP/IP configuration 2-59
- Updating the client station configuration 2-61
- Installing in Windows
 - Choosing the destination location 2-13
 - Choosing the setup type 2-14
 - Delivering and migrating modules 2-19
 - End of installation 2-23
 - Entering user information 2-12
 - Information on module location 2-22
 - Migrating databases during installation 2-17
 - Prerequisites 2-5
 - Procedure 2-7
 - Selecting components 2-16
 - Selecting the installation mode 2-9
 - Selecting the program folder 2-20
 - Starting to copy files 2-21
 - The software license agreement 2-8
- Instance 3-66
- Instance dialog box 3-67
- J language 1-4
- Java code 1-5, 3-3
- Java generator 3-105
- Licence file 2-50
- License contents on the license server 2-62
- License contents on the Objecteering/UML application server 2-63
- License file 2-7, 2-10, 2-56
- License management server procedure 2-38
- Licenses
 - Entering license information 2-64
- Link 3-28
- LM_LICENSE_FILE 2-62
- LM_LICENSE_FILE environment variable 2-34, 2-62
- Imgrd 2-60
- Imtools 2-4, 2-65
- Locked files 2-71
- Log file 2-23
- Macros 3-12
- Masking an element 3-28
- Masking an element in a diagram
 - Context menu on a diagram element 3-28
 - Keyboard shortcut 3-28
 - Options 3-28
- Menu bar 3-7
- Menu shortcut bar 3-7
- Migrating a client installation 2-48
- Migrating databases 2-17
- Migration 2-5
- Model consistency 1-3
- Model driven development 1-5
- Model elements 3-13
- Model exchange services 1-3
- Model transformation rules 1-4
- Modifying a tagged value 3-86

- Modifying your Objectteering/UML installation 2-67
 - Locked files 2-71
 - Read only files 2-71
 - The "Modify" mode 2-68
 - The "Remove" mode 2-70
 - The "Repair" mode 2-70
- Module commands 3-91
- Module licenses 3-11
- Module location 2-22
- Module selection and the properties editor 3-12
- Modules 3-11
- Note 3-43
 - Description 3-98
 - Summary 3-98
- Notes 1-4, 3-43, 3-80, 3-83, 3-91, 3-97, 3-115
- Object diagram 3-29
- Objectteering/Administrating
 - Objectteering Sites 3-115
- Objectteering/C++ 1-3, 3-10, 3-12, 3-107, 3-115
- Objectteering/Design Patterns for C++/Java 1-3, 3-10
- Objectteering/Design Patterns for Java 3-5
- Objectteering/Document Template Editor 3-115
- Objectteering/Documentation 3-10, 3-12, 3-44, 3-94, 3-97
- Objectteering/Java 1-3, 3-5, 3-10, 3-12, 3-94, 3-105, 3-107, 3-115
- Objectteering/Metamodel User Guide 3-115
- Objectteering/Model Dialog Boxes 3-48, 3-83
- Objectteering/UML delivery 2-3
- Objectteering/UML editions 1-3
- Objectteering/UML Enterprise Edition 1-3, 3-115
- Objectteering/UML environment variables
 - O_HELP_BROWSER 2-58
 - OBJING_BINPATH 2-58
 - OBJING_LANG 2-58
 - OBJING_PATH 2-58
- Objectteering/UML First Steps 2-3
- Objectteering/UML installation CD-ROM 2-3
- Objectteering/UML installation procedure 3-11
- Objectteering/UML license file 2-4
- Objectteering/UML Modeler 2-3, 2-5, 3-5, 3-37, 3-115
- Objectteering/UML modules 3-10
- Objectteering/UML Personal Edition 1-3
- Objectteering/UML Professional Edition 1-3, 3-115
- Objectteering/UML Profile Builder 1-3, 1-4, 3-10, 3-115
- Objectteering/UML repository 3-113
- Objectteering/UML site server 2-24
- Objectteering/UML technical support team 2-4, 2-56
- Objectteering/UML tools 2-49
- Objectteering/UML version 2-4
- Objectteering/UML version history 1-6
- Objectteering/Visual Basic 3-12
- objingsrv 2-60
- On-line help 2-3, 3-3
- Opening a UML modeling project 3-9
- Operation 3-13, 3-18, 3-28, 3-56, 3-70
- Organizing classes 3-51

- Organizing the model 3-49
- Package 3-13, 3-23, 3-49, 3-64, 3-105
- Parameter 3-20, 3-82
- Parameter passing modes 3-22
- Parameterizing Objectteering/UML 1-3
- Personal Edition 2-4, 2-7, 2-10
- Producing documentation 3-3
- Professional Edition 2-4, 2-10, 3-115
- Properties editor 1-7, 3-7, 3-12, 3-73, 3-88, 3-91
 - Diagrams tab 3-24, 3-61, 3-65, 3-74
 - Items tab 3-33, 3-43, 3-62, 3-81
- Purpose of modules 3-91
- Read only files 2-71
- Readme file 2-3, 2-23
- Receiving a UML modeling project 3-9
- Redrawing links 3-42
- Release file 2-3, 2-23
- Repeated entry mode 3-56, 3-75
- Right angles in links 3-42
- Saving the model context 3-9
- Saving the user context 3-9
- Saving work contexts 3-9
- Selecting modules 2-22, 3-11
- Sequence diagram 3-29
- Sequence message 3-68
- Sequence object 3-66
- Server installation
 - Modules 2-37
- Server Installation in Windows
 - Activating the FlexIm service 2-38
 - Activating the Objectteering/UML site server 2-38
- Client-server functioning 2-24
- Completing the installation 2-37
- Preparing to install the client station 2-39
- Selecting the Server installation mode 2-27
- TCP/IP configuration 2-36
- Showing a link in a diagram
 - Context menu 3-60
 - drag and drop function 3-61
 - Keyboard shortcut 3-36, 3-60
- Showing a link through the context menu 3-36
- Showing an element in a diagram 3-25, 3-58
 - Context menu on a diagram element 3-27
 - Keyboard shortcut 3-27
 - Options 3-27
- Showing elements in a diagram 3-57
- SQL instructions 1-5
- Stand alone installation 2-43
- Stand installation in Windows 2-6
- State 3-75
- State diagram 3-71
- State machine 3-71
- Status bar 3-7
- Stereotypes 3-115
- Tagged value 3-85
- Tagged values 1-4, 3-43, 3-91
- TCP/IP layer 2-25, 2-40
- Transition 3-76
- Transition dialog box 3-78
- Type of UML modeling project 3-5
- UML model root 3-5, 3-13
- UML model root name tickbox 3-5
- UML model type 3-5

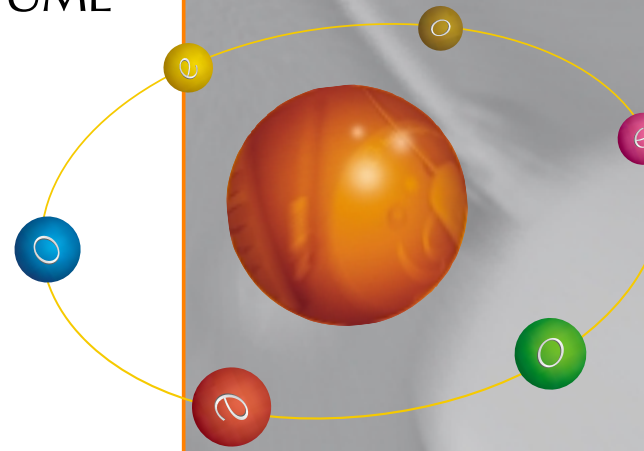
UML Modeling 1-3
UML modeling project 3-3
UML modeling projects 3-4, 3-13
UML Profiles 1-4
UNC names 2-39
UNC paths 2-40
Undo/Redo mechanism 3-3, 3-15, 3-29
Unified Naming Convention 2-39

Universal identification mechanism
1-3, 2-3
Upgrading a UML modeling project
3-9
Upgrading projects 2-5
Visibility 3-19
Visualizing code 3-111
Work products 3-91, 3-105, 3-115

Objecteering/UML

Objecteering/Administrating Objecteering/UML
Sites User Guide

Version 5.2.2



www.objecteering.com

Taking object development one step further

Objecteering

Software

Contents

| | |
|---|------|
| Chapter 1: Introduction | |
| Overview | 1-3 |
| General site structure | 1-4 |
| Glossary | 1-7 |
| Chapter 2: Overview of the administration tool | |
| Overview of the administration tool | 2-3 |
| The System menu | 2-5 |
| Overview of the Administration menu | 2-6 |
| Overview of the Configuration menu | 2-9 |
| Overview of the Repair menu | 2-11 |
| Overview of the Verification menu | 2-14 |
| Chapter 3: Detailed view of the administration tool | |
| Detailed view of the Administration menu | 3-3 |
| Detailed view of the Configuration menu | 3-21 |
| Detailed view of the Repair menu | 3-30 |
| Detailed view of the Verification menu | 3-42 |
| Repairing databases | 3-49 |
| The _predefinedTypes project | 3-50 |
| Index | |

Chapter 1: Introduction

Overview

Introduction to database administration

Welcome to the *Objectteering/Administrating Objectteering Sites* user guide!

The Objectteering/UML administration tool is used to manage the Objectteering/UML repository at site level. This repository is made up of databases, which contain UML modeling projects, UML profiling projects and document template projects.

Databases contain modeling data and additional programming elements, and are used to configure the modules used. This modeling data is essential to user projects and is the result of very sophisticated semantic data storage and administration techniques.

The administration tool

Modeling data is managed by the administration tool, which is used to:

- ◆ manage physical database parameters (localization, multi-platform aspects)
- ◆ manage database configuration (modules used, module parameterization)
- ◆ manage UML modeling projects, UML profiling projects and document template projects contained within databases
- ◆ repair databases, through operations which correct and maintain data in a consistent and usable state after a problem has occurred

Furthermore, for users of the *Objectteering/UML Profile Builder* module, the J language provides administration services which can be used in "batch" mode. For further information, please refer to chapter 7, "Executing J online - Overview", of the *Objectteering/The J Language* user guide.

General site structure

Site overview

A site is constructed as follows:

- ◆ a set of files defining server characteristics, a site and user databases
- ◆ a special database, called the site database, which centralizes site information
- ◆ user databases, physically separated by specific data files
- ◆ UML modeling projects, which are located in user databases and in which models are built
- ◆ UML profiling projects, used to parameterize the Objectteering/UML CASE tool. For further information, please refer to the *Objectteering/UML Profile Builder*, or *Objectteering/Document Template Editor* (where document template projects are concerned) user guides.

Each database has a unique name, and is placed in a physical location (directory) which is defined by the administrator. The administrator can thus centralize databases in a single directory or locate them in several new directories. The site database is "invisible" to the user. Its physical localization is decided upon during installation.

Site identification

Each site is identified in a unique way. Within this site, UML modeling projects and model elements are also individually and uniquely identified. This universal identification mechanism ensures the traceability of the model elements on all the sites and databases in which they are distributed.

Exchanging information between sites (Enterprise Edition only)

Objectteering/UML databases are autonomous, as are the UML modeling projects contained within these databases. Objectteering/UML provides transfer services used to:

- ◆ exchange data between UML modeling projects contained in the same database or in different databases (Enterprise Edition only)
- ◆ exchange data between UML profiling projects in the same database or in different databases (Enterprise Edition only)
- ◆ install modules from UML profiling projects into user databases (Enterprise Edition and Open Edition)
- ◆ externalize database contents in ASCII format (Enterprise Edition only)

Objectteering/UML also provides a service which allows developers to share modeling data. This workspace administration service is described in the *Objectteering/UML Teamwork User Guide* (Enterprise Edition only).

XMI exchange

Objectteering/UML also provides an XMI exchange service, (the OMG model exchange standard), which allows models to be exchanged between CASE tools. This exchange mode does not include the same amount of detail and services as other exchange modes, and should be used only when exchanging models with other CASE tools. *XMI Module Import* is, however, the only exchange mode available with the Objectteering/UML Personal Edition.

Methods of exchanging information between sites

Information between sites can be exchanged in two different ways:

- ◆ By exchanging externalized ASCII files (Enterprise Edition only)
- ◆ By receiving a database belonging to a different site

Transferring modules between sites (Enterprise Edition only)

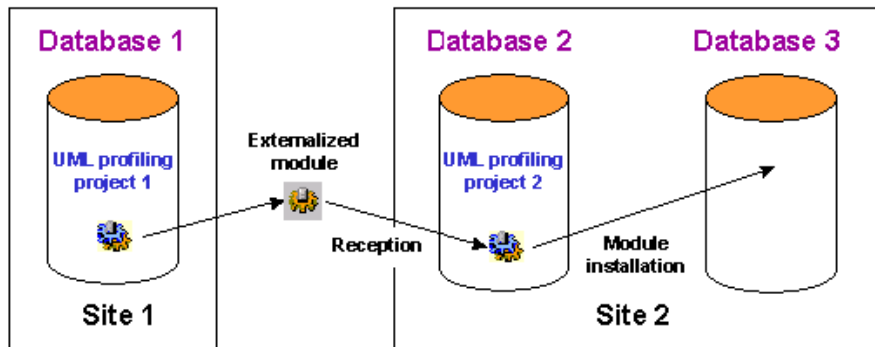


Figure 1-1. Externalizing the module

Objecteering/UML offers an externalization/internalization service which is used to transfer modules from one site to another.

The operation consists of:

- ◆ externalizing the module from the UML profiling project where it has been developed (in the *Objecteering/UML Profile Builder* module)
- ◆ transferring the files to the destination site file system
- ◆ receiving the module (reference, etc.)

The module can then be transferred into a site UML profiling project or installed on a site database (Figure 1-1).

Glossary

Database: binary file containing information related to stored objects. This file is in FP format and takes the "*.ofp" extension.

File identifier: a set of information on a file's characteristics (date of creation, access path).

Module: a "functionally consistent" group of UML profiles and commands, individually packaged (delivery, installation and so on).

Modules.ifo: text file which is used to centralize access to the different site module databases.

Modules.ifo.save: text file in which the "modules.ifo" file is saved.

Object counter: a counter present in a project and used to give a unique number to all objects created in this project.

Physical information: database file access path, as defined by the system.

Project counter: a counter present on the site database, which allocates a unique number to each different project on the site.

Site: physical unit used to group Objectteering/UML databases.

Site identifier: an object identification number. This number must be unique for each different object created in Objectteering/UML, whatever the site and project concerned. This number is composed as follows: <site code><project n°><n° of the object in the project>.

Site.ifo: text file which is used to centralize access to the different site development databases.

Site.ifo.save: text file in which the "site.ifo" file is saved.

UML modeling project: workspace where models are built and used.

UML profiling project: environment where UML profiles and modules are developed.

Zombie: element deleted from the database but which may be restored.

Chapter 2: Overview of the administration tool

Overview of the administration tool

The "administration" tool

The "*administration*" tool centralizes services used to manage Objecteering/UML sites, and allows you to destroy, move, rename or allocate physical parameters to a database. It can also be used to check and repair databases, and to deliver, install and uninstall modules.

The administration tool functions in two different modes:

- ◆ interactive mode, in which you handle databases through the user interface
- ◆ batch mode, which allows you to carry out administration procedures using system commands. This mode, together with the "*J online*" service (please refer to the *Objecteering/The J Language* user guide), offers a high level of flexibility from an administration automation point of view. The batch mode is run through "*DOS*" or the "*baseadm*" shell command. In this chapter, each command and administration option will be presented with its user interface and batch syntax.

The main window

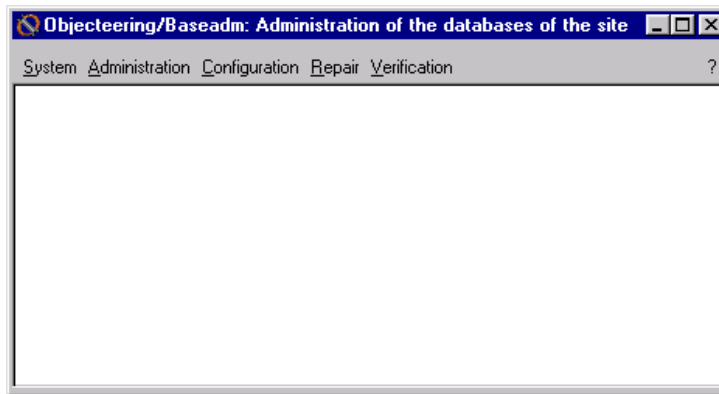


Figure 2-1. The main administration window

| The ... menu | is used to ... |
|----------------|--|
| System | empty the console, record in a file, exit. |
| Administration | copy, move, rename and delete databases, and to list the contents of a database or indeed all the databases belonging to a site. Physical database parameters may also be changed, and UML model types can be created. |
| Configuration | to deliver, install, uninstall or delete a module from the database in question. Database configuration may also be modified. |
| Repair | unlock a database or re-identify a UML modeling project, a UML profiling project, a document template project or a module. |
| Verification | check the physical or logical state of a database, or the semantic consistency of a database. |

The result of these actions appears in the Objecteering/UML console

The System menu

The "System" menu provides the basic functions of the administration tool.

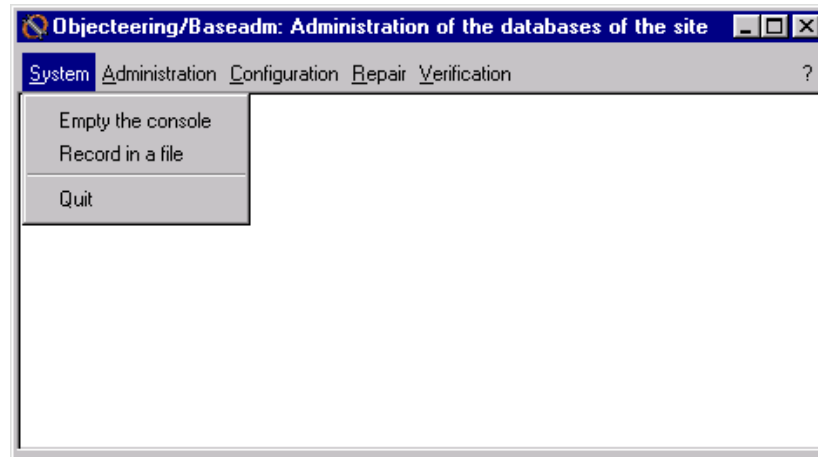


Figure 2-2. The "System" menu

| The ... command | is used to ... |
|-------------------|--|
| Empty the console | clear the content of the database administration window. |
| Record in a file | place in a file everything that is displayed in the console. When this function is run, everything is displayed both in the console and in a file. |
| Quit | quit the administration tool. |

Overview of the Administration menu

Overview

The complex structure of Objectteering/UML data requires that specific services be used to handle databases. Windows or UNIX file administration services must never be directly used to handle the databases. For example, to delete a database, the corresponding file must not be destroyed directly, and when copying a database, you must never copy the associated file.

Objectteering/UML administrative services must be used to carry out these operations.

Locating a site

The administration of a site is centralized in a non-accessible "site" database, whose location is decided upon during the installation procedure. The location of other databases is determined by the administrator.

The "Administration" menu

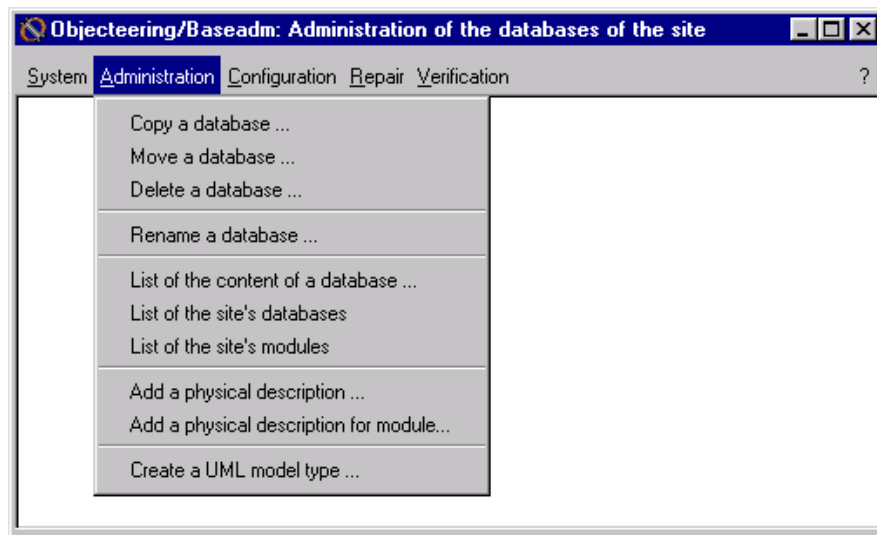


Figure 2-3. The "Administration" menu

| The ... command | is used to ... |
|---------------------------------------|--|
| Copy a database | duplicate a database in a tool. |
| Move a database | move a database to a new location. |
| Delete a database | delete a database. |
| Rename a database | rename a database. |
| List of the content of a database | display the modules, UML modeling projects and UML profiling projects contained within a database. |
| List of the site's databases | display the databases referenced in the site. |
| List of the site's modules | display the modules installed in the site. |
| Add a physical description | provide another physical path, used to access a database from another operating system. This allows you to access a database differently in heterogeneous system configurations. |
| Add a physical description for module | provide another physical path, used to access a module database from another operating system. This allows you to access a module database in heterogeneous system configurations. |
| Create a UML model type | create a "template" database, used when creating UML modeling projects. |

Overview of the Configuration menu

Overview.

The "Configuration" menu in the database administration tool provides the services used to manage modules, which are functionally consistent sets of UML profiles and commands. This includes the delivery of modules to a site, their installation in and uninstallation from a database, and their deletion from a site. Database configuration can also be modified.

The "Configuration" menu

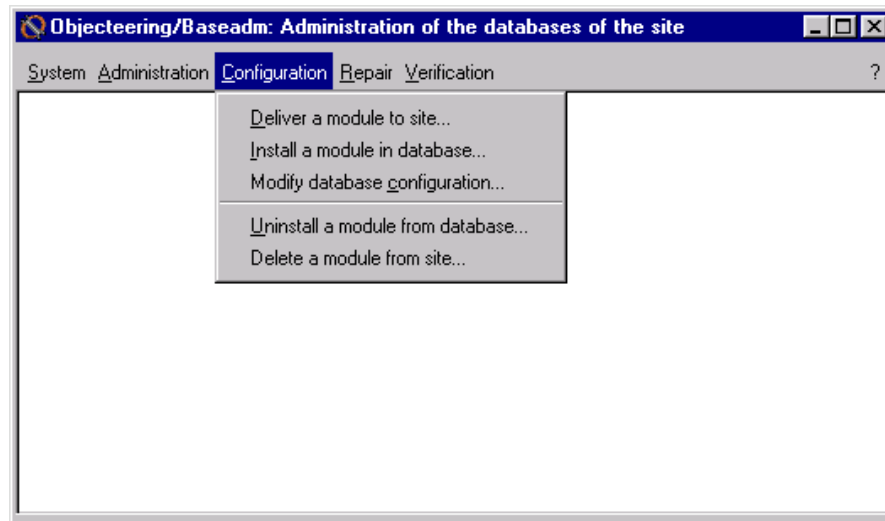


Figure 2-4. The "Configuration" menu

| The ... command | is used to ... |
|--|--|
| <u>D</u> eliver a module to site... | deliver an external module to your site. |
| <u>I</u> nstall a module in database... | install an external module in your database. |
| Modify database <u>c</u> onfiguration... | modify the configuration of the selected database. |
| <u>U</u> ninstall a module from database ... | uninstall a module from the selected database. |
| Delete a module from site... | definitively delete a module from your site. |

Note: When a database is created, the "*_predefinedTypes*" project is always present. For further information on this project, please refer to the "*The _predefinedTypes project*" section in chapter 3 of this user guide.

Overview of the Repair menu

Overview.

Databases can sometimes be corrupted, following system problems (for example, a sudden crash or a full disk), handling errors or a possible problem linked to Objecteering/UML itself.

The consequences of such problems can vary:

- ◆ invisible databases
- ◆ locked databases
- ◆ frequently exiting the tool
- ◆ transfer problems

The Objecteering/UML CASE tool provides a set of mechanisms designed to safeguard and secure your data, which in the vast majority of cases allows you to recover sound and coherent databases, following such problems.

Before carrying out certain operations, you must be familiar with Objecteering/UML identifier logic (the fact that each and every model element, for example, classes, attributes, etc., is identified in a unique way within a UML modeling project). Database administration operations should always be well thought out beforehand.

The "Repair" menu

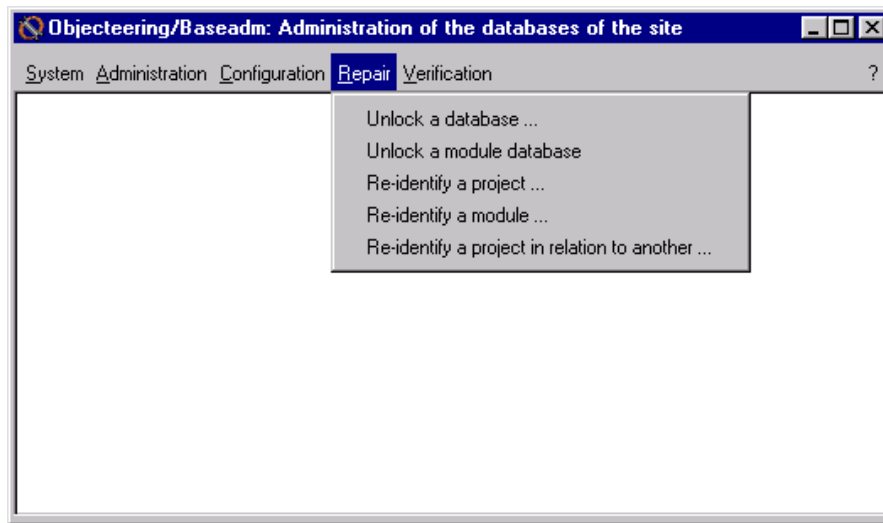


Figure 2-5. The "Repair" menu

| The ... command | is used to ... |
|--|--|
| Unlock a database | unlock a database which is blocked after Objectteering/UML has stopped suddenly. |
| Unlock a module database | unlock a module database. |
| Re-identify a project | allocate a new identifier to a UML modeling, profiling or document template project, as well as to all its components. |
| Re-identify a module | update a module's identifiers from the identifiers of a UML profiling project containing this module. |
| Re-identify a project in relation to another | re-identify the elements of a UML modeling, profiling or document template project, taking into consideration another project. |

Note: A module database is a database in which the "*Deliver a module to site*" command receives modules. For further information on this command, please refer to the "*Detailed view of the Configuration menu*" section in chapter 3 of this user guide.

Overview of the Verification menu

The "Verification" menu

The "Verification" menu of the administration tool allows you to run various checks on a site, on a database, or even on a UML modeling, UML profiling or document template project.

For each check, a "repair" option allows you to run the necessary repairs.

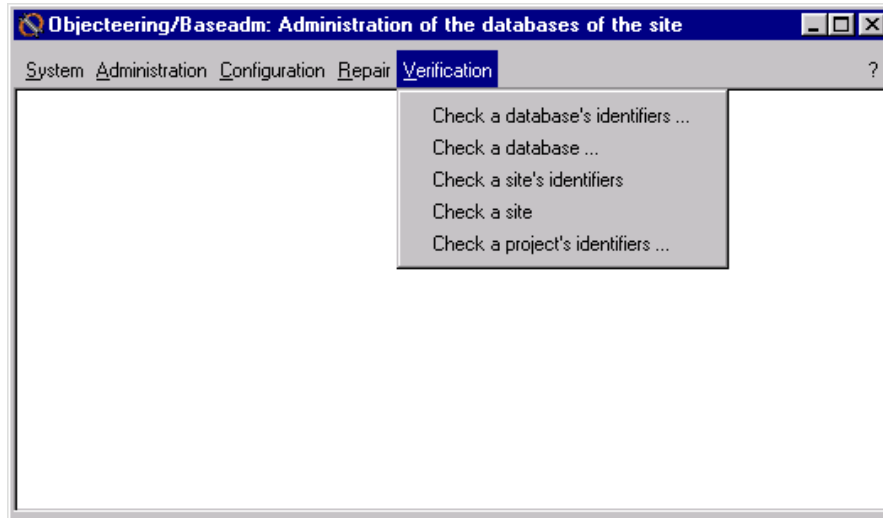


Figure 2-6. The "Verification" menu

| The ... command | is used to ... |
|--------------------------------|---|
| Check a database's identifiers | control the counters and the identifiers of all the database's work projects and their components. |
| Check a database | control the semantic consistency of each of the site's databases |
| Check a site's identifiers | run a check of the identifiers on each of the site's databases. This is to ensure that there are not two or more identifiers which are the same for projects in the site. |
| Check a site | run a check of semantic consistency for each site database. |
| Check a project's identifiers | check project's identifiers (search for two or more identical identifiers) |

Managing identifiers

Each and every model element (classes, attributes, etc.) is identified in a unique way within a project.

The identifiers of a database's elements are especially used for the transfer commands between UML modeling or profiling projects. A transferred element keeps the same identifier. Any conflict (two or more identical identifiers) is a source of error or inconsistency during a transfer.

Note: Conflicts may arise, for example, when manually copying from one database to another. Never use the copy of the files to duplicate databases.

Chapter 3: Detailed view of the
administration tool

Detailed view of the Administration menu

Copying a database

This feature is used to duplicate an existing database in the repository. The window shown in Figure 3-1 displays the list of existing databases.

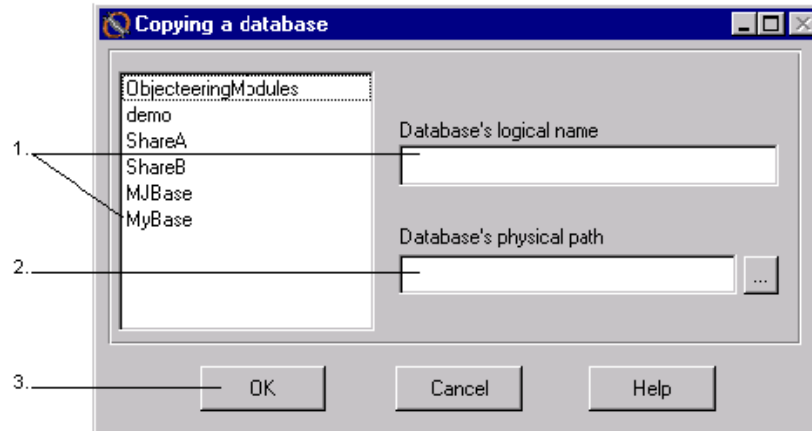



Figure 3-1. The "Copying a database" window

Steps:

- 1 - Enter the name of the database to be copied or double-click on it in the list.
- 2 - Enter the physical path where the new database will be copied or open the "Browse for Folder" window, by clicking on the  button in order to browse your repository's hierarchy.
- 3 - Confirm.

| The ... field | is used to ... |
|--------------------------|--|
| Database's logical name | enter the name of the copied database. |
| Database's physical path | enter the physical path where this new database will be located. |

Syntax

```
baseadm -copy <database name> [<new destination path>] <new  
database name>
```


Moving a database

This command allows you to move a database within your repository.

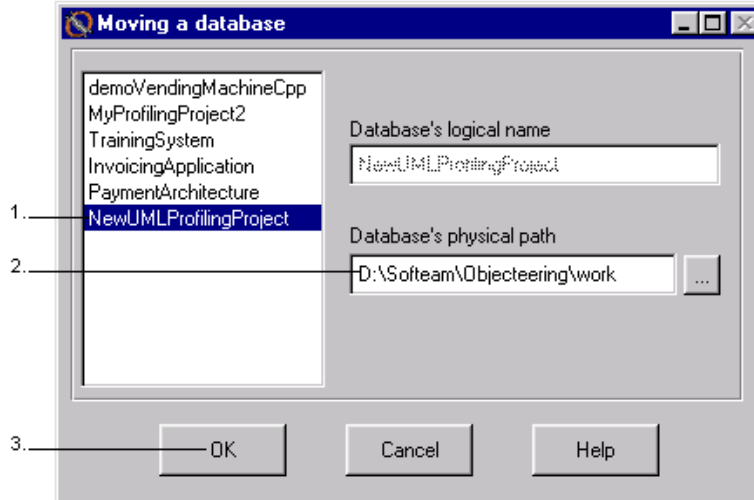



Figure 3-2. The "Moving a database" window

Steps:

- 1 - In the list on the right-hand side of the window, select the database you wish to move.
- 2 - Enter the new physical path of the database. By clicking on the  icon, edit the "Browse for Folder" window, which allows you to browse the repository and to choose a physical location where your database will be moved to.
- 3 - Confirm.

Note: As you can see, the name of the selected database is grayed out in the "Database's logical name" field, and cannot be modified.

| The ... field | is used to ... |
|--------------------------|--|
| Database's logical name | indicate the name of the database which is to be moved. |
| Database's physical path | enter the physical path where the selected database will be renamed. |

Syntax

```
baseadm -move <database name> [<new destination path>]
```

Deleting a database

This command definitively deletes a database. After deletion, the database can no longer be retrieved.

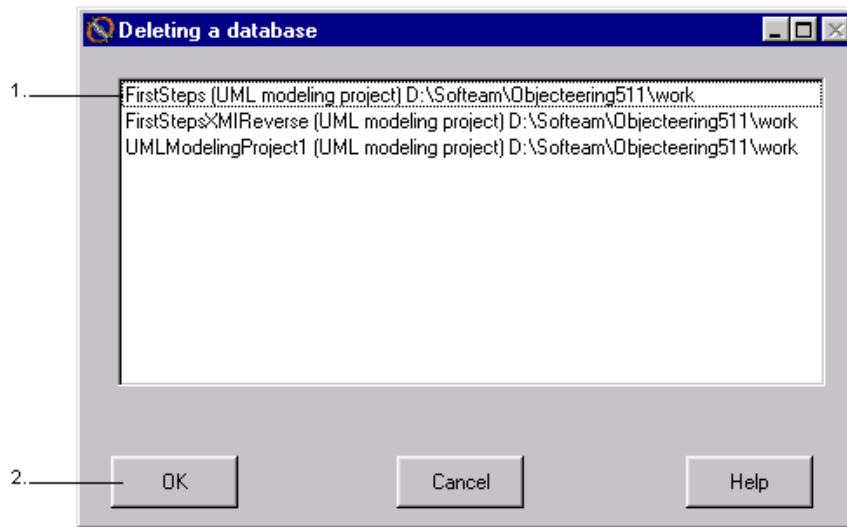


Figure 3-3. The "Deleting a database" window

Steps:

- 1 - Select the database to be deleted (please note that the path of the database is indicated for your information).
- 2 - Confirm.

Syntax

```
baseadm -delete <database name>
```

Note: If your "*site.ifo*" file is locked after the failure of an administration operation (for example, the deletion of a database), the "\$OBJING_PATH/site/lksite" directory must be deleted.

Renaming a database

This command is used to rename an existing database.

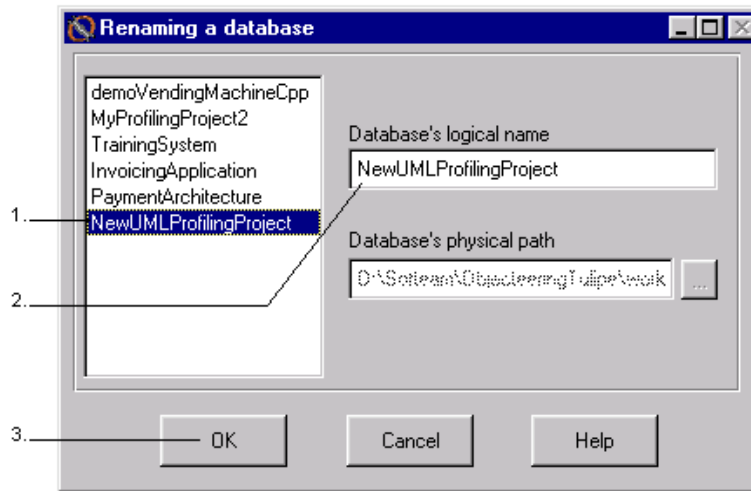


Figure 3-4. The "Renaming a database" window

Steps:

- 1 - In the list on the right-hand side of the window, select the database you wish to rename.
- 2 - Enter the new name in the "Database's logical name" field.
- 3 - Confirm.

Note: As you can see, the location of the selected database is grayed out in the "Database's physical path" field, and cannot be modified.

| The ... field | is used to ... |
|--------------------------|--|
| Database's logical name | enter the new the name of the database which is to be renamed. |
| Database's physical path | indicate the physical path where the selected database is located. |

Syntax

```
baseadm -rename <database name> <new database name>
```

Listing the content of a database

This command allows you to view the contents of a database (modules, UML modeling projects, UML profiling projects and document template projects).

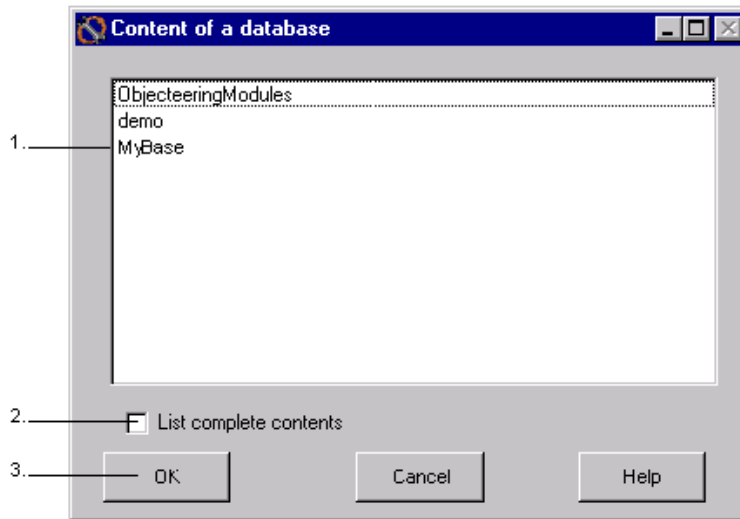


Figure 3-5. The "Content of a database" window

Steps:

- 1 - Select the database whose contents are to be listed.
- 2 - If the "*List complete contents*" tickbox is not checked, only the UML modeling and profiling projects and document template projects contained in the selected database are listed. If this tickbox is checked, then all modules contained in the selected database are also listed, with details on which modules have been selected for each project.
- 3 - Confirm.

Chapter 3: Detailed view of the administration tool

The information concerned is displayed in the administration console (see the example in Figure 3-6).

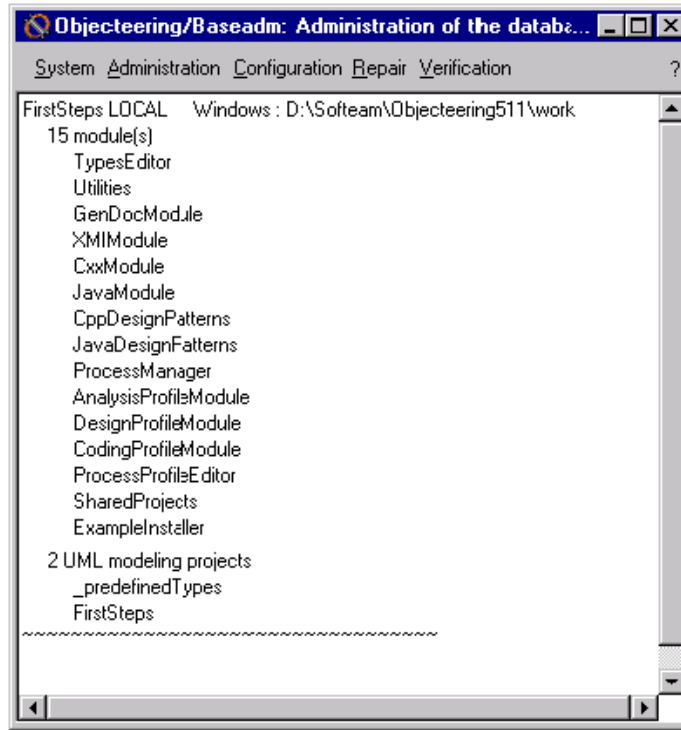


Figure 3-6. Visualizing the contents of the "MyBase" database

The "*System/Empty the console*" command erases the information displayed in the administration console.

Syntax

```
baseadm -list [-all | <database name>] [-complete]]
[-modules]
```

| The ... command | lists ... |
|--|---|
| baseadm -list | the site's databases |
| baseadm -list <database name> | the selected database's contents |
| baseadm -list -all | the contents of all the site databases |
| baseadm -list <database name> -complete | the UML modeling and profiling projects contained within the selected database, as well as the modules contained therein. Details are also provided as to which modules have been selected in each project. |
| baseadm -list -modules | the modules installed in the site |

Listing site databases

This command is used to list the databases referenced in the site in the administration console. The physical path of each database is given, as well as the nature of the database (UML modeling project, UML profiling project or document template project).

A database can have several physical paths. For further information, please refer to the "*Adding a physical description*" theme in this section.

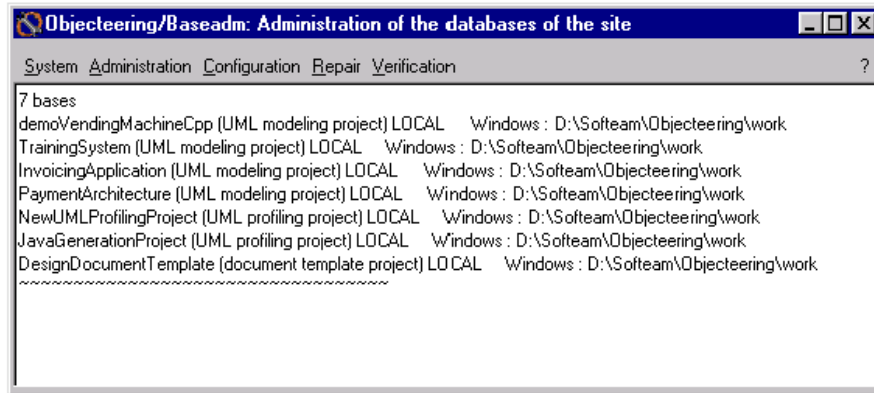


Figure 3-7. The administration window containing the list of site databases

Syntax

```
baseadm -list [-all]
```

List of the site's modules

This command allows you to list those modules installed in your site, along with their version numbers.

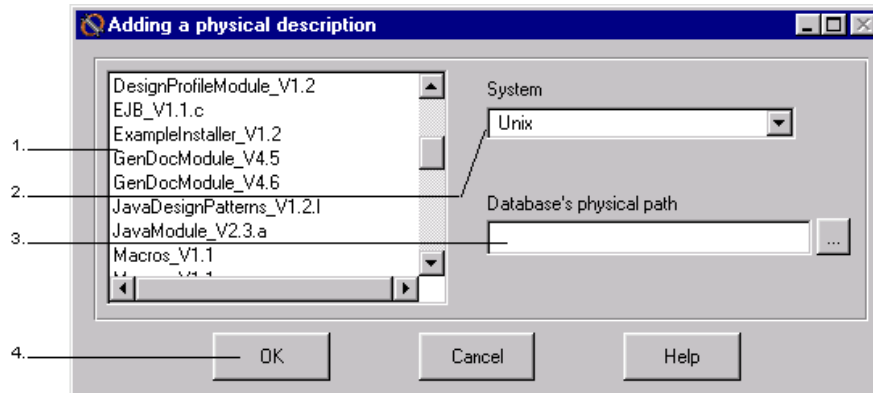


Figure 3-8. The administration window containing the list of site modules

Syntax

```
baseadm -listModules
```

Adding a physical description

For reasons of inter-platform compatibility, Objectteering/UML can access the same database from a UNIX or PC workstation. To do this, each database which can be used on several platforms must have its own access path. The "Adding a physical description" function is provided for this reason. When handling a database with the administration tool, all access paths correspond to the system on which administration is running.

For example, if a database is created from a UNIX work station, the creation path will be a UNIX path. If this database has to be accessible from a PC, it will then be necessary to use the "Adding a physical path" function to specify the access path from this PC.

Warning! As the "move" function cannot modify paths for all the systems, it only carries out the modification for the system on which the administration tool is running.

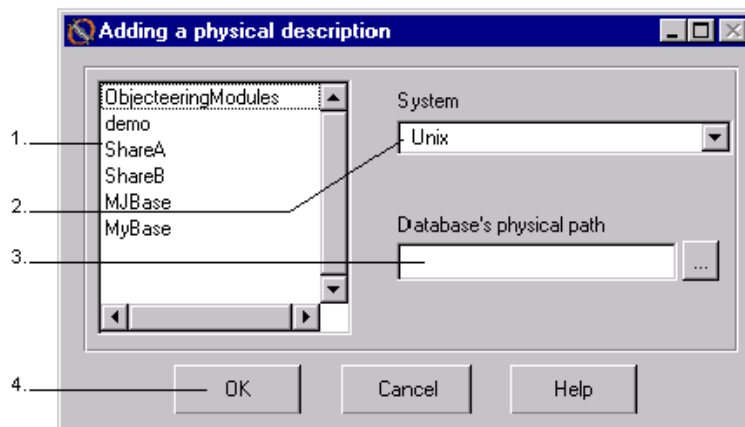



Figure 3-9. The "Adding a physical description" window

Steps:

- 1 - Select the database.
- 2 - Select the system.
- 3 - Enter the database's physical path. By clicking on the  icon, edit the "Browse for Folder" window, which allows you to browse the repository and to choose a physical description for your database.
- 4 - Confirm.

Syntax

```
baseadm -addPath <database name> <OS> <database path>
```

Adding a physical description for modules

This command is similar to the "Add a physical description" command previously described, but is used with module databases.

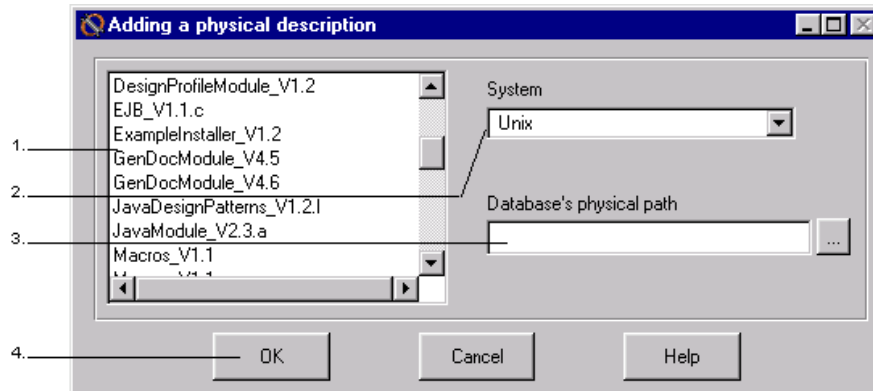



Figure 3-10. The "Adding a physical description" window

Steps:

- 1 - Select the module database.
- 2 - Select the system.
- 3 - Enter the module database's physical path. By clicking on the  icon, edit the "Browse for Folder" window, which allows you to browse the repository and to choose a physical description for your module database.
- 4 - Confirm.

Syntax

```
baseadm -addPath <database name> <OS> <database path> [-  
modules]
```

Creating a UML model type

This command is used to create a new UML model type, in other words, a database "template" which it will then be possible to use when creating new UML modeling projects.

UML model types are very practical, since they pre-configure your new database with regard to modules installed and selected. UML model types are located in the \$OBJJING_PATH/template directory.

The UML modeling project which has been transformed into a UML model type is no longer available after this operation, but its associated file is saved in a .SAVE file.

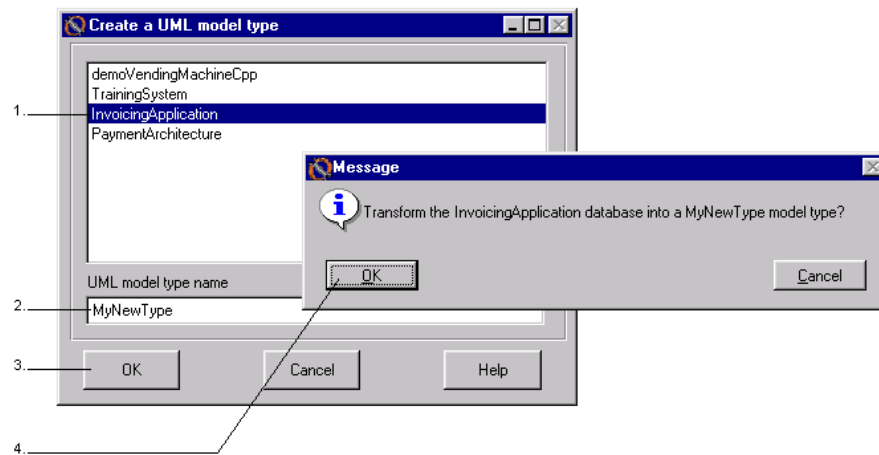


Figure 3-11. The "Create a UML model type" window and confirmation box

Chapter 3: Detailed view of the administration tool

Steps:

- 1 - Select the UML modeling project you wish to transform into a UML model type.
- 2 - Give a name to your new UML model type.
- 3 - Confirm by clicking on "OK".
- 4 - A confirmation dialog box then appears, asking you if you really want to transform you UML modeling project into a UML model type. Click on "OK" to confirm.

Note: Where a later version of a module used in a UML model type is deployed, the UML model type must be created again, so as to allow the user to take advantage of the new version. If this is not done, then any UML modeling projects created using the UML model type will continue to use the earlier version of the module.

Note: When a UML model type is created by a heavyweight client, it is, in fact, created on the server. When a new database is created, the list of available UML model types is only available on the server.

Syntax

```
baseadm -moveAsTemplate <database name> [<template name>]
```

Detailed view of the Configuration menu

Overview

The administration services which are available in the "*Configuration*" menu of the baseadm tool are used to manage modules, as well as to handle the configuration of the database.

Delivering a module to a site

If you wish to use modules developed externally, either by yourself or by other third parties, you should simply "*deliver*" them to your site and then install them in the selected database(s) as shown later in this section. To deliver a module to a site, follow the steps detailed below (Figure 3-12).

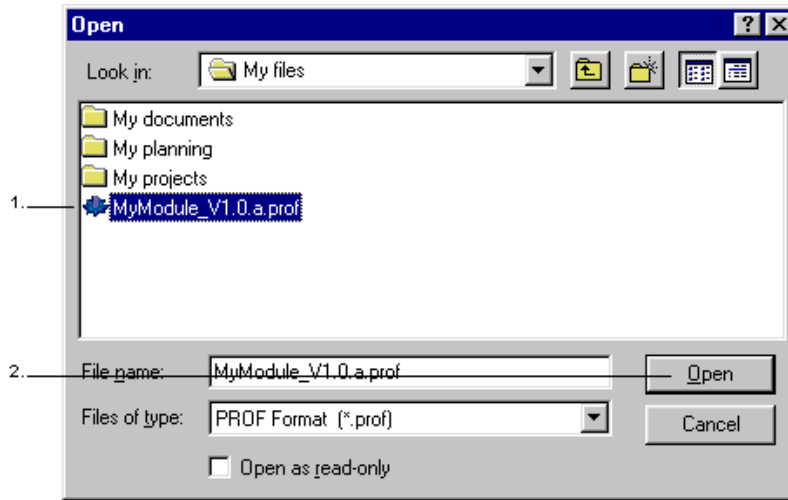


Figure 3-12. Delivering a module to a site

Steps:

- 1 - Click on "*Configuration/Deliver a module to site*" and then select the .prof file for the module you wish to deliver from the window which appears.
- 2 - Confirm by clicking on "*Open*".

Note: A .prof file is a module delivery file, which contains everything necessary to the installation and use of the module in the selected site and the selected database.

The result of the "*Deliver a module to site*" command is displayed in the administration tool console (as shown in Figure 3-13).

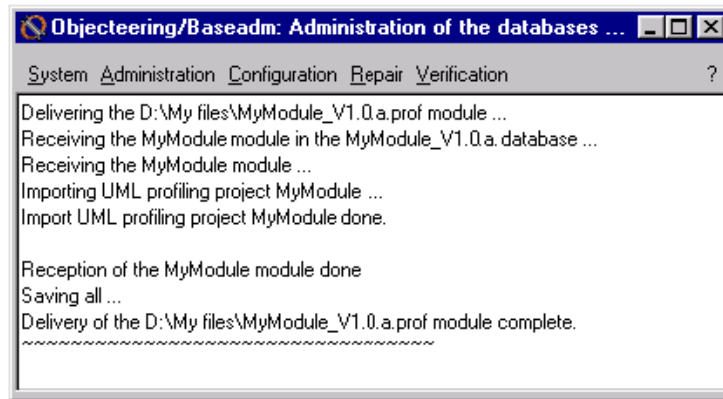



Figure 3-13. Result of the "*Deliver a module to site*" command displayed in the console

Once the module has been delivered, it is shown in your explorer. It is represented by the  icon, in order to make it easily distinguishable.

Syntax

```
baseadm -deliver <deliverModuleFileName> [<database name>*]
```

Installing a module in a database

Before a module can be installed in a database, it must previously have been delivered to the site containing the database in question (please see the previous theme in this help section).

To install a module, carry out the steps illustrated in Figure 3-14.

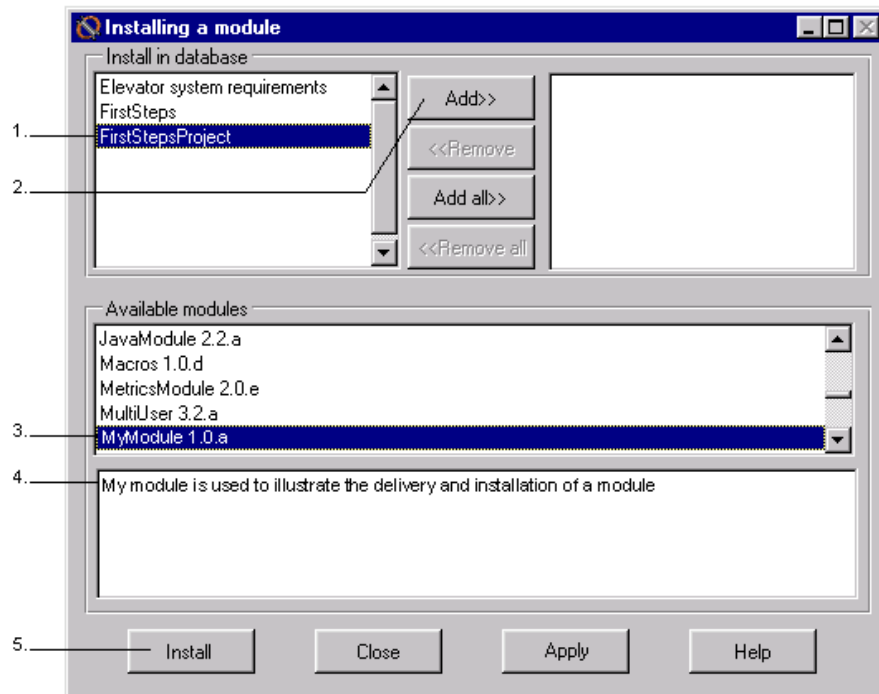


Figure 3-14. Installing a module in a database

Steps:

- 1 - Click on "*Configuration/Install a module in database...*". In the "*Install in database*" window, select the database in which you wish to install the module. In our example, we have chosen the "*FirstStepsProject*" database.
- 2 - Click on "*Add*". The selected database then appears in the right-hand list.
- 3 - In the "*Available modules*" window, select the module you wish to install. In our example, we are going to install the "*MyModule 1.0.a*" module (which is the module previously delivered to the site in our example).
- 4 - Please note that the lower window provides a description of the selected module.
- 5 - Click on "*Install*" to confirm your choice and close this dialog box, or on "*Apply*" to confirm your choice and keep the dialog box open.

The result of the "*Install a module in database*" command is displayed in the console (as shown in Figure 3-15).

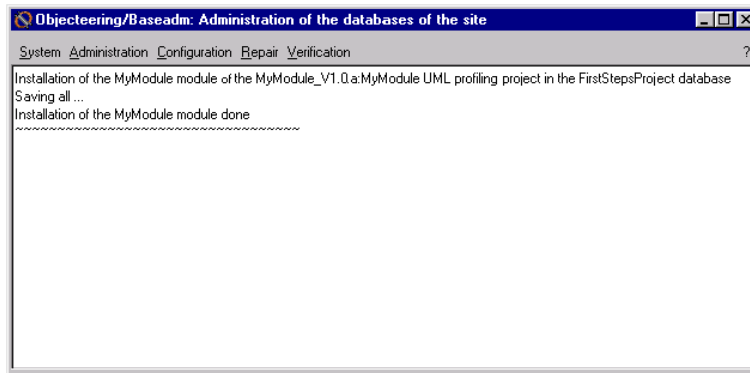


Figure 3-15. Result of the "*Install a module in site*" command displayed in the console

Syntax

```
baseadm -install <module name> [version <version target>]
<database name> [<database name>]* [-from <database name>]
```

Modifying database configuration

To modify database configuration, carry out the operations illustrated in Figure 3-16.

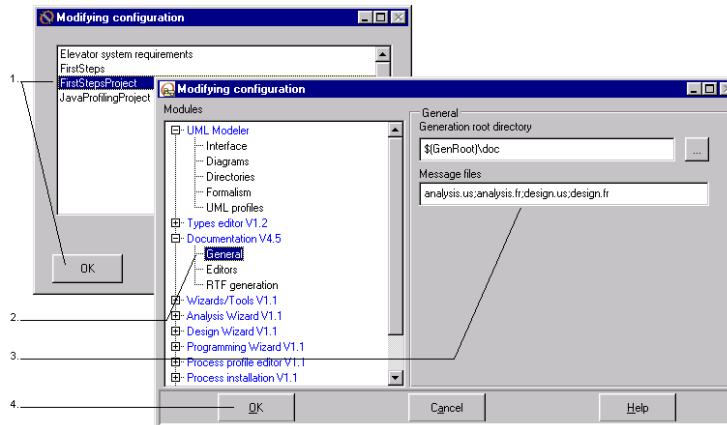


Figure 3-16. Modifying the configuration of the "MyBase" database

Steps:

- 1 - Select "*Configuration/Modify database configuration*" and then select the database whose configuration you wish to modify. Confirm by clicking on "OK". The "*Edit configuration*" dialog box then appears.
- 2 - In this dialog box, the parameters for all the modules installed in the selected database are presented in hierarchical form. In our example, we are going to modify the configuration of the *Documentation* module at database level.
- 3 - Make the desired changes to configuration.
- 4 - Confirm by clicking on "OK".

Note 1: As you can see, certain modules have several sub-entries in their hierarchy, whilst others only have one.

Note 2: Modifying configuration at database level means that any modifications you make will be taken into account by all the UML modeling projects contained in the database in question.

Uninstalling a module from a database

To uninstall a module from a database, carry out the steps detailed below.

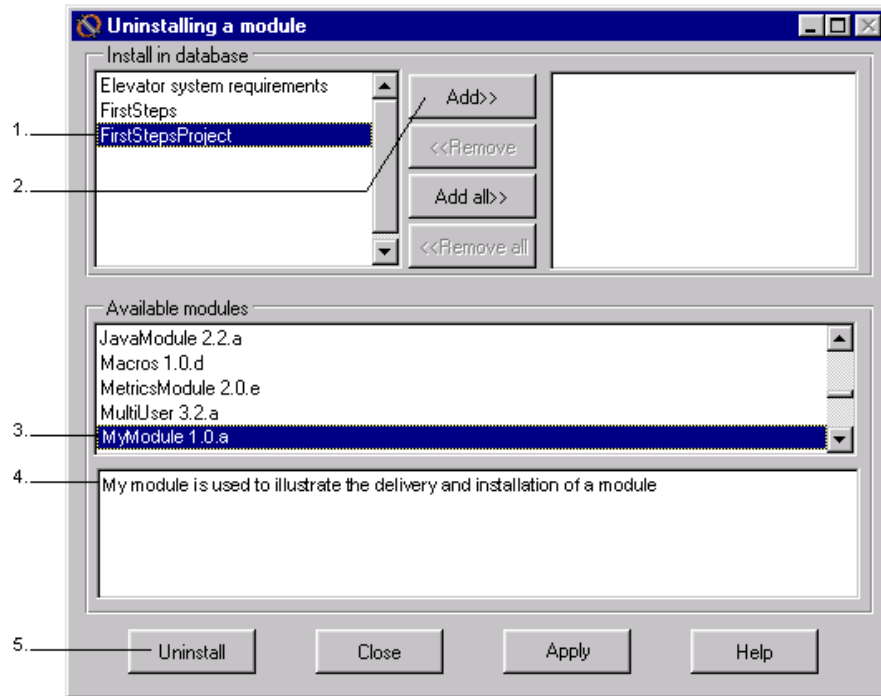


Figure 3-17. Uninstalling a module from a database

Steps:

- 1 - Select "*Configuration/Uninstall a module from database...*". In the "*Uninstall*" window, select the database from which you wish to uninstall the module. In our example, we have chosen the "*FirstStepsProject*" database.
- 2 - Click on "*Add*". The selected database then appears in the right-hand list.
- 3 - In the "*Available modules*" window, select the module you wish to uninstall. In our example, we are going to uninstall the "*MyModule*" module, previously delivered and installed in our example.
- 4 - Please note that the lower window provides a description of the selected module.
- 5 - Click on "*Uninstall*" to confirm your choice and close this dialog box, or on "*Apply*" to confirm your choice and keep the dialog box open.

Note: You can track the progress of module uninstallation in the console.

Syntax

```
baseadm -uninstall <module name> <database name> [<database name>] *
```


Deleting a module from a site

To delete a module from a site, carry out the steps detailed below.

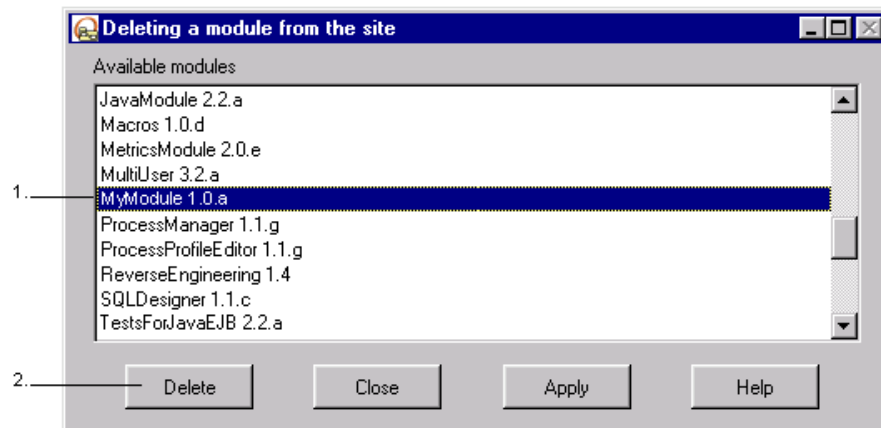


Figure 3-18. Uninstalling the "MyModule" module from the site

Steps:

- 1 - Click on "*Configuration/Delete a module from site*" and then select the module you wish to delete from the site. In our example, we are going to delete the "*MyModule*" module, which has previously been delivered, installed and then uninstalled.
- 2 - Click on "*Delete*" to confirm your choice and close this dialog box, or on "*Apply*" to confirm your choice and keep the dialog box open.

Syntax

```
baseadm -suppress <module name> [-version <version target>]
```

Detailed view of the Repair menu

Overview

The various repair services provided in the administration tool are used to repair databases, UML modeling projects, UML profiling projects or modules and to re-identify those elements of a UML modeling or profiling project taken into consideration by another UML modeling or profiling project.

Unlocking a database

When Objectteering/UML has stopped suddenly, the *"Unlock a database"* operation should be run. The database is put in the last coherent state resulting from a complete backup. If problems occurred at the time of backup, the state retrieved will be the one previous to the backup.

The *"Unlock a database"* option should be used when an error message indicates that the database is *"locked"* after trying to open a database. If a database is locked during a save, the unlocking function retrieves transactions which were running at the time.

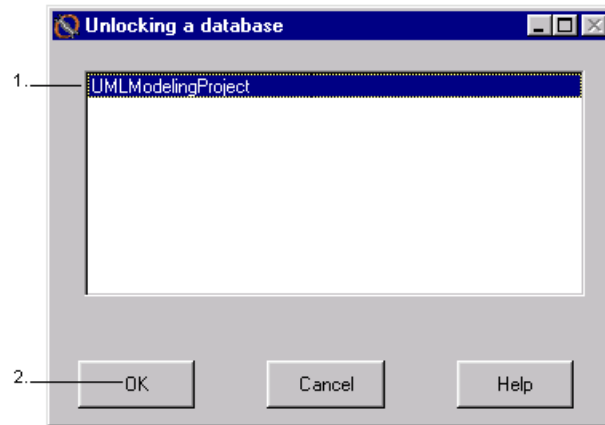


Figure 3-19. The *"Unlocking a database"* window opened by the *"Unlock a database"* command

Steps:

- 1 - Select the database to be unlocked.
- 2 - Confirm.

The administration tool's console displays the repairs carried out in the database.

Syntax

```
baseadm -unlock <database name>
```

Unlocking a module database

When Objectteering/UML has not been able to complete a module installation operation, the "*Unlock a module database*" operation should be run. This command puts the database in the last coherent state resulting from a complete backup.

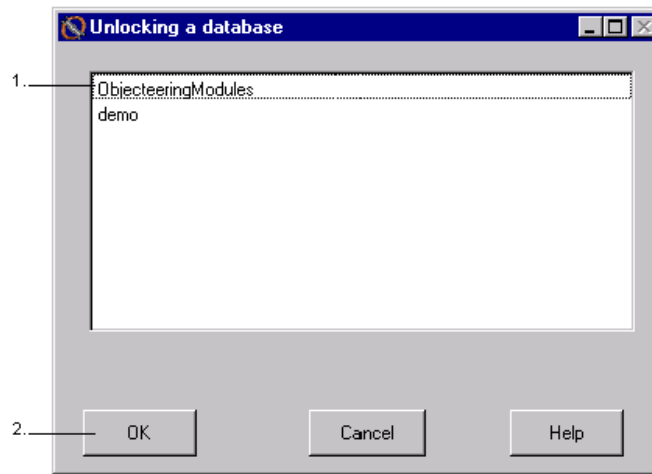


Figure 3-20. The "*Unlocking a database*" window opened by the "*Unlock a module database*" command

Steps:

- 1 - Select the module database to be unlocked.
- 2 - Confirm.

The administration tool's console displays the repairs carried out in the module database selected.

Syntax

```
baseadm -unlock <database name> [-modules]
```

Re-identification operations

When an object is created, Objectteering/UML attributes it an identifier, which cannot subsequently be modified. This identifier is used to recognize an object (or a copy of the object obtained after an *"Import"* operation) in two different databases. It is also used to find module elements (tagged values, note types etc.), so as to recreate links towards these elements (between the tagged element and the tagged value, for example).

Object identifiers are created from a counter stored in the UML modeling project.

The following operations are used to repair identifier problems:

- ◆ *"Re-identify a project"*
- ◆ *"Re-identify a module"*
- ◆ *"Re-identify a project in relation to another"*

Re-identification logic

Re-identification is used to re-compose a database in which two elements have the same identifier. This operation has many consequences, and should only be run when identifier repetition occurs.

Re-identifying a project changes the identifiers of all its components. If some of these exist in other databases, Objectteering/UML will no longer recognize them as being identical. It is important to note that transfer operations, for example, will no longer function. To avoid this, we recommended that you use the *"Re-identify in relation to another project"* service, if another project can be a reference for the other identifiers.

Re-identifying a project

This feature is used to give a new identifier to each element (contained within a UML modeling project, a UML profiling project or a document template), as if these elements had just been created.

The identifiers of module elements installed in the database (tagged values, note types, etc.) are not affected.

Warning: This project re-identification operation is not a simple maintenance operation, and should only be used in rare cases, as improper use could have serious consequences. For example, the loss of an original identifier prevents all updates using the "Import" command, which leads to the duplication of UML elements.

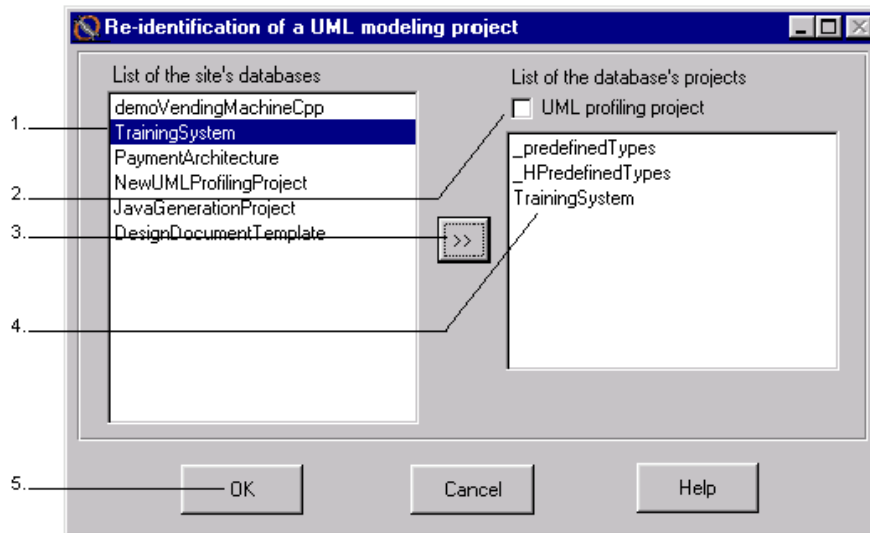



Figure 3-21. The window for re-identifying a project

Steps:

- 1 - Select a database in the list of the site's databases.
- 2 - If you wish to see the list of UML profiling projects, check the "UML profiling project" button



- 3 - Click on the  button.
- 4 - Select the UML modeling or UML profiling project from the list.
- 5 - Confirm.

The console now displays the repair messages.

| The ... field | represents ... |
|---------------------------------|--|
| List of the site's databases | the content of the existing databases. |
| List of the database's projects | the content of the projects for a selected database. |

Syntax

```
baseadm -reidentify <database name> <UML modeling project
name> [-profiling]
```

Typical use of the project re-identification operation

One of the most common reasons for the appearance of repeated element identifiers is when a database is copied without using the Objecteering/UML administration tool (copying an .ofp file over another existing file).

If Objecteering/UML does not detect that the database has been duplicated, it does not exchange counters between the two databases, and consequently elements created in the databases receive the same identifier. This means that it is impossible to import data from these two databases. To solve this problem, at least one of these databases must be re-identified, before using the "*Re-identify a project in relation to another*" command on the other database.

Note: If you encounter a problem regarding the repetition of identifiers within the same database, we recommend against using this command. We suggest instead that you use the "*Check a project's identifiers*" command (with the "*Run the repair*" tickbox checked), as this will only repair repeated elements.

Re-identifying a module

This operation gives each element of a module installed in a database (tagged values, note types, etc.) the same identifier as that of the element of the same type, with the same name and in the same location within the reference profile.

There is usually no need to use the "*Re-identify a module...*" command. A module is defined and packaged using the *Objecteering/UML Profile Builder* tool, and is then delivered to and installed in the databases where it is to be used. Its content should not change.

In the most serious cases of database corruption, a module can be re-installed in the usual way.

It is compulsory to re-identify the module, but only after having re-identified the UML profiling project that contains the module, and all the databases in which this module is installed. If you do not do this, it will not be possible to re-install an upgraded version of the module (because of identifier repetition).

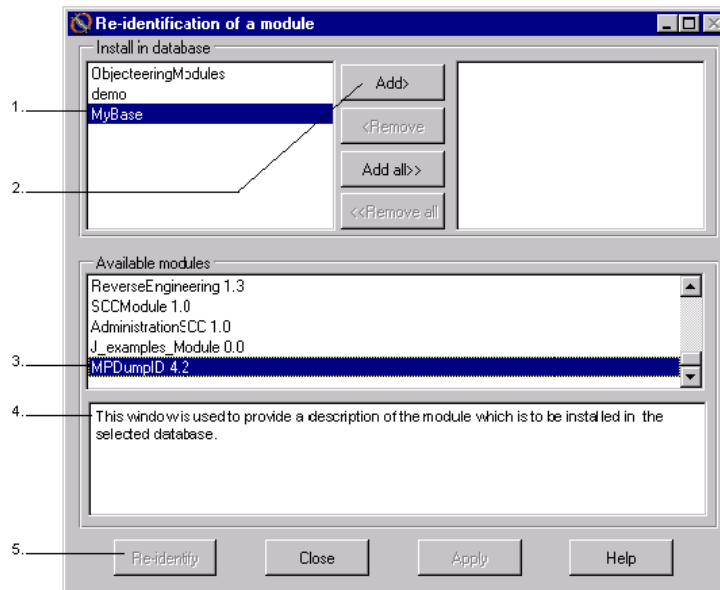


Figure 3-22. The window for re-identifying a module

Chapter 3: Detailed view of the administration tool

Steps:

- 1 - Click on "*Repair/Re-identify a module...*". In the "*Install in database*" window, select the database in which you wish to install the module. In our example, we have chosen the "*MyBase*" database.
- 2 - Click on "*Add*". The selected database then appears in the right-hand list.
- 3 - In the "*Available modules*" window, select the module you wish to re-identify. In our example, we are going to re-identify the "*MPDumpID 4.2*" module.
- 4 - Please note that the lower window provides a description of the selected module.
- 5 - Click on "*Re-identify*" to confirm your choice and close this dialog box, or on "*Apply*" to confirm your choice and keep the dialog box open.

Note: You can track the progress of module re-identification in the console.

| The ... field | is used to ... |
|---------------------|--|
| Install in database | display all the site's databases. Individual databases are selected by clicking on the database in question and then clicking on the " <i>Add</i> " button. Similarly, to remove database, click on it and then click on the " <i>Remove</i> " button. |
| Available modules | display all the modules of the selected database. |

Syntax

```
baseadm -reidentify <database name> <module name> < ref.  
database name> <ref. UML modeling project name> [-profiling]
```

Typical use of the module re-identification operation

The "*Re-identify a module*" command is typically used where a profile developer has re-identified his development database or corrected identifier repetition problems and then created a new version of the module. In a database which uses an earlier version of the module, this command is used to update the module without deleting the earlier version (and therefore losing tagged values, note types and so on in the module).

Re-identifying a project by comparison

This operation is used to give each element (contained within a UML modeling project, a UML profiling project or a document template) the same identifier as the UML element of the same type, with the same name and in the same location within the reference UML model. This operation is indispensable to enable the transfer of data between two projects, after having re-identified the first project.

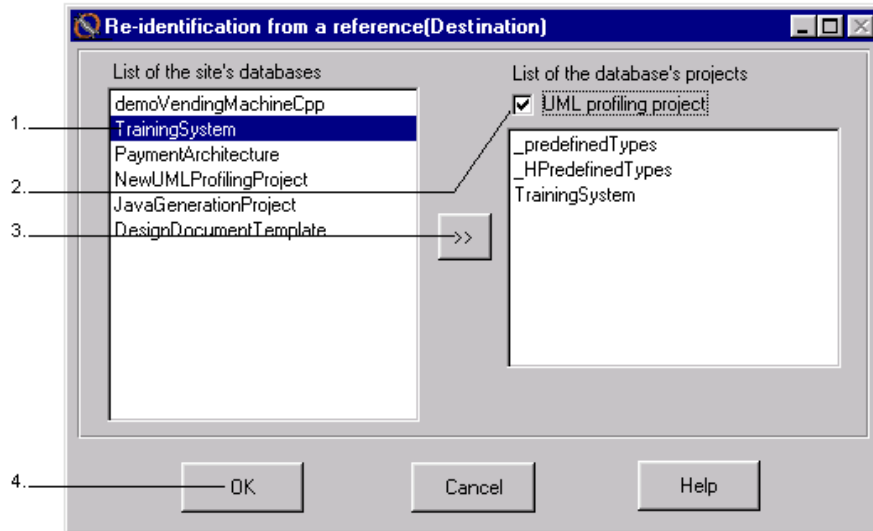



Figure 3-23. The window for re-identifying a project by comparison

Chapter 3: Detailed view of the administration tool

Steps:

- 1 - Select the database in the list.
- 2 - Click on the "UML profiling project" button if you wish to see the list of UML profiling projects.



- 3 - Click on the  button to edit the list of UML modeling and profiling projects.
- 4 - Confirm.
- 5 - Carry out steps 1 to 3 again to select the reference UML modeling or profiling project.

The administration tool's console displays a trace the re-identification operations carried out in the database.

| The ... field | displays ... |
|---------------------------------|---|
| List of the site's databases | all the site's databases. |
| List of the database's projects | the list of all the UML modeling or profiling projects or UML profiling projects of a database. |

Syntax

```
baseadm - reidentifyFrom <database name> <UML modeling  
project name> < ref. database name> <ref. UML modeling  
project name> [-profiling]
```

Typical use of the project re-identification by comparison operation

This operation can be used to resolve a problem caused by the appropriate use of the "*Re-identify a project...*" operation.

It can also be used to solve identifier repetition repair problems within a database, so that elements re-identified by the repair have the same identifier in all the site's databases.

Regenerating the module XML file

This command is not available through the *Objecteering/Administration* tool's menus. It regenerates the module description file from the modules contained in module databases. The previous file is saved in `modules.xml.bak`

The syntax of the command is as follows:

```
baseadm -generateModuleXMLFile
```

Adding a description of modules to the XML file

This command is not available through the *Objecteering/Administration* tool's menus. It adds the description of the modules contained in the `<database name>` database to the module description file. The database should be defined in the module databases.

The syntax of this command is as follows:

```
baseadm -addToModuleXMLFile <database name>
```

Detailed view of the Verification menu

Overview

Various Objectteering/UML checks can be performed at different levels.

| The checking of... | involves ... |
|-------------------------------------|---|
| a site | all the site's databases, the UML modeling and profiling projects of all the databases. |
| a database | all the UML modeling and profiling projects in a database. |
| a UML modeling or profiling project | all the identifiers of a UML modeling and profiling project. |

For each check, a "*Run the repair*" option is used to run the necessary repairs immediately. This must be selected before confirmation.

Checking database identifiers

This verification checks the counters of the UML modeling and profiling projects (no object can be larger than the counter), as well as the possible repetition of identifiers of the UML modeling and profiling projects in the selected database.

If the "*Run the repair*" option is selected, a new identifier will be given each time there is a repetition.

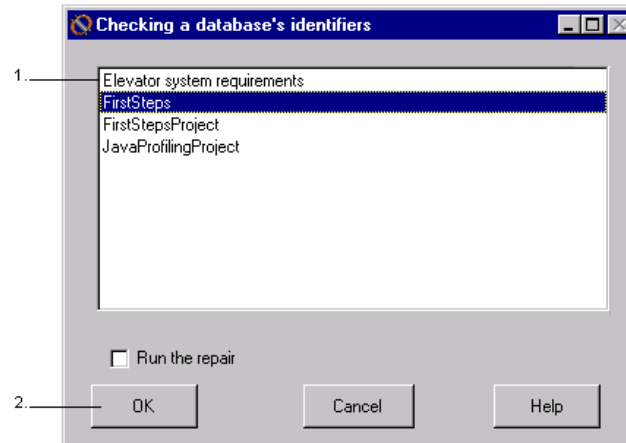


Figure 3-24. Window for checking a database's identifiers

Steps:

- 1 - Select the database to be checked.
- 2 - Confirm.

The administration tool console displays the errors and the repairs to be carried out if the "*Run the repair*" option has not been selected. If it has, it displays details on any repairs that were carried out.

Syntax

```
baseadm -checkBaseId <database name> [-repair]
```

Checking a database

The physical structure of objects is checked and objects that are not physically valid are deleted.

Semantic consistency checks of the model are carried out on all objects. There is no automatic repair as a result of these checks. Rather, those in charge of the model in the database must open it in *Objectteering/UML Modeler*, and correct the listed inconsistencies.

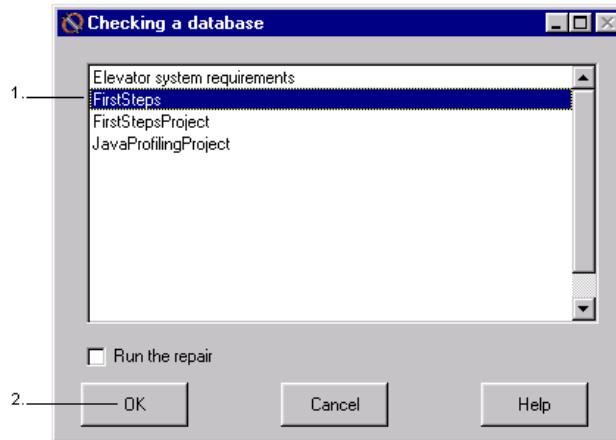


Figure 3-25. Window for checking a database

Steps:

- 1 - Select the database to be checked.
- 2 - Confirm.

You may observe in the administration tool's console the checks that have been carried out, as well as the repairs if the "*Run the repair*" option has been selected.

Syntax

```
baseadm -checkBase <database name> [-repair]
```


Checking site identifiers

This feature checks consistency between the different databases. All the site's databases, as well as their UML modeling and profiling projects, will be checked.

Note: Depending on the size of the site, this process can take some time.

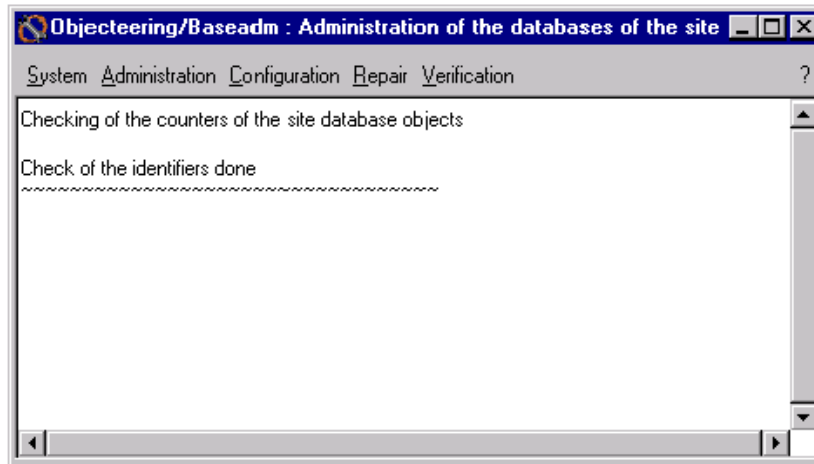


Figure 3-26. Main window containing the information related to the verification of the site's identifiers

To clear the contents of the console, run the *"Empty the console"* command in the *"System"* menu.

Syntax

```
baseadm -checkSiteId [-repair]
```

Checking the site

All the site's databases, as well as their UML modeling and profiling projects, will be checked, and the console will display any errors that may have occurred.

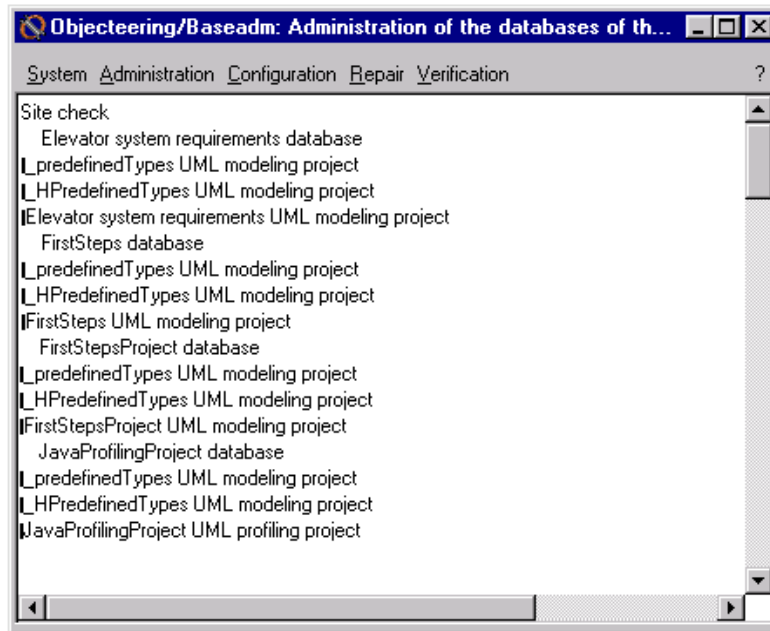


Figure 3-27. Main window containing the information related to site verification

To empty the console content, use in the "System" menu, the "Empty the console" option.

Syntax

```
baseadm -checkSite [-repair]
```

Checking project identifiers

This function allows you to check the consistency of a project's identifiers.

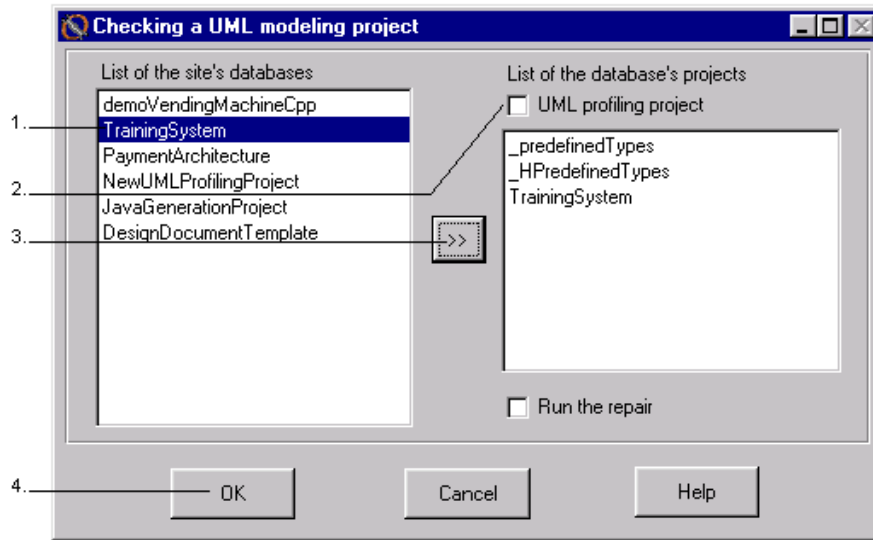



Figure 3-28. Window for checking a UML modeling project

Steps:

- 1 - Select the database.



- 2 - Click on the  button to display the list of UML modeling and profiling projects, or double-click on the database.

- 3 - Select a UML modeling or a profiling project.

- 4 - Confirm.

Chapter 3: Detailed view of the administration tool

The "*Run the repair*" option, if selected, will proceed with an immediate repair.

The administration tool's console will display the errors and the repairs that have been carried out (if the option has been selected).

| The ... field | displays ... |
|---------------------------------|--|
| List of the site's databases | all the site's databases |
| List of the database's projects | all the UML modeling or profiling projects of a database |

Syntax

```
baseadm checkProjectId <database name> <UML modeling project  
name> [-profiling] [-repair]
```

Note: If the name of the project is the same as the name of the database, you no longer need to indicate it.

Repairing databases

Overview

Because of certain rare anomalies which can provoke the sudden exit of the baseadm tool during verification operations ("*checkBase*" or "*checkProject*"), you have at your disposal, in command line only, an option which can be used to repair databases where necessary.

Syntax

```
baseadm -repairComponent <database name> <UML modeling  
project name> [-profiling]
```

Procedure

Before launching this command, the database must be unlocked.

This option loads all elements of the UML modeling project which are indicated, and checks the hierarchy. If an element is found twice in this hierarchy, a repair is carried out, as well as a save, and a message informs the user of the element concerned.

Once this command has been carried out, it is preferable to launch a verification of the UML modeling project, in order to be sure that the database is correct (the "*checkProject*" option).

The `_predefinedTypes` project

When a database is created, the "`_predefinedTypes`" UML modeling project is always present. This project has a specific purpose, and must not generally be used to build your models. The packages defined in this project are used by default by all the packages of any other project of the same database.

The "`S_predefinedTypes`" package contains Objecteering/UML predefined types, such as "integer", "string", and so on. Other packages such as "`JavaTypes`" exist, and are used for code generation purposes. Accessors and basic type declarations are specified and may be customized here. Our Objecteering/UML code generation user guides, such as the *Objecteering/Java* user guide ("Enterprise" edition only), explain how to use this for customization purposes.

Example:

We are going to create a "`date`" class, visible by all the models:

- 1 - Create a new package in the "`_PredefinedTypes`" project.
- 2 - In this package, create the "`date`" class, with the "`primitive`" nature.

Result: When entering an attribute in any project in the same database, the "`date`" type will be available.

The "*S_PredefinedTypes*" package is delivered by default with Objecteering/UML. It defines a set of database primitive classes:

| The ... primitive class | represents ... |
|-------------------------|--|
| integer | integers |
| boolean | boolean, with "FALSE" or "TRUE" values |
| real | float reals |
| char | characters |
| string | character strings. They can have the following size: <ul style="list-style-type: none">- unlimited: string (*)- undefined: string ()- imposed at N: string (N) |
| undefined | a non allocated type. This is found, for example, in the case of the attributes or parameters for which the user has not supplied a type. |

Note: all these types can be redefined except "*undefined*".

Index

- .prof file 3-22
- Accessor 3-50
- Adding a description of modules to the XML file 3-41
- Adding a physical description 3-16
 - Syntax 3-17
- Adding a physical description for a module
 - Syntax 3-18
- Administration functioning modes
 - Batch mode 2-3
- Administration functioning modes
 - Interactive mode 2-3
- Administration menu 2-4
 - Commands 2-7
 - Overview 2-6
- Administration menu commands
 - Add a physical description 3-16
 - Add a physical description for module 3-18
 - Copy a database 3-3
 - Creating a UML model type 3-19
 - Delete a database 3-7
 - List of the content of a database 3-11
 - List of the site's databases 3-14
 - List of the site's modules 3-15
 - Move a database 3-5
 - Rename a database 3-9
- Administration tool
 - Administration menu 2-6
 - Configuration menu 2-9
 - Repair menu 2-11
 - System menu 2-5
 - Verification menu 2-14
- Administration tools
 - Menus 2-4
- ASCII 1-5
- Attribute 2-11, 2-15
- Batch syntax 2-3
- Checking a database 3-44
 - Syntax 3-44
- Checking a project's identifiers 3-36
- Checking database identifiers 3-43
 - Syntax 3-43
- Checking project identifiers 3-47
 - Syntax 3-48
- Checking site identifiers 3-45
 - Syntax 3-45
- Checking the site 3-46
 - Syntax 3-46
- Class 2-11, 2-15
- Configuration menu 2-4
 - Commands 2-9
 - Overview 2-9
- Configuration menu commands
 - Delete a module from a site 3-29
 - Deliver a module to a site 3-22
 - Install a module in database 3-24
 - Modifying database configuration 3-26
 - Uninstalling a module from a database 3-27
- Consistency checks 3-44
- Console 2-4, 3-12, 3-14, 3-23, 3-31, 3-32, 3-35, 3-40, 3-43, 3-44, 3-46, 3-48
- Copying a database 2-6, 3-3
 - Syntax 3-4
- Creating a UML model type 3-19
 - Syntax 3-20
- Data exchange services 1-5

- Database 1-3, 1-5, 1-7, 2-3, 2-6, 2-11, 3-11, 3-14, 3-16, 3-33, 3-46, 3-50
- Database configuration 1-3, 2-9
- Database file access path 1-7
- Database parameters 1-3
- Deleting a database 2-6, 3-7
 - Syntax 3-8
- Deleting a module from a site 3-29
 - Syntax 3-29
- Delivering a module to a site 3-22
 - Syntax 3-23
- Document template project 1-3
- Document template projects 1-4
- Duplication 3-36
- Enterprise Edition 3-50
- Exchanging data between UML modeling projects 1-5
- Exchanging data between UML profiling projects 1-5
- Exchanging information between sites 1-5
 - Methods 1-5
 - XMI exchange 1-5
- Externalization/internalization services 1-6
- Externalizing a module 1-6
- Externalizing database contents 1-5
- File identifier 1-7
- FP format 1-7
- Identifier 3-33, 3-39
- Identifiers 3-33
- Installing a module in a database 3-24
 - Syntax 3-25
- Installing modules 1-5
- Inter-platform compatibility 3-16
- Invisible databases 2-11
- J Language 1-3
- Listing site databases 3-14
 - Syntax 3-14
- Listing site modules 3-15
 - Syntax 3-15
- Listing the content of a database 3-11
 - Syntax 3-13
- Locating a site 2-6
- Locked databases 2-11
- Main window 2-4
- Managing database configuration 1-3
- Managing physical database parameters 1-3
- Managing projects 1-3
- Model element identifiers 2-15
- Model element traceability 1-4
- Model elements 1-4
- Modifying database configuration 3-26
- Module 1-3, 1-6, 3-11, 3-30, 3-37
- Module commands 2-9
- Module parameterization 1-3
- Modules 1-7, 2-3
- Modules.ifo 1-7
- Modules.ifo.save 1-7
- Moving a database 3-5
 - Syntax 3-6
- Multi-platform 1-3
- Object counter 1-7
- Objecteering/Document Template Editor 1-4
- Objecteering/Java 3-50
- Objecteering/The J Language 1-3, 2-3
- Objecteering/UML Enterprise Edition 1-5

- Objecteering/UML installation 1-4, 2-6
- Objecteering/UML Modeler 3-44
- Objecteering/UML Personal Edition 1-5
- Objecteering/UML Profile Builder 1-3, 1-4, 1-6, 3-37
- Objecteering/UML repository 1-3
- OMG model exchange standard 1-5
- Physical information 1-7
- Primitive classes
 - boolean 3-51
 - char 3-51
 - integer 3-51
 - real 3-51
 - string 3-51
 - undefined 3-51
- Project 2-3, 2-11
- Project counter 1-7
- Receiving a database 1-5
- Receiving the module 1-6
- Regenerating the module XML file 3-41
- Re-identification logic 3-33
- Re-identification operations 3-33
- Re-identifying a module 3-37
 - Syntax 3-38
- Re-identifying a project
 - Syntax 3-35
- Re-identifying a project by comparison
 - Syntax 3-40
- Re-identifying a UML modeling project 3-34
- Re-identifying a UML modeling project by comparison 3-39
- Re-identifying a UML profiling project 3-34
 - Syntax 3-35
- Re-identifying a UML profiling project by comparison 3-39
 - Syntax 3-40
- Renaming a database 3-9
 - Syntax 3-10
- Repair menu 2-4
 - Commands 2-12
 - Overview 2-11
- Repair menu commands
 - Re-identification of a module 3-37
 - Re-identification of a project 3-34
 - Re-identifying a UML modeling project by comparison 3-39
 - Re-identifying a UML profiling project 3-34
 - Re-identifying a UML profiling project by comparison 3-39
- Unlock a database 3-31
- Unlock a module database 3-32
- Repairing databases 1-3
 - Procedure 3-49
 - Syntax 3-49
- Repeated element identifiers 3-36
- Running checks on a database 2-14
- Running checks on a project 2-14
- Running checks on a site 2-14
- Semantic data 1-3
- Server 1-4
- Sharing modeling data 1-5
- Site 1-5, 1-6, 1-7, 2-3, 2-14, 3-14, 3-42
- Site database 1-4, 1-7, 2-6
 - Objecteering/UML installation 1-4
- Site identification 1-4
- Site identifier 1-7
- Site structure 1-4
- Site.ifo file 1-7, 3-8

- Site.ifo.save 1-7
- System menu 2-4
 - Commands 3-12, 3-45
- Transfer operations 3-33
- Transfer problems 2-11
- Transferring files 1-6
- Transferring modules between sites 1-6
- Typical use of the module re-identification operation 3-38
- Typical use of the project re-identification by comparison operation 3-40
- Typical use of the project re-identification operation 3-36
- UML modeling project 1-3, 1-7
- UML modeling projects 1-4, 3-11, 3-30
- UML profiles 1-7, 2-9
- UML profiling project 1-3, 1-7
- UML profiling projects 1-4, 1-5, 3-30
- Uninstalling a module from a database 3-27
 - Syntax 3-28
- Universal identification mechanism 1-4, 1-7, 2-15
- Unlocking a database 3-31, 3-49
 - Syntax 3-31
- Unlocking a module database 3-32
 - Syntax 3-32
- User databases 1-4
- User interface 2-3
- Verification menu 2-4
 - Commands 2-15
 - Managing identifiers 2-15
 - Overview 2-14
- Verification menu commands
 - Check a database 3-44
 - Check a database's identifiers 3-43
 - Check a project's identifiers 3-47
 - Check a site 3-46
 - Check a site's identifiers 3-45
- XMI exchange 1-5
 - Exchanging information between sites 1-5
- Zombie 1-7