Using the Rational Administrator

Version 2000.02.10

Rational®
the e-development company™

Using the Rational Administrator

Revised 04/2000

This manual prepared by:
Rational Software Corporation
20 Maguire Road
Lexington, MA 02421
U.S.A.

Phone:
800-433-5444
408-863-4000

E-mail: support@rational.com
Web: http://www.rational.com

P/N 800-023362-000

Contents

Preface

1 About the Rational Repository

2 Managing a Rational Repository

4 Managing a SQL Anywhere Database Server

A Customizing Toolbars

Glossary

Index

Preface

This manual describes how to use the Rational Administrator and Rational repositories. This manual is intended for system administrators, application developers, quality assurance managers, and quality assurance engineers.

Other Resources

- This product contains complete online Help. From the main toolbar, choose an option from the **Help** menu.

  For information about context-sensitive Help, see the following section.

- All manuals for this product are available online in PDF format. These manuals are on the *Rational Solutions for Windows* Online Documentation CD.

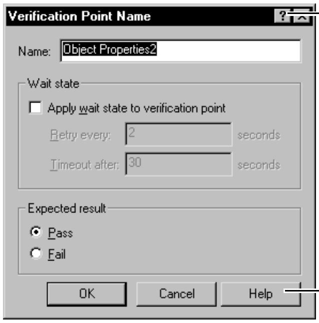
- For information about training opportunities, see the Rational University Web site: http://www.rational.com/university.

Using Help

This product contains context-sensitive Help for dialog boxes, menus, and toolbars.

Dialog Box Help

Most dialog box Help includes overviews and detailed item information.



Verification Point Name

Name: Object Properties2

Wait state

□ Apply wait state to verification point

Retry every: 2 seconds

Timeout after: 30 seconds

Expected result

● Pass

○ Fail

Click this and then click an item to see information about the item.

Click this to see an overview of the dialog box.

Menu Command Help

For menu command Help, highlight the command and press F1, or click the Help button on the toolbar and select the command. A brief description of the command also appears in the status bar.

Toolbar Button Help

For toolbar button Help, pause the pointer over the button. A yellow ToolTip appears below the button, and a brief description appears in the status bar. For more detailed information, click the Help button on the toolbar, and then select the button for which you want more information.

Contacting Rational Technical Publications

To send feedback about documentation for Rational products, please send e-mail to our technical publications department at techpubs@rational.com.

Contacting Rational Technical Support

If you have questions about installing, using, or maintaining this product, contact Rational Technical Support as follows:

Rational Technical Support

Location	Contact Information	Notes
North America	Telephone: 800-433-5444 408-863-4000 E-mail: support@rational.com	Please be prepared to supply the following information: – Your name, telephone number, and company name – Computer make and model – Operating system and version number – Product release number and serial number – Your Case ID number (if you are calling about a previously reported problem)
Europe	Telephone: +31 (0) 20 4546 200 E-mail: support@europe.rational.com	
Asia Pacific	Telephone: +61-2-9419-0111 E-mail: support@apac.rational.com	
World Wide Web	http://www.rational.com	Click the Technical Support link.

‣ ‣ ‣ CHAPTER 1

About the Rational Repository

This chapter introduces the Rational repository and describes the different types of datastores. It includes the following topics:

- What is a Rational repository?
- Parts of a Rational repository
- About Rational RequisitePro Databases
- About Rational Test database types
- About Rational ClearQuest databases
- Determining a database type

What Is a Rational Repository?

A **Rational repository** stores software testing and development information for your projects. All Rational Test components on your computer update and retrieve data from the same active repository.

NOTE: The type of data in a Rational repository depends on the Rational software that you have installed.

A Rational repository can contain several different datastores. A **datastore** contains one or more databases and other types of files. A Rational repository consists of the following types of datastores:

- Rational Test datastore – Stores application testing information such as test assets, logs, reports, builds, computers, users, and groups. For more information, see *About the Rational Test Datastore* on page 1-4.

- Rational ClearQuest datastore – Stores change-request information for software development, including enhancement requests, defect reports, and documentation modifications. Each ClearQuest datastore consists of one schema and one database. In ClearQuest, the term **schema** refers to all attributes associated with a database. This includes field definitions, field behaviors, the state transition table, actions, and forms. For more information, see *About the Rational ClearQuest Datastore* on page 1-5.
- Rational RequisitePro datastore – Stores product or system requirements, software and hardware requirements, user requirements, quality assurance procedures, and test plans. Each datastore consists of project documents and a dynamically linked database. For more information, see *About the Rational RequisitePro Datastore* on page 1-6.
- Rational Rose models – Stores visual models for business processes, software components, classes and objects, and distribution and deployment processes. For more information, see *About Rational Rose Models* on page 1-7.

Benefits of a Rational Repository

When you create a Rational repository, you can associate different Rational product datastores that are used for storing critical software development and testing information. In addition, within a Rational repository you can categorize this information by project.

Once you associate datastores with a Rational repository, you can link specific data from one datastore to another by using individual product features or by using the Rational Synchronizer. For more information about the Rational Synchronizer, see *About Rational Synchronizer Rules* on page 1-7.

NOTE: The Rational Synchronizer is available only in Rational Suite products.

The following table provides specific examples of how you can benefit from associating different datastores with one another in a Rational repository as you design, develop, and test software applications.

When you associate	Then you can
A ClearQuest datastore with a Rational Test datastore	Generate a defect using Rational LogViewer, and automatically store the defect in the correct project in a ClearQuest database.
A RequisitePro database with a project within a Rational Test datastore	Create a test requirement using Rational TestManager, and automatically see the test requirement in the associated RequisitePro database. Create a product or software requirement using RequisitePro and view the same requirement from Rational TestManager. A quality engineer, using TestManager, can then view up-to-date requirements for an application-under-test.
A RequisitePro database with a Rose model	You can also use the Rational Synchronizer to automatically create or update a requirement in RequisitePro based on a new or updated use case in a Rose model.

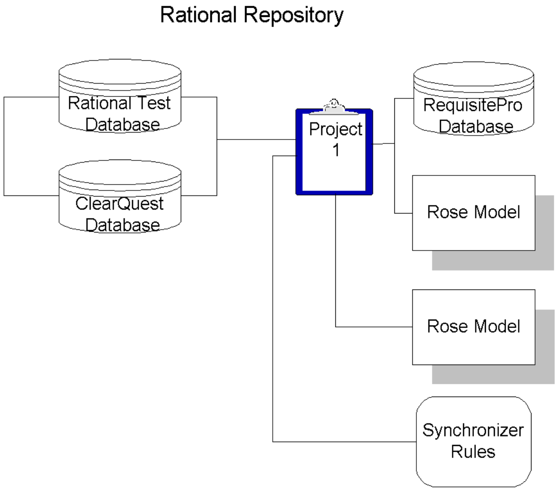
Parts of a Rational Repository

A Rational repository can consist of the following parts:

- One Rational Test datastore
- One ClearQuest datastore
- One or more projects, consisting of:
  - One RequisitePro datastore
  - One or more Rose models
  - Rational Synchronizer rules

NOTE: The Rational Synchronizer is available only in Rational Suite products.

The following figure shows the parts of a Rational repository:



About the Rational Test Datastore

The Rational Test datastore stores application testing information. Each datastore consists of a database and several file directories. In general, the database contains data entered into Rational Test applications. The non-database file directories contain files related to testing, such as verification points and script files.

The database contains information about:

Builds

Computers

Coverage reports

Datapools

Datatypes

Groups

Layouts

Listing reports

LoadTest report output

LoadTest reports

Log filters

Log folders

Logs

Projects

Queries

Schedules

Scripts

Sessions

Test analysis reports

Test documents

Users

About the Rational ClearQuest Datastore

A Rational ClearQuest datastore consists of one schema and one database. A ClearQuest database stores change-request records (such as defects) as well as reports, report formats, records, tables, and all other data entered by users. A ClearQuest schema defines the following information for each change-request record:

- Forms used to submit and modify a record
- Field types, legal values, and so on
- States a record can be in (optional)
- Actions used to modify or change the state of a record
- Hooks written in Visual Basic Script that further customize legal field values, dependent fields, and actions (optional)

NOTE: A full version of Rational ClearQuest comes with Rational Suite TestStudio. You need the full version of ClearQuest to add fields to a schema or to connect to a database to which fields have been added.

About Projects

Projects help you categorize your software development and testing information for easy tracking. The number of projects in a repository depends on the complexity of the software development project or application-under-test. The number of projects also depends on the number of ongoing testing efforts. For example, you can divide a large testing effort into several smaller projects within one repository, or you can create separate repositories for unrelated testing efforts.

A project usually consists of the following:

- Data in Rational Test datastores
- Data in RequisitePro datastores
- One or more Rose Models (optional)
- Rational Synchronizer rules

NOTE: The Rational Synchronizer is available only in Rational Suite products.

For instructions about how to add or delete a project, see *Managing Projects* on page 2-25.

Rational Test Datastore Information in Projects

Some Rational Test datastore information is stored either as project-specific or as project-independent information.

The following table shows which type of information is project-specific and which type is project-independent in Rational Test:

Type of information	Project-specific information	Project-independent information
Baseline and actual verification point data	x	
Computers		x
Customized terms		x
DAT files		x
LoadTest run schedule results	x	
LoadTest schedules	x	
LoadTest sessions	x	
Logs	x	
Queries		x
Reports and report layouts	x	
Scripts	x	
SQABasic library source code		x
Test requirements	x	
Users and groups		x

About the Rational RequisitePro Datastore

All requirements for a project are stored in a Rational RequisitePro datastore. Each datastore consists of requirements documents and a dynamically-linked RequisitePro database. A project document can store text, graphics, or other objects (such as OLE objects) that identify project needs and are meaningful to users. To open a requirements datastore, start RequisitePro and refer to the RequisitePro online Help.

Rational TestManager incorporates a baseline version of RequisitePro that lets you manage your test requirements. With this version of RequisitePro, you can build a test requirements hierarchy and associate scripts and schedules with the test requirements in the hierarchy.

With the full version of RequisitePro, you can customize the requirements database and use other advanced requirements management features such as traceability and requirements-attribute management. You can also manage all of your project's requirements — for example, user requirements, software specifications, and use cases.

NOTE: For information about baseline and full versions of RequisitePro that ship with Rational Test products, contact your sales representative.

About Rational Rose Models

A Rose model file stores a graphical representation of a design model. A model is a graphical representation of the structure and relationship of the parts of a system to the entire system. A model can represent business processes, software components, classes and objects, and distribution and deployment processes.

A Rose model contains files similar to the source language files for a program. A Rose model can consist of the following types of text files:

- Logical view package information
- Packages in the component diagram
- Process information
- Property files for various elements

About Rational Synchronizer Rules

You can share or link data between applications that store data in Rational Test, Rational RequisitePro, or Rational Rose datastores. However, when you update data in one of these products, you often want to update this data in other products. The Rational Synchronizer is a software tool that lets you:

- Ensure consistency of data across several products — a single change in one type of data can be updated in other types of data simultaneously.
- Jump-start work in one product with the data from another product — for example, you can create requirements using the TestManager component of Rational Robot or Rational RequisitePro, and then import one or more of these requirements to a Rose model.

- Conduct impact analysis — you can synchronize data in several different products to gain a deeper understanding of how changes in requirements or design changes impact your overall software development process.

You can start the Synchronizer from the Administrator. For more information about starting the Synchronizer from the Administrator, see *How the Synchronizer Works* on page 2-28.

NOTE: The Rational Synchronizer is available only in Rational Suite products.

About Rational RequisitePro Databases

All requirements for a project are stored in a Rational RequisitePro datastore. Each datastore consists of requirements documents and a dynamically-linked RequisitePro database.

When you create a project in the Administrator, you can use an existing RequisitePro database or you can create a new RequisitePro database. If you create a new RequisitePro database for a project using the Administrator, the Administrator creates a Microsoft Access type of database. If you use an existing RequisitePro database, the database can be any type of RequisitePro database that RequisitePro allows. For information about the types of RequisitePro databases you can create using RequisitePro, see the Rational RequisitePro Help.

For information about creating projects, see *Adding a Project* on page 2-25.

About Rational Test Database Types

A Rational Test database stores functional and performance testing information such as verification point data, queries, reports and report layouts, scripts, and information about users, computers, and groups. When you create a new repository, you can create two types of databases: Microsoft Access databases or Sybase SQL Anywhere Database Server databases. For a summary of which type of database software to install, see *Determining a Database Type* on page 1-10.

Microsoft Access – Microsoft Access is the default database type for repositories and installs automatically when you install your Rational software. We recommend using Microsoft Access for a database with seven or fewer concurrent users. However, with more than seven users, data access is slower than for a comparably sized Sybase SQL Anywhere Database Server database. This type of database is virtually maintenance-free. For example, you can back up the database simply by backing up the Rational repository directory. If you are accessing the database over a wide-area network, we recommend using Sybase SQL Anywhere for your database.

Sybase SQL Anywhere Database Server – Sybase SQL Anywhere Database Server software comes with your Rational software, but does not install automatically. We recommend using this type of database for a database with eight or more concurrent users. Data access for a SQL Anywhere Database Server database is faster than for a comparably sized Microsoft Access database. For information about installing this software, see your Rational installation manual.

For information about managing a Rational Test database with a SQL Anywhere database server, see Chapter 4, *Managing a SQL Anywhere Database Server*.

About Rational ClearQuest Databases

Rational ClearQuest comes with special defect forms and reports to help you manage your functional testing efforts. These form and reports are stored in a ClearQuest database. This database is different from the Rational Test database.

NOTE: The version of ClearQuest that comes with Rational TeamTest differs from standard ClearQuest in one way: you cannot add a field to a ClearQuest schema. To add fields to a schema, you must purchase standard Rational ClearQuest software.

A ClearQuest database consists of a **schema repository** and a **user** database. A ClearQuest schema repository stores all schemas and all data associated with existing schemas in a special repository. The schema repository does not store any user data. A ClearQuest user database contains all user data and a copy of the schema associated with the database. User data in a ClearQuest database includes change-request records such as defects, as well as reports, records or tables, and all data entered by users.

Choosing a Rational ClearQuest Database

When you create a ClearQuest user database, it can be either a Microsoft Access, Sybase SQL Anywhere Database Server, Microsoft SQL Server, or Oracle database. For a summary of which type of database software to install, see *Determining a Database Type* on page 1-10.

Rational ClearQuest Web lets you remotely enter change-request records using a Web-based interface.

NOTE: Rational ClearQuest Web does not support Microsoft Access databases.

Microsoft Access – Microsoft Access is the default database type for Rational repositories and installs automatically when you install your Rational software. We recommend using Microsoft Access for a database with seven or fewer concurrent users. However, with more than seven users, data access is slower than for a comparably sized Sybase SQL Anywhere Database Server database. This type of database is virtually maintenance-free. For example, you can back up the database simply by backing up the Rational repository directory.

Sybase SQL Anywhere Database Server – Sybase SQL Anywhere Database Server software comes with your Rational software, but does not install automatically. We recommend using this type of database for a database with eight or more concurrent users. Data access for a SQL Anywhere Database Server database is faster than for a comparably sized Microsoft Access database. For information about installing this software, see your Rational installation manual.

Microsoft SQL Server – Microsoft SQL Server does not come with your Rational software. You can use this software for ClearQuest databases; however, you must purchase the software and install it separately.

Oracle – Oracle does not come with your Rational software. You can use this software for ClearQuest databases; however, you must purchase the software and install it separately.

Determining a Database Type

The following table summarizes which type of database software to use for a Rational Test database.

If your Rational Test database	Then
Has seven or fewer concurrent users and is not accessed over a wide-area network	Use a Microsoft Access database. This is the default database and installs automatically when you install your Rational software.
Has eight or more concurrent users or is accessed over a wide-area network	Install the Sybase SQL Anywhere Database Server software that comes with your Rational software. For more information about installing Sybase SQL Anywhere Database Server software, see your Rational installation manual.

The following table summarizes which type of database software to use for a ClearQuest database.

If your Rational ClearQuest database	Then
Has seven or fewer concurrent users and is not accessed over a wide-are network	Use a Microsoft Access database. This is the default database and installs automatically when you install your Rational software.
Has eight or more concurrent users or is accessed over a wide-area network	Install the Sybase SQL Anywhere Database Server software that comes with your Rational software. For more information about installing Sybase SQL Anywhere Database Server software, see your Rational installation manual.
Has eight or more concurrent users and you want to use Microsoft SQL Server for your database	1. Purchase Microsoft SQL Server software. 2. Install Microsoft SQL Server to use with ClearQuest. For information about installing Microsoft SQL Server for ClearQuest, see the *Installing Rational ClearQuest* manual. 3. Configure Microsoft SQL Server to use with ClearQuest. For information about configuring Microsoft SQL Server for ClearQuest, see the *Administering Rational ClearQuest* manual. Both of these manuals are on the Rational online documentation CD.
Has eight or more concurrent users and you want to use Oracle for your database	1. Purchase Oracle software. 2. Install Oracle to use with ClearQuest. For information about installing Oracle for ClearQuest, see the *Installing Rational ClearQuest* manual. 3. Configure Oracle to use with ClearQuest. For information about configuring Oracle for ClearQuest, see the *Administering Rational ClearQuest* manual. Both of these manuals are on the Rational online documentation CD.

▸▸▸ CHAPTER 2

Managing a Rational Repository

This chapter explains how to use the Rational Administrator software to perform administrative tasks on a Rational repository. It includes the following topics:

- About the Rational Administrator
- Starting and exiting the Rational Administrator
- The Rational Administrator main window
- Rational repository privileges
- About the Rational License Key Administrator
- Repository tasks
- Working with a Rational repository
- Upgrading an SQA 6.x repository
- Creating a repository
- Managing projects
- Viewing the properties of a repository

About the Rational Administrator

The Rational Administrator centralizes the management of a Rational repository. For more information about the Rational repository, see Chapter 1, *About the Rational Repository*.

With the Rational Administrator, you can:

- Create and delete a repository.
- Connect to a repository.
- Configure a SQL Anywhere database server.
- Create and manage users, groups, and computers for a Rational Test database.

- Create and manage projects containing RequisitePro databases and Rose models.
- Manage security privileges for the entire Rational repository.
- Synchronize data among Rational Test datastores, Rational RequisitePro datastores, and Rational Rose using the Rational Synchronizer.

> **NOTE:** The Rational Synchronizer is available only in Rational Suite products.

- Change Rational Test and ClearQuest database types.
- Use the centralized Rational License Key Administrator.

Starting and Exiting the Rational Administrator

To start the Rational Administrator:

- Click **Start** → **Programs** → *Rational product name* → **Rational Administrator**.

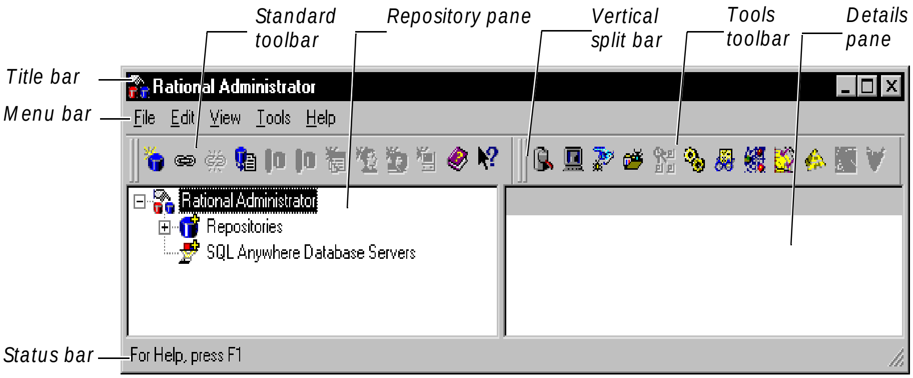
To exit the Administrator:

- Click **File** → **Exit**.

The Administrator disconnects from all repositories before exiting.

The Rational Administrator Main Window

After you start the Rational Administrator, the main window appears:



For Help, press F1

The Repository Pane

The left pane of the main window, the Repository pane, displays a list of repositories and SQL Anywhere database servers. When you connect to a repository, you can add or delete projects. You can also add or delete users, groups, and computers from a Rational Test database.

The Details Pane

When you select an item in the left pane of the main window, information about the selected item appears in the right pane, the Details pane.

The Status Bar

The status bar at the bottom of the main window displays a description of the currently selected menu item or operational messages.

Rational Repository Privileges

When you log into a Rational repository, your Rational Test database privileges determine the privileges you have for managing a repository. For more information about privileges, see *Rational Test Database Privileges* on page 3-1.

You can set the Rational Test database privileges for a user to:

- Perform administrator tasks for a repository — manage projects, users, groups, and computers, as well as delete repositories.
- Change the database type of a ClearQuest or Rational Test database. Attach an existing ClearQuest database to a Rational repository.
- Create, modify, and delete test assets, reports and layouts, builds, logs, and requirements.
- Customize test assets and builds.
- Create, modify, and delete logs and reports.

About the Rational License Key Administrator

The Rational License Key Administrator is a centralized license manager that lets you review and manage licenses for your Rational products.

Starting the Rational License Key Administrator

To start the Rational License Key Administrator from the Rational Administrator:

- Click **Start** → **Programs** → *Rational product name* → **Rational License Key Administrator**.

For more information about using the Rational License Key Administrator, see the Rational License Key Administrator Help.

Repository Tasks

The following table provides an overview of the typical tasks that you perform to manage a Rational repository.

Task	Description	See
Start the Administrator.		*Starting and Exiting the Rational Administrator* on page 2-2
Configure ClearQuest or attach an already existing ClearQuest database to a repository.	To use ClearQuest to store defects, you must configure ClearQuest before you create a ClearQuest database or attach an already existing ClearQuest database to a repository.	*Configuring ClearQuest* on page 2-12 and *Create or Attach a Rational ClearQuest Database* on page 2-17
Register an existing repository.	If you have an existing repository, you can add it to the list of repositories that appear in the main window of the Administrator.	*Registering and Unregistering an Existing Repository* on page 2-21
Create a repository.	You can create a Microsoft Access or SQL Anywhere Server database type for a Rational Test database. You can create a Microsoft Access, Sybase SQL Anywhere Server, Microsoft SQL Server, or Oracle database type for a ClearQuest database.	*Creating a Repository* on page 2-13

(Continued)

Task	Description	See
Connect to a repository.	You must connect to a repository before you can add a project, user, group, or computer.	*Connecting to a Repository* on page 2-20
Add one or more projects to a repository.	A project stores data from Rational Test programs. Other files stored in the projects related to testing are scripts, files, and schedules. Optionally, a project can store RequisitePro requirements, Rose models, and Synchronizer rules.	*Managing Projects* on page 2-25
Add one or more groups to a Rational Test database and assign privileges to a group.	Create a group or groups with privileges for a repository. When you add a user to a group, the user takes on the privileges of that group.	*Rational Test Database Privileges* on page 3-1
Add one or more users to a Rational Test database.	You must add a user to a repository before a user can use other Rational software. You must also make a user a member of a group to grant user privileges.	*Managing Users* on page 3-6 and *Rational Test Database Privileges* on page 3-1
Add one or more computers to a repository.		*Managing Computers* on page 3-9
Change a Rational Test database type.	You can change a Rational Test database from a Microsoft Access database to a SQL Anywhere Server database, or from a SQL Anywhere Server database to a Microsoft Access database.	*Changing a Rational Test Database Type* on page 2-22

The tasks in the following table are optional.

Task	Description	See
Change a ClearQuest database type.	You can change a ClearQuest database to one of the following database types: Microsoft Access, Sybase SQL Anywhere Server, Microsoft SQL Server, or Oracle.	*Changing a ClearQuest Database Type* on page 2-23
Attach an existing ClearQuest database.	You can attach an existing ClearQuest database to a Rational repository.	*Attaching an Existing ClearQuest Database* on page 2-24
Start the Synchronizer.	You can ensure consistency of data across Rational Test, Rational RequisitePro, and Rational Rose products by using Synchronizer rules.	*Starting the Synchronizer* on page 2-28 **NOTE:** The Rational Synchronizer is available only in Rational Suite products.

Working with a Rational Repository

Before working with a Rational application for the first time, you must create and connect to a Rational repository.

When you create a new repository, you create a Rational Test database as either a Microsoft Access database or a SQL Anywhere Server database.

If you create a ClearQuest database and attach it to a repository, you create it as a Microsoft Access, a Sybase SQL Anywhere Server, a Microsoft SQL Server, or an Oracle database.

When you manage a repository, you can:

- Create or delete a repository. To delete a repository, you must have privileges. For more information, *Rational Test Database Privileges* on page 3-1.
- Connect or disconnect from a repository.
- Register an existing repository so it appears in the current list of repositories in the left pane of the main widow.
- Change database types for Rational Test or ClearQuest databases.
- Attach a ClearQuest database to a repository.

Upgrading an SQA 6.x Repository

You can use data from an existing SQA Suite 6.x repository in a Rational repository. When you create a new Rational repository, the Create Repository wizard gives you the option of initializing a repository with data from an existing SQA Suite 6.x repository.

For information about creating a Rational repository with SQA Suite 6.x data, see *Creating a Repository* on page 2-13.

When to Convert SQA Suite 6.x Repository Data

We recommend that you convert SQA Suite 6.x repository data to Rational repository data after you finish a project.

For information about creating a Rational repository with SQA Suite 6.x data, see *Creating a Repository* on page 2-13.

The following table lists the data that is converted from a SQA Suite 6.x repository to a Rational repository:

Type of SQA Suite 6.x data	**Converted?**
Custom reports and graphs	No
Defects (For more information about defects, see *Details About Converting Defects* on page 2-8.)	Yes
E-mail rules	No
Filters for defects or test procedures	No
Groups (Privileges are not converted.)	Yes
List reports	No
LoadTest schedules	No
Projects	Yes
Requirements	Yes
Test cases (Called *verification points* in a Rational repository.)	Yes
Test logs	No

(Continued)

Type of SQA Suite 6.x data	Converted?
Test plans	Yes
Test procedures (Called *scripts* in Rational repository.)	Yes
Users (Privileges are not converted.)	Yes
Virtual user test procedures (Called *virtual user scripts* in a Rational repository.)	No

Conversion Details

When you create a Rational repository with SQA Suite 6.x data, the following conversion takes place:

- A verification point ID or script file name with a period (.) in the name is prefixed with _RENAMED_.
- A period (.) in a script file name or verification point ID converts to an underscore (_).

For example, a script named wn95.tst converts to _RENAMED_wn95_tst.
A verification point ID named alpha.b2 converts to _RENAMED_alpha_b2.

Details About Converting Defects

Rational testing products incorporate the change-request management technology of Rational ClearQuest to track defects. For your convenience, Rational ClearQuest includes a specially designed defect form, the TeamTest defect form, similar to the SQA 6.x Manager defect form. For more information about using the TeamTest defect form, see the *Using the Rational Robot* manual.

The following table lists each field of an SQA 6.x defect and the comparable Rational TeamTest defect field. The table also describes the conversion of each field.

NOTE: The word *List* in the Description column of this table indicates that if you customize a list in SQA 6.x, it converts to a list in the TeamTest defect form.

SQA 6.x Defect Field	TeamTest Defect Field	Description
ID	New ID generated	The SQA 6.x ID does not appear on the defect form, but you can use the query feature of ClearQuest to find it. For more information, see the Rational ClearQuest Help.
Description	Headline/ Description	Converts the first 125 characters or up to the first carriage return and places this text in the Headline field. The entire SQA 6.x Description field also appears in the Description field of the TeamTest defect form.
Priority	Priority	List
Severity	Severity	List
Occurrences	(Not converted)	
Keywords	Keywords	List
Symptoms	Symptoms	List
Build Found	Notes	In SQA 6.x, a Build Found field is a text field. The text data from the Build Found field is stored in Notes. In Rational 7.x, **Builds** are objects that contain log folders and logs. A Build object is not created for each SQA 6.x Build Found field.
Build Fixed	Notes	In SQA 6.x, a Build Fixed field is a text field. The text data in the Build Fixed field is stored in Notes. In Rational 7.x, Builds are objects that contain log folders and logs. A Build object is not created for each SQA 6.x Build Fixed field.
Proc	Script	
Case	Verification Point	
Cycle	(Not converted)	

(Continued)

SQA 6.x Defect Field	TeamTest Defect Field	Description
Reported By	Reported By Contact	
Reported By Company	Reported By Company	
Hardware	Hardware	List
Operating System	Operating System	List
Other	Other Environment	
Test Station	(Not converted)	In SQA 6.x, a Test Station field is a text field. In Rational 7.x the Computers field replace the Test Station field and is an object. A Computer object is not created for each SQA 6.x Test Station field.
Log	(Not converted)	
Custom 1	Custom 1	List If you customized the label for this field in SQA 6.x, the label is not converted. Use the ClearQuest Designer to customize this label.
Custom 2	Custom 2	List If you customized the label for this field in SQA 6.x, the label is not converted. Use the ClearQuest Designer to customize this label.
Custom 3	Custom 3	If you customized the label for this field in SQA 6.x, the label is not converted. Use the ClearQuest Designer to customize this label.
Attachment	Attachments	You can attach more than one file after you convert to a 7.x Rational repository.
Requirement	Requirement	

(Continued)

SQA 6.x Defect Field	TeamTest Defect Field	Description
Status History	History	
Resolution	Resolution	List
Resolution Description	Resolution Note	
Modified Software	(Not converted)	

Troubleshooting

The convert.txt file in the converted project directory contains the conversion status of each verification point and script. Use this file as a diagnostic tool if you have any problems with your data after conversion.

Before Creating a Repository

Before you create a repository, you must perform the following steps:

1. Decide what type of database to use. For information about database types, see *About Rational Test Database Types* on page 1-8 and *Choosing a Rational ClearQuest Database* on page 1-9. For a summary of database types, see *Determining a Database Type* on page 1-10.
2. We recommend that you create a shared directory for each shared database or repository. For more information, see *Create a Shared Directory* on page 2-13.
3. If you use SQL Anywhere for a Rational Test database or a ClearQuest database, you must do the following before you create a repository:

   a. Install the Sybase SQL Anywhere software.

      For information about installing Sybase SQL Anywhere software, see the installation guide for your Rational product.

   b. Create a SQL Anywhere database server.

      For information about creating a SQL Anywhere database server, see *Creating a SQL Anywhere Database Server* on page 4-2.

4. If your Rational software comes with ClearQuest, you must create a ClearQuest database to store defects for your functional testing efforts. This database is separate from the Rational Test database. If you use Oracle or Microsoft SQL Server for a ClearQuest database, do the following:

   a. Purchase, install, and configure Oracle or Microsoft SQL Server database software.

   > **NOTE:** If you use either a Microsoft SQL Server database or an Oracle database with ClearQuest, you must install and configure the database specifically for ClearQuest or you will not be able to create a repository. For information about installing Oracle or Microsoft SQL Server for ClearQuest, see the *Installing Rational ClearQuest* manual. For information about configuring Oracle or Microsoft SQL Server for ClearQuest, see the *Administering Rational ClearQuest* manual. Both of these manuals are on the Rational online documentation CD.

   b. Configure ClearQuest by creating a new schema repository or by connecting to an existing schema repository. For information, see the next section, *Configuring ClearQuest*.

Configuring ClearQuest

Before you create a repository with a ClearQuest database, or attach an existing ClearQuest database to a repository, you must configure ClearQuest by creating a new schema repository or by connecting to an existing schema repository. You only need to do this once for all repositories. If you attach an existing ClearQuest database to a Rational repository that has an existing schema repository, you do not have to create a new schema repository. For more information about ClearQuest schemas, see *About Rational ClearQuest Databases* on page 1-9 and the ClearQuest Help.

For information about attaching an existing ClearQuest database when you create a repository, see *Creating a Repository* on page 2-13. For information about attaching an existing ClearQuest database to an existing repository, see *Attaching an Existing ClearQuest Database* on page 2-24.

To configure ClearQuest:

1. Click **Tools → Rational ClearQuest Maintenance Tool.**

2. Follow the directions in the ClearQuest Setup Wizard to create a new schema repository or to connect to an existing schema repository.

3. Optionally, create a sample database. ClearQuest provides a sample database containing sample records that you can use to familiarize yourself with ClearQuest.

For more information about the ClearQuest Setup Wizard, click **Help** on the wizard. The wizard creates two default ClearQuest accounts. You need the user ID and password to log into ClearQuest. The user IDs and passwords are as follows:

- Default administrator account – The user ID is **admin**, with no password.
- Default user account – The user ID is **user**, with the password, **password**.

Create a Shared Directory

If you want to share a database or repository, we recommend that you create a shared directory, as follows:

1. Create a directory in the Windows Explorer.
2. Right-click the directory, and then click **Sharing**.
3. Click **Shared As**.
4. Type the name of the shared directory in the **Share Name** box.
5. Click **OK**.

Creating a Repository

Before working with a Rational Test application for the first time, you use the Administrator to create a repository.

NOTE: You need at least 10 MB of disk space to create a repository.

There are three steps that are required to create a Rational repository:

1. Specify a path and database type for a Rational repository.
2. Optionally, initialize a Rational Test database with data from an existing SQA Suite 6.x or Rational repository.

   NOTE: You can only initialize a Rational Test database with global repository data, such as users, groups, permissions, and computers from an existing Rational repository. Initializing does not copy project information.

3. Optionally, create or attach a ClearQuest database to the repository.

Specify a Path and Database Type

To create a SQL Anywhere Server type of repository, you must create a repository on a server with SQL Anywhere Server software installed.

To specify a path and database type for a Rational repository:

1. Click **Start** → **Programs** → *Rational product name* → **Rational Administrator**.
2. Click **File** → **Create Repository**.
3. Take one of the following actions:
   - **a.** To share a repository, we recommend that you create the repository in a shared directory and use the Uniform Naming Convention (UNC) for the directory name. (For more information about creating a shared directory, see *Create a Shared Directory* on page 2-13.)

     ```
     \\machinename\sharename\reponame
     ```

     where *machinename* is the network name of the repository computer, *sharename* is the name of a networked or shared directory, and *reponame* is the name of the new repository.

     For example:
     `\\dell300\defects\repo1`

   - **b.** If you do not need to share a repository, type the drive and pathname for the repository, or click **Browse** to select the path for the repository.

     For example:
     `C:\defects\repo1`
4. Click **Next** to continue.

5. Take one of the following actions:

If you want to create a Rational Test database using	Then
Microsoft Access software	**a.** In the **Database type** list, select **Microsoft Access**. **b.** Click **Next** to continue.

Sybase SQL Anywhere software	**NOTE:** You must install Sybase SQL Anywhere Server software and create a SQL Anywhere database server before you create this type of repository. For information about installing SQL Server software, see the installation guide for your Rational product. For information about creating a SQL Anywhere database server, see *Creating a SQL Anywhere Database Server* on page 4-2.
	a. In the **Database type** list, select **Sybase SQL Anywhere**.
	b. In the **SQL Anywhere database server** list, select the name of a SQL Anywhere database server.
	c. Type the database name in the **Database name** box.
	d. Type the drive and pathname for new database files in the **Location of database files** box, or click **Browse** to select the path.
	NOTE: We recommend that you create a shared directory for database files. For more information about creating a shared directory, see *Create a Shared Directory* on page 2-13.
	We also recommend that you use the Uniform Naming Convention (UNC) for the directory name:
	`\\machinename\sharename`
	where *machinename* is the Network Name of the repository computer, and *sharename* is the name of a networked or shared directory.
	For example:
	`\\dell300\defects\`
	e. Click **Next** to continue.

6. Proceed to the next section, *Initialize a Rational Test Database with Data.*

Initialize a Rational Test Database with Data

When creating a new Rational repository, you can use data from a SQA Suite 6.x repository or from another Rational repository. We recommend that you do this after you finish a project.

NOTE: You can only initialize a Rational Test database with global repository data, such as users, groups, permissions and computers from an existing Rational repository. Initializing does not copy project information.

1. In the Create Repository wizard, take one of the actions listed in the following table.

If you	Then
Do not want to initialize a Rational Test database with data from an existing Rational repository.	**a.** Click **Do not initialize the new repository using data from an existing repository.** **b.** Click **Next** to continue, and then proceed to step 2.
Want to initialize a Rational Test database with data from an existing Rational repository.	**a.** Click **Initialize the new repository using data from an existing Rational repository.** **b.** Type the drive and pathname of an existing repository in the box, or select a repository from a list of registered Rational repositories. Alternatively, click **Browse** to select the path for an existing repository. **c.** Click **Next** to continue, and then proceed to step 2.
Want to initialize the Rational Test database with data from an existing SQA 6.x repository.	**a.** Click **Initialize the new repository using data from an existing SQA 6.x repository.** **b.** Type the drive and pathname of the existing SQA 6.x repository in the box, or select a repository from a list of registered Rational repositories. Alternatively, click **Browse** to select the path for an existing repository. **c.** Click **Next** to continue, and then proceed to step 2.

2. Take one of the following actions:

If you	**Then**
Do not want to create or attach a ClearQuest database to a new repository.	**a.** Clear the **Create or Attach ClearQuest Database** check box. **b.** Click **Next** to continue. A summary page confirms the type and location of your new repository. **c.** Click **Finish** to create the new repository.
Want to create or attach a ClearQuest database to a new repository.	Proceed to the next section, *Create or Attach a Rational ClearQuest Database*.

Create or Attach a Rational ClearQuest Database

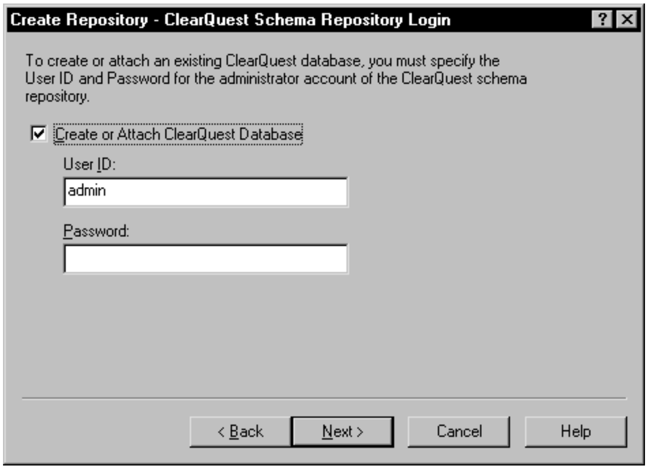
If your Rational software comes with ClearQuest, you must create a ClearQuest database to store defects for your functional testing efforts. This database is separate from the Rational Test database.

If you create a Sybase SQL Anywhere Server type of database, you must install Sybase SQL Anywhere Server software and create a SQL Anywhere database server before you create this type of database. For information about installing SQL Anywhere Server software, see the installation guide for your Rational product. For information about creating a SQL Anywhere database server, see *Creating a SQL Anywhere Database Server* on page 4-2.

If you use either a Microsoft SQL Server database or an Oracle database with ClearQuest, you must install and configure the database specifically for ClearQuest or you will not be able to create a repository. For information about installing Oracle or Microsoft SQL Server for ClearQuest, see the *Installing Rational ClearQuest* manual. For information about configuring Oracle or Microsoft SQL Server for ClearQuest, see the *Administering Rational ClearQuest* manual. Both of these manuals are on the Rational online documentation CD.

To create or attach a Rational ClearQuest database to the Rational repository that you are creating:

1. In the Create Repository wizard, select the **Create or Attach ClearQuest Database** check box.



Create Repository - ClearQuest Schema Repository Login

To create or attach an existing ClearQuest database, you must specify the User ID and Password for the administrator account of the ClearQuest schema repository.

☑ Create or Attach ClearQuest Database

User ID: admin

Password:

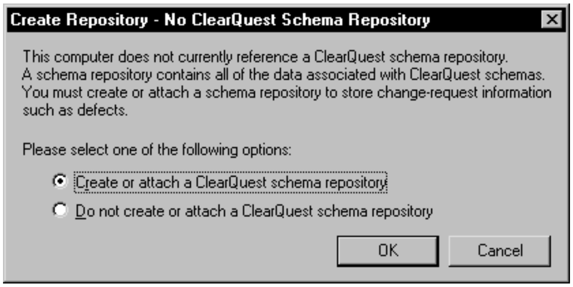
2. Log into the administrator account of the ClearQuest schema repository.

> **NOTE:** The user ID for the default administrator account is **admin**, with no password.

   a. Type the **User ID**.

   b. Type the **Password**.

3. Click **Next** to continue.
4. If you did not configure ClearQuest, the following dialog box appears:



Create Repository - No ClearQuest Schema Repository

This computer does not currently reference a ClearQuest schema repository. A schema repository contains all of the data associated with ClearQuest schemas. You must create or attach a schema repository to store change-request information such as defects.

Please select one of the following options:

● Create or attach a ClearQuest schema repository

○ Do not create or attach a ClearQuest schema repository

a. Click one of the following:

Create or attach a ClearQuest schema repository

Do not create or attach a ClearQuest schema repository

b. Click **OK**.

c. If you click **Create or attach a ClearQuest schema repository**, follow the directions in the ClearQuest Setup Wizard. For more information about the ClearQuest Setup Wizard, click **Help** on the wizard. The wizard creates two default ClearQuest accounts, with the following user IDs and passwords:

- Administrator account – The User ID is **admin**, with no password.
- User account – The User ID is **user**, with the password, **password**.

5. Take one of the following actions:

If you want to	Then
Attach an existing ClearQuest database to a new repository.	**a.** Click **Attach an existing ClearQuest database**. **b.** Select an existing ClearQuest database from the **Database Name** column. **c.** Click **Next**. A summary page confirms the type and location of your new repository. **d.** Click **Finish** to create the new repository.
Create a new ClearQuest database.	**a.** Click **Create a new ClearQuest database**. **b.** Type the database name of the new database. **c.** Type a description of the new database. **d.** Click **Next**. **e.** Select an item from the **Database type** box: **Microsoft Access** **SQL Anywhere** **SQL Server** **Oracle8** **f.** Fill in the properties of the ClearQuest database type you chose, and then click **Next.** **g.** Type the length of time, in minutes, for ClearQuest to maintain an idle-connection to a ClearQuest database in the **Timeout: Minutes** box. **h.** Type the length of time, in minutes, for ClearQuest to poll the database before disconnecting in the **Poll Interval: Minutes** box.

(Continued)

If you want to	Then
Create a new ClearQuest database.	**NOTE:** To prevent ClearQuest from disconnecting from a database, type 0 in the **Timeout: Minutes box** and 0 in the **Poll Interval: Minute**s boxes. **i.** Click **Next**. **j.** Select an existing ClearQuest schema from the **Schema Name** column. For Rational testing software, (Rational Suite Enterprise, Rational Suite PerformanceStudio, Rational Suite TestStudio, or Rational TeamTest) we recommend that you select the **TestStudio** schema. This is the Rational schema that is designed to work with Rational testing software. **NOTE:** If you are using Rational TeamTest, you must select the TestStudio schema or ClearQuest will not work. The version of ClearQuest that comes with TeamTest differs from standard Rational ClearQuest and does not allow you to add a field to a schema or to use any schema other than the TestStudio schema. To add fields to a schema or to use a schema other than the TestStudio schema, you must purchase standard Rational ClearQuest software or a Rational Suite product that comes with standard Rational ClearQuest. **k.** Click **Next**. A summary page confirms the type and location of your new repository. **l.** Click **Finish** to create the new repository.

Connecting to a Repository

You must connect to a repository if you want to add a project to a repository or add a user, group, or computer to a Rational Test database. When you connect to a repository, you gain access to the entire Rational repository.

To connect to a repository:

1. Right-click the repository to which you want to connect, and then click **Connect**.
2. Click **OK**.

3. Take one of the following actions:

   a. If you are logging into a repository for the first time:

      i. Type **Admin** for the **User ID**. The User ID is not case-sensitive.

      ii. Leave password blank.

> **NOTE:** When you connect to a repository for the first time, you should define a password for the Admin user to prevent unauthorized access. The Admin user always has full user access privileges and cannot be deleted from a repository.

      iii. Change the password of the Admin user to prevent other users from using the Administrators group privileges. For more information about changing a password, see *Changing a Password* on page 3-9.

   b. If you are not logging in for the first time:

      i. Type your User ID. The User ID is not case-sensitive.

      ii. Type your password. The password is case-sensitive.

   c. Click **OK**.

Disconnecting from a Repository

To disconnect from a repository:

▶ Click **File** → **Disconnect.**

If you are connected to more than one repository, a dialog box appears with a list of all connected repositories. Select the repository that you want to disconnect from, and then click **OK**.

Registering and Unregistering an Existing Repository

If you have an existing repository, you can add it to the list of repositories that appear in the main window of the Administrator by registering it.

When you unregister a repository, the Administrator removes the repository from the list of repositories in the main window of the Administrator. It does not delete the repository.

To register an existing repository:

1. Click **File** → **Register Existing Repository**.
2. Type the drive and pathname to an existing repository, or click **Browse** to select the path to an existing repository.
3. Click **OK**.

To unregister an existing repository:

▸ Right-click the repository you want to unregister, and then click **Unregister**.

Deleting a Repository

When you delete a repository, the Rational Administrator deletes all parts of the repository created by the Administrator. If you attached a ClearQuest database or a Rose model not created by the Administrator, they are not deleted when you delete the repository. Only the association with the repository is deleted.

To delete a repository:

1. Make sure that no one else is using the repository that you want to delete.

> **NOTE:** You must have privileges to delete a repository. For information about privileges, see *Rational Test Database Privileges* on page 3-1.

2. Select the repository you want to delete from the left pane of the main window.
3. Click **Edit** → **Delete** to delete the repository.
4. If you are not connected to the repository you want to delete, you are prompted to log in.
   - **a.** Type your User ID. The User ID is not case-sensitive.
   - **b.** Type your password. The password is case-sensitive.
5. The Administrator prompts you to confirm the deletion. Click **Yes**.

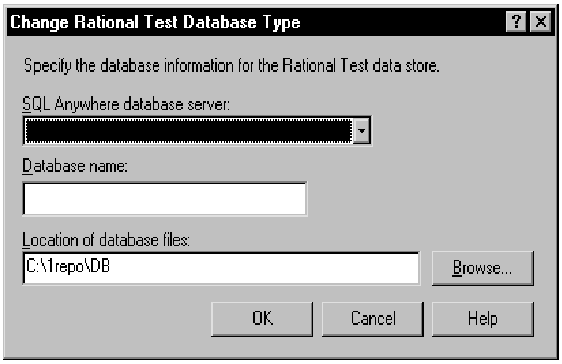
Changing a Rational Test Database Type

You must connect to a repository before you change a Rational Test database type. For more information about connecting to a repository, see *Connecting to a Repository* on page 2-20.

To change a Rational Test database type:

1. Click **File** → **Change Rational Test Database Type**.

2. Depending on the type of database, take one of the following actions:
   - If your Rational Test database is a SQL Anywhere database, the Administrator prompts you to confirm the change to Microsoft Access. Click **OK**.
   - If your Rational Test database is a Microsoft Access database, the following dialog box appears:



Change Rational Test Database Type
?
Specify the database information for the Rational Test data store.
SQL Anywhere database server:
Database name:
Location of database files:
Browse...
OK
Cancel
Help

Location of database files: C:\1repo\DB

   a. Fill in the information for the new type of database.
   b. Click **OK**.

Changing a ClearQuest Database Type

NOTE: If you change your ClearQuest database to a Sybase SQL Anywhere database, you must install Sybase SQL Anywhere database server software and create a SQL Anywhere database server before you change your database type. For information about installing SQL Server software, see the installation guide for your Rational product. If you use either a Microsoft SQL Server database or an Oracle database with ClearQuest, you must install and configure the database specifically for ClearQuest or you will not be able to change the database type. For information about installing Oracle or Microsoft SQL Server for ClearQuest, see the Installing Rational ClearQuest manual. For information about configuring Oracle or Microsoft SQL Server for ClearQuest, see the *Administering Rational ClearQuest* manual. Both of these manuals are on the Rational online documentation CD.

To change a ClearQuest database type:

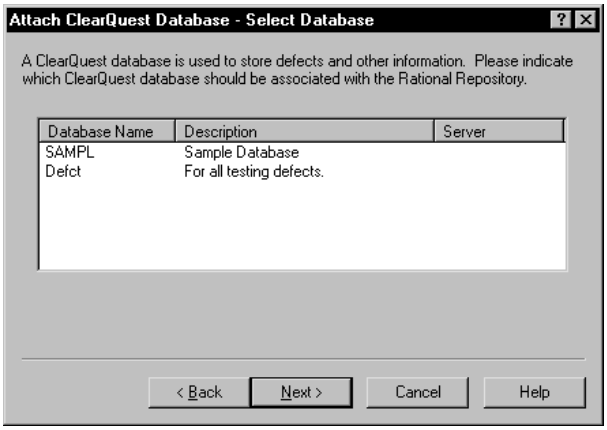
1. Connect to the repository that you want to change.
2. Click **File** → **Change ClearQuest Database Type.**

3. Type the **User ID** and the **Password** of the administrator account of the ClearQuest schema repository, and then click **Next**.
4. Select the new **Database type**.
5. Type the information required by your new type of database.
6. Click **Next**.
7. Click **Finish**.

Attaching an Existing ClearQuest Database

To attach a ClearQuest database to a repository:

1. Click **File** → **Attach ClearQuest Database.**
2. Type the **User ID** and the **Password** of the administrator account of the ClearQuest schema repository, and then click **Next**.
3. Select the ClearQuest database from the **Database Name** column that you want to associate with the Rational repository, and then click **Next**.



A ClearQuest database is used to store defects and other information. Please indicate which ClearQuest database should be associated with the Rational Repository.

Database Name	Description	Server
SAMPL	Sample Database	
Defct	For all testing defects.	

4. Click **Finish**.

Managing Projects

When you create a repository, the Administrator creates a default project for you. You can add projects to a repository. The number of projects in a repository depends on the complexity of the work you are doing or on the number of ongoing unrelated efforts. For example, you could divide a large testing application into several smaller projects, or you could define a separate project for each unrelated testing effort.

You cannot modify the name of a project once you create it.

Adding a Project

When you add a project to a repository, you can associate a project with one or more Rose models. See step 6.

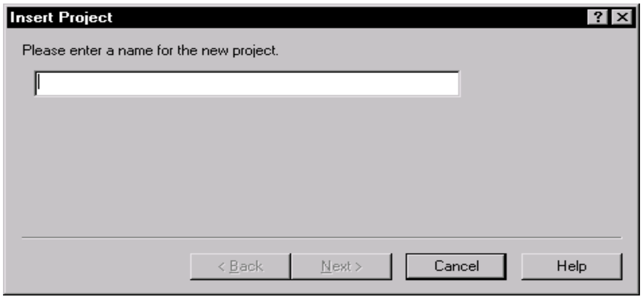
When you create a project in the Administrator, you can use an existing RequisitePro database or you can create a new RequisitePro database. If you create a new RequisitePro database for a project, the Administrator creates a Microsoft Access type of database. If you use an existing RequisitePro database, the database can be any type of RequisitePro database that RequisitePro allows. For information about the types of RequisitePro databases that you can create using RequisitePro, see the Rational RequisitePro Help.

NOTE: You must connect to a repository before you can add a project to a repository. For more information about connecting to a repository, see *Connecting to a Repository* on page 2-20. You must have privileges to add a new project to a repository. For more information about privileges, *Rational Test Database Privileges* on page 3-1.

To add a project to a repository:

1. Click **Insert** → **Project**.
2. If you are connected to more than one repository, a dialog box appears with a list of all connected repositories. Select the repository that you want to add a project to, and then click **OK**.

3. Type a string of up to 40 alphanumeric, double-byte, or extended ASCII characters in the box, and then click **Next**.



Insert Project

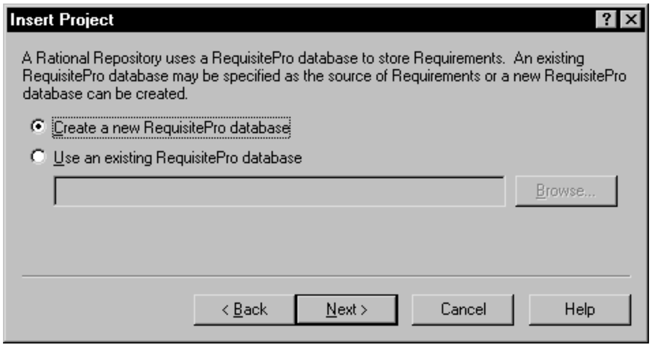
Please enter a name for the new project.

NOTE: If you have a ClearQuest project with the same name, the Administrator prompts you to confirm that you want to use the same project name in the Rational repository.

4. Take one of the following actions:

   – Click **Create a new RequisitePro database**.

   – Click **Use an existing RequisitePro database** to use an existing database. Then type the drive and pathname for an existing Requisite database, or click **Browse** to select the path for an existing database.



Insert Project

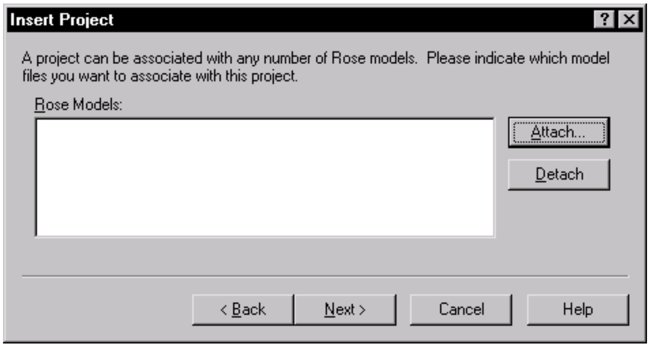
A Rational Repository uses a RequisitePro database to store Requirements. An existing RequisitePro database may be specified as the source of Requirements or a new RequisitePro database can be created.

● Create a new RequisitePro database

○ Use an existing RequisitePro database

5. Click **Next**.

6. Optionally, click **Attach** to associate a project with one or more Rose models. To change your selection, click **Detach**.



Insert Project

A project can be associated with any number of Rose models. Please indicate which model files you want to associate with this project.

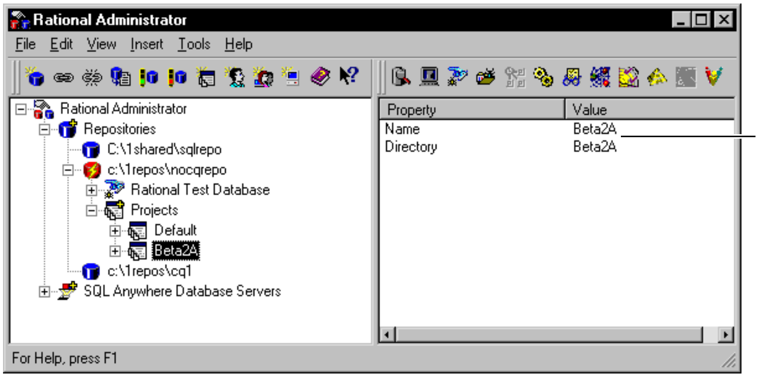
Rose Models:

7. Click **Next** to continue.

   A summary page confirms the selections you have made for a new project.

8. Click **Finish** to create a new project.

   The project name appears in the left pane of the main window under **Projects**. The project name and project directory appear in the right pane of the main window.



Rational Administrator

Property	Value
Name	Beta2A
Directory	Beta2A

Project name and directory

How the Synchronizer Works

You use the **Rational Synchronizer** to ensure consistency of data across Rational Test, Rational RequisitePro, and Rational Rose datastores. The Rational Synchronizer uses a rules-based engine to operate on data or items in one or more products. You select which products and which data the Synchronizer affects. You can preview how your selections will affect data before you use the Synchronizer. For more information about Synchronizer rules and where the rules are stored, see *About Rational Synchronizer Rules* on page 1-7.

NOTE: The Rational Synchronizer is available only in Rational Suite products.

A set of default rules ship with the Rational Synchronizer. The Administrator stores the default Synchronizer rules in each project in a Rational repository. You can use the default rules, or you can customize them by adding your own rules or editing the existing ones. For more information about using, adding, or editing rules, see the Rational Synchronizer Help.

Starting the Synchronizer

The Rational Synchronizer is available only in Rational Suite products.

To start the Synchronizer:

1. Connect to the repository that contains the project you want to synchronize.
2. Select the project you want to synchronize.
3. Click **Tools** → **Rational Synchronizer**.

For more information about how to use the Synchronizer or how to create your own rules, see the Rational Synchronizer Help.

Deleting a Project

Deleting a project removes all project data for that project from the repository. When you delete a project, the Administrator deletes everything in the project folder — the RequisitePro database, any Rose models created within a project folder, and all Rational Synchronizer rules in the folder.

To delete a project:

1. Double-click **Projects** in the left pane of the main window to display all projects.
2. Select the project you want to delete.

3. Click **Edit** → **Delete**.

   > **NOTE:** You must have privileges to delete a project from a repository. For more information about privileges, see *Rational Test Database Privileges* on page 3-1.

   The Administrator prompts you to confirm the deletion.

4. Click **Yes** to delete the project, including all project data.

Viewing the Properties of a Repository

You can view the properties of the following parts of a repository:

- Rational Test database
- ClearQuest database
- Projects
- RequisitePro database
- Rose models
- SQL Anywhere database servers

To view the properties of a repository:

- Right-click an item in the left pane of the main window, and then click **Properties**.

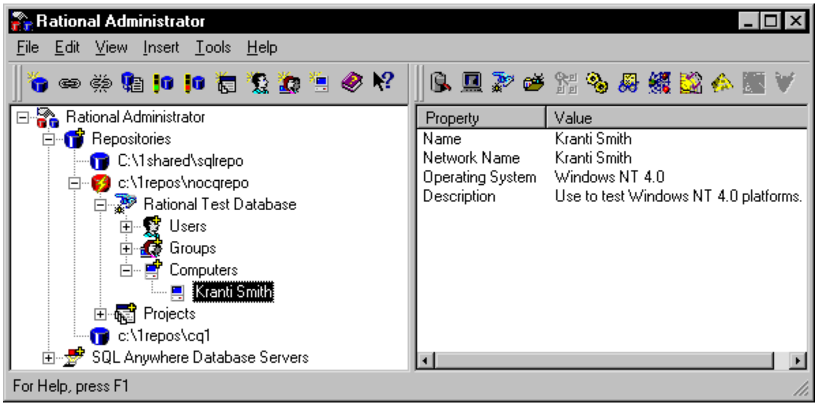
  See the Rational Administrator Help for a description of the properties of each of these items in the repository.

To view the properties of an item in the right pane of the Administrator:

- Under **Repositories** in the left pane of the main window, click the folder that you want to open, and then click the item that you want to view.

  > **NOTE:** You must connect to a repository before you can view a project, user, group, or computer in a repository. For more information about connecting to a repository, see *Connecting to a Repository* on page 2-20.

The properties of the object that you select appear in the right pane of the main window. For more information about displaying property pages, see *property pages* in the Rational Administrator Help.



Property	Value
Name	Kranti Smith
Network Name	Kranti Smith
Operating System	Windows NT 4.0
Description	Use to test Windows NT 4.0 platforms.

▸▸▸ CHAPTER 3

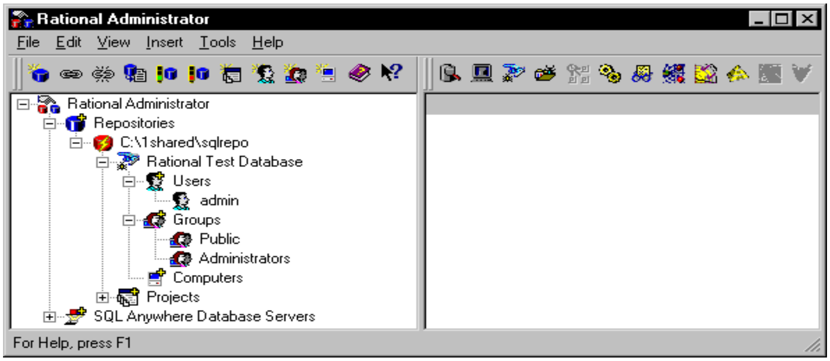
Managing a Rational Test Database

This chapter explains how to manage a Rational Test database. It includes the following topics:

- Rational Test database privileges
- Managing users
- Managing computers

Rational Test Database Privileges

When you create a repository, the Administrator creates a Rational Test database. The Rational Test database contains the Users, Groups, and Computers folders. The Users folder contains an **admin** user. The Groups folder contains a **Public** group and an **Administrators** group. These default objects appear in the left pane of the main window.



When you log into a Rational repository, your Rational Test database privileges determine the privileges that you have for managing a repository.

You determine the security of a repository by assigning privileges to a group. Groups allow you to set privileges for each user. When you add a user to a group, the user takes on the privileges of that group.

You can set the Rational Test database privileges for a user to:

- Create, modify, and delete test assets, reports and layouts, builds, logs, and requirements.
- Customize test assets and builds.
- Create, modify, and delete logs and reports.
- Perform administrator tasks for a repository: manage projects, users, groups, and computers, as well as delete repositories.
- Change the database type of a ClearQuest or Rational Test database. Attach an existing ClearQuest database to a Rational repository.

Overview of the Administrators Group

Each repository that you create comes with an Administrators group that includes a default user, *Admin*. The Admin user is a member of the Administrators group and takes on all privileges of the Administrators group.

NOTE: The Administrators group in every repository has privileges to add, modify, or delete a group. You must be a member of this group to manage groups; otherwise, ask your administrator for this privilege.

For example, as administrator of a repository, you can create a group with privileges that allow the group to create and modify test assets, builds, logs, and reports in a repository, but deny the group the privilege of deleting a repository.

You can also make a group the default group. When you add a user to a Rational Test database, and this option is set, every user that you create takes on the privileges of that default group.

A member of the Administrators group can:

- Create, modify, and delete test assets.
- Create, modify and delete reports and layouts, builds, logs, and requirements.
- Customize test assets and builds.
- Create, modify, and delete logs and reports.

- Perform administrator tasks for a repository: manage projects, users, groups, and computers, as well as delete repositories.
- Change the database type of a ClearQuest or Rational Test database. Attach an existing ClearQuest database to a Rational repository.

There is no default password for the Admin user. You can log in as an Admin user, and you do not need a password. This lets you log into any Rational Test component after you first install the software.

Setting Up Security for the Admin User

When you connect to a repository for the first time, you should define a password for the Admin user to prevent unauthorized access.The Admin user always has full user access privileges and cannot be deleted from a repository. For more information about modifying a password, see *Changing a Password* on page 3-9.

Overview of the Public Group

Each repository that you create comes with a Public group, which is set as the default group. Each user that you add to the Rational Test database is a member of the Public group and takes on all privileges of the Public group.

If you are a member of the Administrators group, you can change the Public group privileges or make another group the default group for all new users.

For more information about security, see *Rational Test Database Privileges* on page 3-1.

Adding a Group

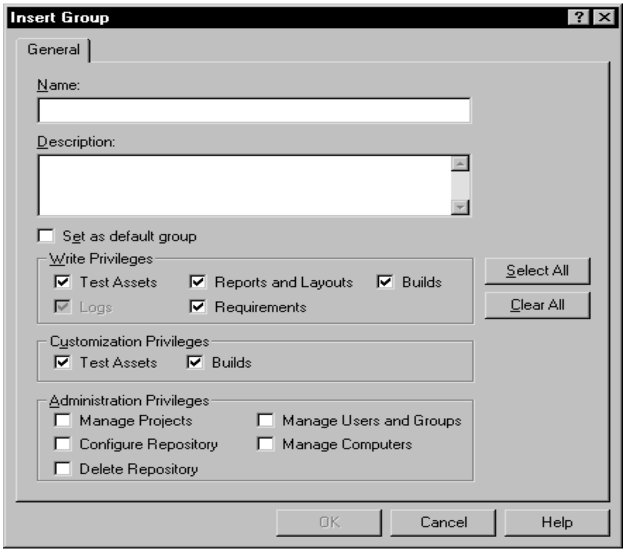
Groups allow you to set privileges for each user.

NOTE: You must connect to a repository before you can add a project, user, group, or computer to a Rational Test database. For more information, see *Connecting to a Repository* on page 2-20.

To add a group to a Rational Test database:

1. Click **Insert** → **Group**.
2. If you are connected to more than one repository, a dialog box appears with a list of all connected repositories. Select the repository to add a group to, and click **OK**.

The Insert Group dialog box appears.



Name:

Description:

☐ Set as default group

Write Privileges

☑ Test Assets ☑ Reports and Layouts ☑ Builds

☑ Logs ☑ Requirements

Customization Privileges

☑ Test Assets ☑ Builds

Administration Privileges

☐ Manage Projects ☐ Manage Users and Groups

☐ Configure Repository ☐ Manage Computers

☐ Delete Repository

3. Type an alphanumeric string of up to 40 characters for the name of the group.
4. Type a description of the group.
5. Optionally, click **Set as default group** to set a group as the default group for all users added.
6. Optionally, click **Select All** to select all group privileges. Alternatively, add and remove privileges from a group by selecting or clearing the appropriate check boxes. Click **Clear All** to clear all selected privileges.

   Write Privileges – Select the appropriate check boxes to give privileges to a group to create, modify, or delete test assets, logs, reports and layouts, requirements, and builds. When you insert a group, the **Builds** check box is automatically selected with the **Logs** check box grayed out to ensure that you have access to the logs for each build that you create, modify, or delete. To clear the **Logs** check box, clear the **Builds** check box.

Customization Privileges – Select the appropriate check boxes to give privileges to a group to customize test assets or builds. Test assets include scripts, and schedules. You can customize the properties of scripts, schedules, and build states using Rational TestManager. For more information about customizing scripts and schedules, see the chapter about planning tests in the *Using Rational Robot* manual. For more information about customizing build states, see the chapter about managing builds, log folders, and logs in the *Using Rational Robot* manual.

Administration Privileges – Select the appropriate check boxes to give privileges to a group to manage projects, users, groups, or computers.

7. Click **OK**.

Modifying a Group

You must have privileges to modify a group. For more information about privileges, see *Rational Test Database Privileges* on page 3-1.

To modify existing group information:

1. Connect to the repository that you want to modify. For more information about connecting to a repository, see *Connecting to a Repository* on page 2-20.
2. If **Groups** is not visible in the left pane of the main window, double-click **Rational Test Database**.
3. If the group that you want to modify is not visible, double-click **Groups** to display all groups.
4. Select the group to modify.
5. Click **Edit** → **Properties**.
6. Change the group information.
7. Click **OK** to apply your changes.

Deleting a Group

When you delete a group, you delete the group privileges for any users in the deleted group. You do not delete any users who are members of the deleted group.

You must have privileges to delete a group from a Rational Test database. For more information about privileges, see *Rational Test Database Privileges* on page 3-1.

To delete a group from a Rational Test database:

1. Connect to the repository that you want to modify. For more information about connecting to a repository, see *Connecting to a Repository* on page 2-20.

2. If **Groups** is not visible in the left pane of the main window, double-click **Rational Test Database**.
3. If the group that you want to delete is not visible, double-click **Groups** to display all groups.
4. Select the group to delete, and then click **Edit** → **Delete**.

   The Administrator prompts you to confirm the deletion.
5. Click **Yes**.

RequisitePro and ClearQuest Database Security

To set security privileges for a RequisitePro or ClearQuest database, see the RequisitePro or ClearQuest Help.

Managing Users

Users are typically the QA managers, developers, and test engineers who use Rational Test components to create, edit, run, monitor, analyze, and manage scripts that run across a network.

If you have privileges, you can add new users and modify information for existing users. For each user, you can specify a user ID, password, first name, last name, company, title, department, phone number, and email address.

You can assign privileges to a user by adding a user to a group. When you add a user to a group, the user takes all privileges of that group. A user can be a member of one or more groups.

You must have privileges to add new users, modify user information, and change user privileges. For more information about privileges, see *Rational Test Database Privileges* on page 3-1.

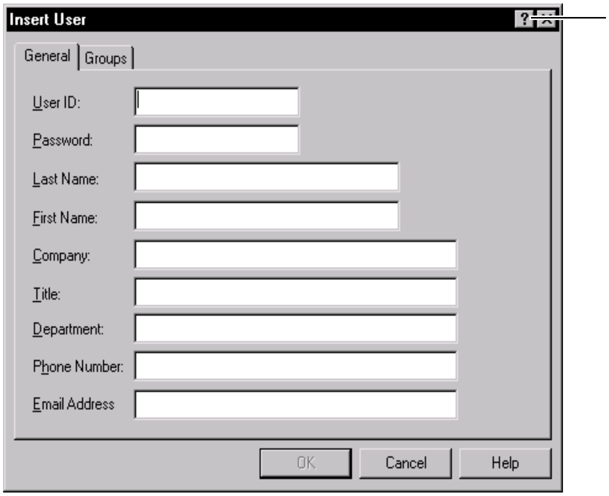
Adding a User

NOTE: You must connect to a repository before you can add a project, user, group, or computer to a Rational Test database. For more information, see *Connecting to a Repository* on page 2-20.

To add a user to a Rational Test database:

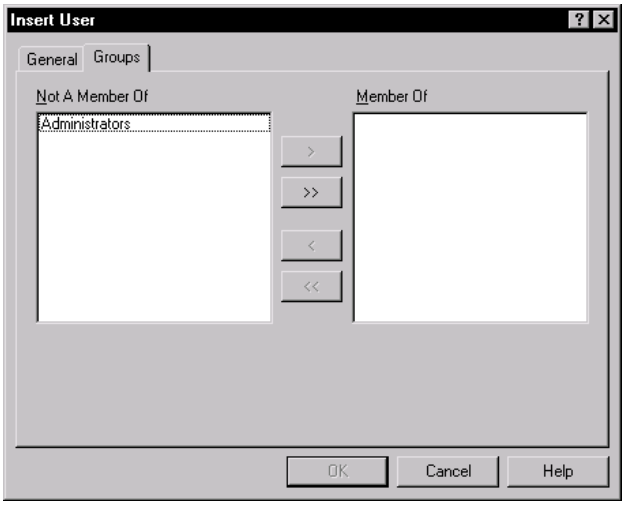
1. Click **Insert** → **User**.
2. If you are connected to more than one repository, a dialog box appears with a list of all connected repositories. Select the repository to add a user to, and click **OK**.

3. Type information about a user in the **General** tab of the dialog box.



For detailed information about an item, click the question mark, and then click the item.

4. Click the **Groups** tab to add or remove a user from a group.



5. In the **Not A Member Of** list, select one or more groups to make the user a member, and click > or > > .
6. Click **OK**.

For more information about groups, see *Rational Test Database Privileges* on page 3-1.

Modifying a User

You must have privileges to modify a user in a Rational Test database. For more information about privileges, see *Rational Test Database Privileges* on page 3-1.

To modify existing user information:

1. Connect to the repository that you want to modify. For more information about connecting to a repository, see *Connecting to a Repository* on page 2-20.
2. If **Users** is not visible in the left pane of the main window, double-click **Rational Test Database**.
3. If the user that you want to modify is not visible, double-click **Users** to display all users.
4. Select the user to modify.
5. Click **Edit** → **Properties**.
6. Click the **General** tab and change the general information.
7. Click the **Groups** tab and change the groups information.
8. Click **OK** to apply your changes.

Deleting a User

You must have privileges to delete a user from a Rational Test database. For more information about privileges, see *Rational Test Database Privileges* on page 3-1.

To delete a user from a Rational Test database:

1. Connect to the repository that you want to modify. For more information about connecting to a repository, see *Connecting to a Repository* on page 2-20.
2. If **Users** is not visible in the left pane of the main window, double-click **Rational Test Database**.
3. If the user that you want to delete is not visible, double-click **Users** to display all users.
4. Select the user to delete.

   You cannot delete the **admin** user.
5. Click **Edit** → **Delete**.

   The Administrator prompts you to confirm the deletion.
6. Click **Yes**.

Changing a Password

To change a password:

1. Double-click **Rational Test Database** in the left pane of the main window.
2. Double-click **Users** to display all users.
3. Select the user name of the password that you want to change.
4. Click **Edit** → **Properties**, or right-click and then click **Properties**.
5. Click the **General** tab of the dialog box.
6. Select the original password (xxxxx) in the **Password** box, and then type a new password over it.

   Type an alphanumeric string of up to 20 characters for the password. Passwords are case-sensitive. A user with a defined password must type that password and user ID to log into the Rational software.
7. Click **OK**.

Managing Computers

To manage your testing efforts, you can add a computer to a Rational Test database, modify computer information, and delete a computer from a Rational Test database.

Before You Add a Computer

Before you add a computer to a Rational Test database, you must determine the network name of the computer.

Determining the Network Name of a Computer

To determine the network name of a Windows computer:

1. Click **Start** → **Settings** → **Control Panel** → **Network**.
2. Click the **Identification** tab and write down the **Computer Name** (network name). You will need the network name when you add the computer to a Rational Test database.
3. Click **OK**.

To determine the network name of a UNIX computer:

1. At the UNIX prompt, type: `# hostname`
2. Write down the network name. You will need the network name when you add the computer to a Rational Test database.

Adding a Computer

When you add a computer to a Rational Test database, you do the following:

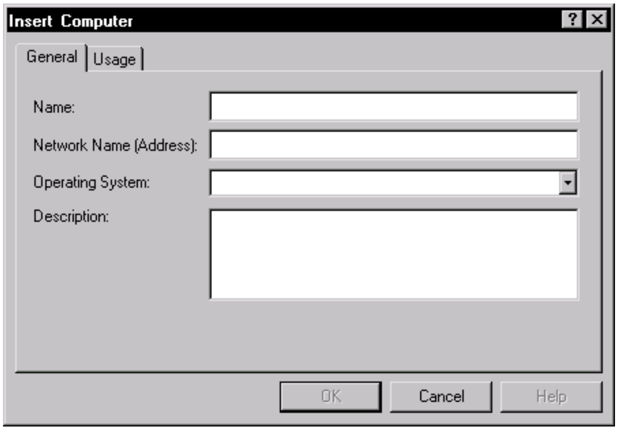
- Type the network name, operating system, and a description of the computer.
- Optionally, define how Rational LoadTest uses the computer in a schedule.
- Optionally, type the port name and number of the computer used for proxy recording with Rational LoadTest. For more information, see the *Using Rational LoadTest* manual.

Before you add a computer to the Rational Test database, you must determine the network name of your computer. For information about determining the network name of a computer, see *Determining the Network Name of a Computer* on page 3-9.

NOTE: You must connect to a repository before you can add a project, user, group, or computer to a Rational Test database. For information about connecting to a repository, see *Connecting to a Repository* on page 2-20.

To add a computer to a Rational Test database:

1. Click **Insert** → **Computer** to add a computer.
2. If you are connected to more than one repository, a dialog box appears with a list of all connected repositories. Select the repository to add a computer to, and then click **OK**.
3. Click the **General** tab, and then type information about the computer.



a. Type a descriptive name of up to 40 characters in the **Name** box.
   Use the **Name** box to help you easily identify your computer if the network name is not very descriptive. For example, you can type `WinNT4GUIagent` as the name of a Windows NT 4.0 system running GUI agents.

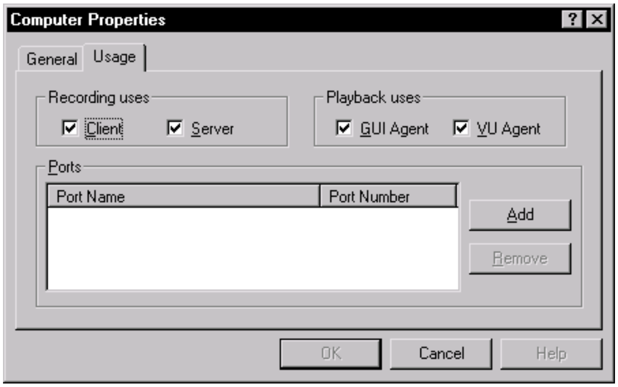
b. Type the **Network Name** or the TCP/IP address. For more information about determining a computer's network name, see *Before You Add a Computer* on page 3-9.

c. Select an operating system from the **Operating System** list.

NOTE: If your operating system does not appear in the **Operating System** list, you can add it to this list. For more information about adding an operating system, *Customizing Operating Systems* on page 3-13.

d. Optionally, type a description of the computer in the **Description** box.

4. Click the **Usage** tab and type information about how you plan to use the computer.



☑ Client ☑ Server

☑ GUI Agent ☑ VU Agent

a. Select the appropriate check box under **Recording uses** for the type of LoadTest recording on your computer.

   Client – Select if you are using this computer as a client.

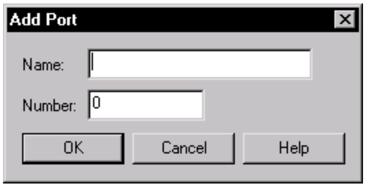
   Server – Select if you are using this computer as a database server, proxy server, or Web server.

b. Select the appropriate check box under **Playback uses** for the type of LoadTest playback on your computer.

GUI Agent – Select if you are using this computer to play back a GUI script.

VU Agent – Select if you are using this computer to play back a virtual user script.

c. Click **Add** to type the port name and number of the computer for the application-under-test.



Name:

Number: 0

i. Optionally, type the port name. The port name helps to identify the computer when you are recording tests.

i. Type the port number.

ii. Click **OK**.

To remove a port name and number, select the port name, and then click **Remove**.

5. Click **OK**.

If You Have Problems Adding a Computer

The Administrator checks the network name for you. If the network cannot find your computer, an error message appears.

If you typed in a valid network name for your computer and an error message still appears, we recommend that you type a static TCP/IP address as the **Network Name** in the **Insert Computer** dialog box.

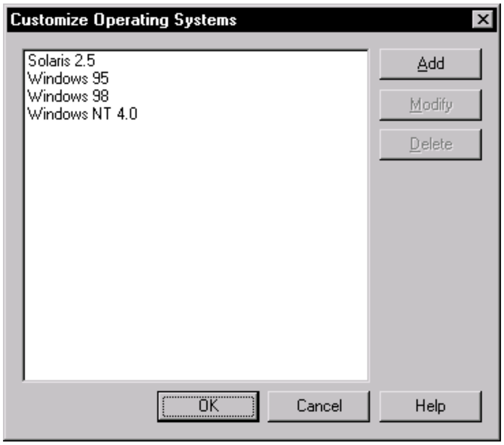
NOTE: If you use a TCP/IP address, it must be a static address. Contact your network administrator for the static TCP/IP address of the computer that you want to add to a Rational Test database.

Customizing Operating Systems

When you add a computer to a Rational Test database, if your computer's operating system does not appear in the list of operating systems, you can add it to the list.

To add an operating system to the list of operating systems:

1. Click **Tools** → **Customize Operating Systems**.
2. If you are connected to more than one repository, a dialog box appears with a list of all connected repositories. Select the repository to connect to, and then click **OK**.



Solaris 2.5
Windows 95
Windows 98
Windows NT 4.0

3. Click **Add**, and then type the name of the operating system that you want to add in the **New List Entry** box.
4. Click **OK**.
5. Click **OK**.

Modifying a Computer

You can modify the following information about a computer in the Rational Test database:

- Name
- Network name
- Operating system
- Description
- Recording uses and playback uses
- Ports name and number

You must have privileges to modify computers. For more information about privileges, see *Rational Test Database Privileges* on page 3-1.

To modify existing computer information:

1. Connect to the repository that you want to modify. For more information about connecting to a repository, see *Connecting to a Repository* on page 2-20.
2. If **Computers** is not visible in the left pane of the main window, double-click **Rational Test Database**.
3. If the computer that you want to modify is not visible, double-click **Computers** to display all computers.
4. Select the computer to modify.
5. Click **Edit** → **Properties**.
6. Click the **General** tab and change the general information.
7. Click the **Usage** tab and change the usage information.
8. Click **OK** to apply your changes.

Deleting a Computer

You must have privileges to delete a computer from a Rational Test database. For more information about privileges, see *Rational Test Database Privileges* on page 3-1.

To delete a computer from a Rational Test database:

1. Connect to the repository that you want to delete. For more information about connecting to a repository, see *Connecting to a Repository* on page 2-20.
2. If **Computers** is not visible in the left pane of the main window, double-click **Rational Test Database**.

3. If the computer that you want to delete is not visible, double-click **Computers** to display all computers.
4. Select the computer to delete.
5. Click **Edit** → **Delete**.

   The Administrator prompts you to confirm the deletion.
6. Click **Yes**.

▸▸▸ CHAPTER 4

Managing a SQL Anywhere Database Server

This chapter explains how to configure and manage a SQL Anywhere database server. It includes the following topics:

- Network server requirements
- Server terminology
- Creating a SQL Anywhere database server
- Changing a SQL Anywhere database server configuration
- Starting a SQL Anywhere database server
- Stopping a SQL Anywhere database server
- Deleting a SQL Anywhere database server
- Using the Sybase Central database management tool
- Using the Sybase ISQL utility
- Using the SQL Anywhere Service Manager
- What to do next

Network Server Requirements

If you use Sybase SQL Anywhere software for your database, make sure that your network server meets the following minimum hardware and software requirements:

Operating system	Windows NT 4.0 Server or Workstation 4.0
Processor	Pentium
Memory	64 MB
Disk space	100 MB
Network protocol	TCP/IP
Microsoft service pack	5

NOTE: Novell NetWare is no longer supported for network servers.

Server Terminology

The following table lists some terms that apply to a SQL Anywhere database server:

Term	Definition
server	The computer with Sybase SQL Anywhere Server software installed on it
service	Starts the SQL Anywhere database server
SQL Anywhere database server	A service that manages database requests for one or more repositories

Creating a SQL Anywhere Database Server

To create a SQL Anywhere database server, you perform the following tasks:

1. Install the Sybase SQL Anywhere Server software before you create a SQL Anywhere database server.

   For information about installing the SQL Anywhere Server software, see the installation guide for your product.

2. Choose a user account to run the SQL Anywhere database server and set privileges for that account.

   For information about setting privileges, see the next section, *Setting Privileges*.

3. Create a SQL Anywhere database server.

   For information about creating a SQL Anywhere database server, see *How to Create a SQL Anywhere Database Server* on page 4-5.

Setting Privileges

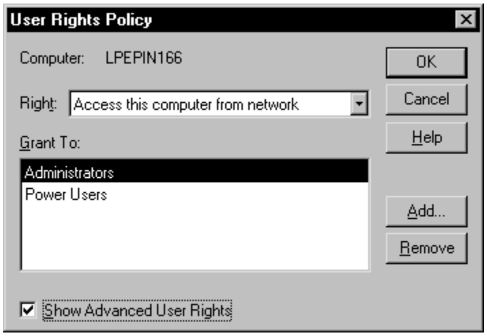
To set privileges for a user account running a SQL Anywhere database server:

1. Do one of the following:

If a computer is running	Then
Windows 2000 Server	**a.** Click **Start** → **Programs** → **Administrative Tools** → **Local Security Policy**. **b.** Click **Local Policies**, and then double-click **User Rights Assignment**.
Windows 2000 Professional	**a.** Click **Start** → **Settings** → **Control Panel**. **b.** Click **Administrative Tools**. **c.** Click **Local Security Policy**.
NT Server	**a.** Click **Start** → **Programs** → **Administrative Tools (Common)** → **User Manager for Domains.** **b.** Click **User** → **Select Domain** and type the local machine in the **Domain** box, and then click **OK.** **c.** Click **Policies** → **User Rights**. **d.** Click **Show Advanced User Rights**.
NT Workstation	**a.** Click **Start** → **Programs** → **Administrative Tools (Common)** → **User Manager.** **b.** Click **Policies** → **User Rights**. **c.** Click **Show Advanced User Rights**.

2. Do the following:

If a computer is running	Then
Windows 2000	**a.** Click **Start** → **Settings** → **Control Panel**. **b.** Click **Administrative Tools**. **c.** Click **Local Security Policy**.
Windows NT	**a.** Click **Access this computer from network** in the **Right** list and complete steps 3–8. **b.** Click **Log on as a service** in the **Right** list and complete steps 3–8. **c.** Click **Log on locally** in the **Right** list and complete steps 3–8.



User Rights Policy

Computer: LPEPIN166

Right: Access this computer from network

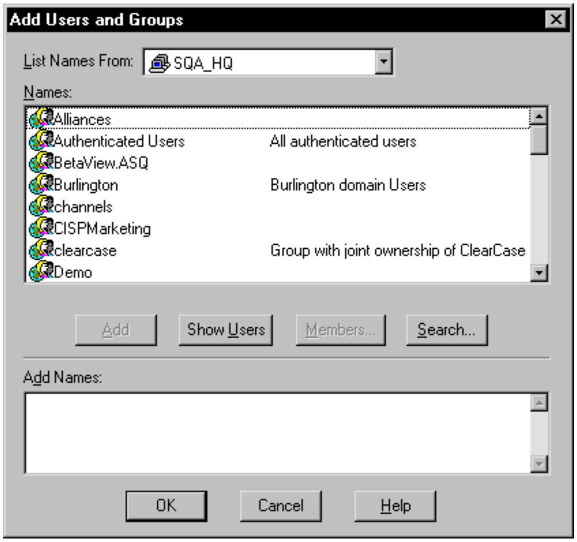
Grant To:

Administrators

Power Users

☑ Show Advanced User Rights

3. Click **Add**.



List Names From: SQA_HQ

Alliances	
Authenticated Users	All authenticated users
BetaView.ASQ	
Burlington	Burlington domain Users
channels	
CISPMarketing	
clearcase	Group with joint ownership of ClearCase
Demo	

4. Click the domain of the account running the SQL Anywhere database server from the **List Names From** list.
5. Click **Show Users**.
6. Click the account running the SQL Anywhere database server from the **Names** list.
7. Click **Add**, and then click **OK**.
8. Go back to step 4 to set all privileges.
9. Click **OK**.
10. Click **User** → **Exit**.

How to Create a SQL Anywhere Database Server

To create a SQL Anywhere database server:

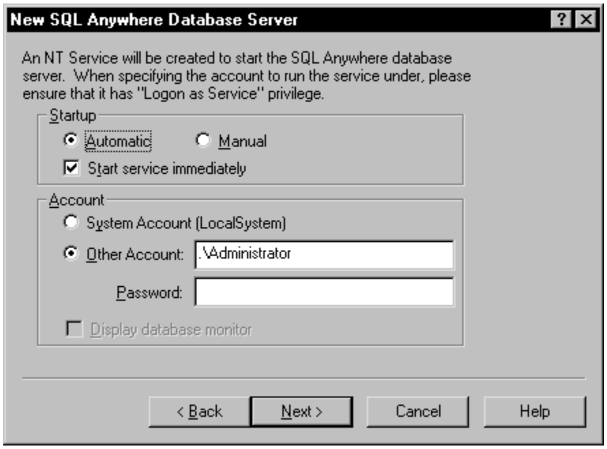
1. Click **Start** → **Programs** → *Rational product name* → **Rational Administrator**.
2. Click **File** → **New SQL Anywhere Database Server**.
3. Type an alphanumeric string of up to 30 characters that uniquely identifies this database server on the network, and then click **Next**.

4. Type the amount of space for caching data for the SQL Anywhere database server, and then click **Next**. We recommend 2048K for each Rational repository on the server.

5. Select one or more protocols to communicate with the SQL Anywhere database server, and then click **Next**.

> **NOTE:** To reduce connection time, check only those protocols that are actually used on your network.

6. Choose the **Startup** options, as described in the following steps.



New SQL Anywhere Database Server

An NT Service will be created to start the SQL Anywhere database server. When specifying the account to run the service under, please ensure that it has "Logon as Service" privilege.

Startup

● Automatic ○ Manual

☑ Start service immediately

Account

○ System Account (LocalSystem)

● Other Account: .\Administrator

Password:

□ Display database monitor

a. Click one of the following buttons to set the NT Service Startup options. These options start the SQL Anywhere database server when you turn on your system.

   – Click **Automatic** to start the NT Service that starts the SQL Anywhere database server when you start your system.

   – Click **Manual** to not start the NT Service that starts the SQL Anywhere database server when you start your system.

b. Select or clear the **Start service immediately** check box. This option starts the SQL Anywhere database server immediately after you create it. If the box is cleared, you must start the server manually.

7. Choose the **Account** options:

   a. Click one of the following buttons:

      – Click **System Account (LocalSystem)** to run the service under a local account.

> **NOTE:** If you choose this option, you will not be able to remotely configure a ClearQuest database using UNC.

      – Click **Other Account** to allow users access to a shared ClearQuest database or shared repository. We recommend that you choose this option because it allows users the most flexibility for accessing a database or repository.

      i. Type a domain and user account name in the **Other Account** box.

         The user account name is case-sensitive. This user account must have the appropriate privileges to access a shared database or repository. For information about setting privileges, see *Setting Privileges* on page 4-3.

      ii. Type the password for the account.
         The password is case-sensitive.

   b. Click **Display database monitor** to display the SQL Anywhere database monitor when the SQL Anywhere database server starts. This option is not available when you click **Other Account**.

8. Click **Nex**t.

   A summary appears which confirms your choices.

9. Click **Finish**.

Changing a SQL Anywhere Database Server Configuration

To change the configuration of a SQL Anywhere database server:

1. Click **Start** → **Programs** → *Rational product name* → **Rational Administrator**.
2. In the left pane, right-click a server under **SQL Anywhere Database Servers**, and then click **Properties**.
3. Optionally, click the **Basic** tab and change any of the options.
4. Optionally, click the **Advanced** tab and change any of the advanced settings.

> **NOTE:** The advanced settings are intended for use by, or in close cooperation with, Rational Technical Support. In most cases, these settings do not need modification.

5. Click **OK**.

Starting a SQL Anywhere Database Server

The Rational Administrator generates a startup command that can help you start a SQL Anywhere database server. The Administrator lets you manually start a SQL Anywhere database server on a Windows NT workstation or a Windows NT server.

To start a SQL Anywhere database server:

1. Click **File → Start SQL Anywhere Database Server**.

   If you created more than one database server, a dialog box appears with a list of all database servers.

2. Select the SQL Anywhere database server that you want to start, and then click **OK**.

SQL Anywhere Server Startup Switches

The basic startup command syntax is:

```
DBSRV50 -n ServerName -c CacheMemory -x Protocols
drive:\database\DB\RAS.DB -n Databasename
```

For example:

```
DBSRV50 c:\SQLDIR\DB\RAS.DB -n Proj1DB
```

or for a shared directory:

```
DBSRV50 \\KRANTI\SQLDIR\DB\RAS.DB -n Proj1DB
```

The default SQL Anywhere startup switches are:

SQL Anywhere Startup Switches	Description
DBSRV50	The SQL Anywhere server executable.
-n *ServerName*	The SQL Anywhere server name.
-c *CacheMemory*	The amount of cache memory. (We recommend using the default cache memory which is set to 2048K for each repository.)
-x *Protocols*	A comma-separated list of protocols.
drive:\Database or ***\\machinename\shared\***	The drive and location of the repository directory. We recommend that you use the Universal Naming Convention (UNC) here. (The **\DB\RAS.DB** path is required.)

For complete descriptions of all SQL Anywhere switches and components, see the SQL Anywhere Help.

Stopping a SQL Anywhere Database Server

To stop a SQL Anywhere database server:

1. Click **File** → **Stop SQL Anywhere Database Server**.

   If you have more than one database server running, a dialog box appears with a list of the running database servers.

2. Select the SQL Anywhere database server that you want to stop, and then click **OK**.

Deleting a SQL Anywhere Database Server

You must have privileges to delete a SQL Anywhere database server. For more information about privileges, see *Rational Test Database Privileges* on page 3-1.

To delete a SQL Anywhere database server:

1. Select the server to delete from the list of **SQL Anywhere Database Servers** in the left pane of the main window.
2. Click **Edit** → **Delete**.

   The Administrator prompts you to confirm the deletion.
3. Click **Yes** to delete the server.

Using the Sybase Central Database Management Tool

You can start the Sybase Central database management tool from the Administrator. This tool exposes SQL Anywhere database server settings, properties, and utilities in a graphical user interface. For more information about using the Sybase Central database management tool, see the SQL Anywhere Help.

To start the Sybase Central database management tool from the Administrator:

- Click **Tools** → **Sybase Central**.

Using the Sybase ISQL Utility

The Sybase ISQL provides you with an interactive environment for database browsing and for sending SQL statements to a particular SQL Anywhere database server. For more information, see the SQL Anywhere Help.

To start the Sybase ISQL utility from the Rational Administrator:

- Click **Tools** → **ISQL**.

Using the SQL Anywhere Service Manager

The SQL Anywhere Service Manager lets you start, stop, or pause a SQL Anywhere Server service. You can also edit the configuration parameters of each service. For more information, see the SQL Anywhere Help.

To start the SQL Anywhere Service Manager from the Rational Administrator:

- Click **Tools** → **SQL Anywhere Service Manager**.

What To Do Next

As the next step, you can create or delete a repository, or you can change the type of database. The following table lists the sections that provide more information about these tasks.

Task	See
Create a repository.	*Creating a Repository* on page 2-13
Change a database type.	*Changing a Rational Test Database Type* on page 2-22 and *Changing a ClearQuest Database Type* on page 2-23
Delete a repository.	*Deleting a Repository* on page 2-22

▸▸▸ APPENDIX A

Customizing Toolbars

The Rational Administrator has two default toolbars:

- **Standard** – Contains buttons for choosing the most frequently used commands for that component.
- **Tools** – Contains buttons for choosing other components.

All toolbar buttons correspond to menu commands. Click a toolbar button to immediately access the menu command. The toolbar buttons are dynamic. This means that some toolbar buttons are enabled only when you select related menu or toolbar commands.

Viewing Information About Toolbar Buttons

There are several ways to view information about a toolbar button and its corresponding menu command:

- To see the name of the button in a ToolTip, point to the button and pause.
- To see a brief description in the status bar, point to the button or menu command.
- To see more detailed information about the button or menu command, do one of the following:
  - Point to the button or menu command and press F1.
  - Click the **Help Pointer** icon on the right side of the Standard toolbar, and then point to the button or menu command and click the mouse button.

For a list of the toolbar buttons and a description of the corresponding commands, see the Help for that product or component.

Displaying Toolbars

To display or hide a toolbar:

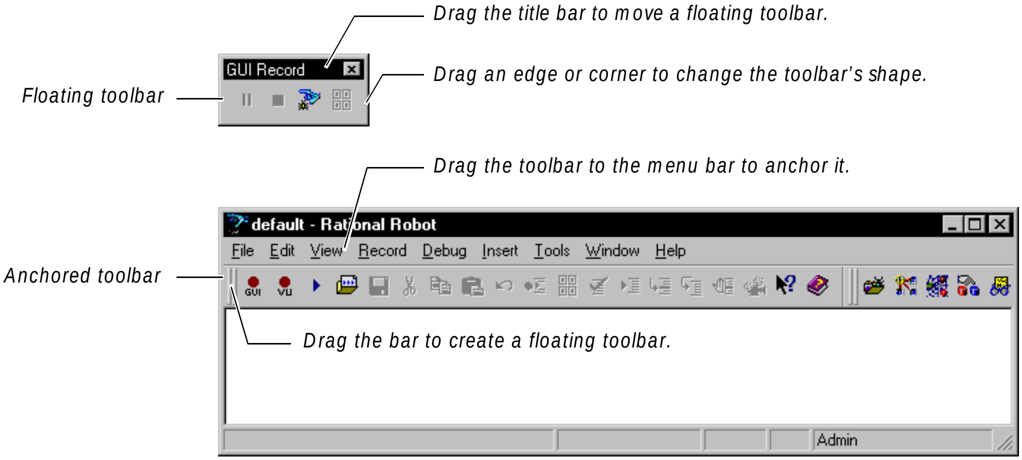
▶ Click **View** → **Toolbars**, and then click the name of the toolbar you want to display or hide.

A check mark appears in front of the name of each displayed toolbar.

Anchoring and Floating Toolbars

The Standard and Tools toolbars are anchored within each component's main window below the menu bar. However, you can drag an anchored toolbar from within a window and make it a floating toolbar, which you can position and resize independently of the main window. When you do this, the toolbar remains visible even when you minimize a component or when the component is hidden behind another application. You can also drag a floating toolbar and anchor it inside of the window.

Floating toolbars are always on top of all other windows. This ensures that they are never hidden. The following figure shows a floating toolbar and an anchored toolbar.



Drag the title bar to move a floating toolbar.

Drag an edge or corner to change the toolbar's shape.

Floating toolbar

Drag the toolbar to the menu bar to anchor it.

Anchored toolbar

Drag the bar to create a floating toolbar.

Setting Toolbar Options

To set the toolbar options:

1. Click **View** → **Toolbars** → **Customize**, or right-click a toolbar and click **Customize**.
2. In the **Toolbars** tab, select or clear the appropriate check boxes:

   Show ToolTips – Displays a ToolTip when you point to a button and pause.

   Cool Look – Changes the appearance of the toolbar buttons so that they have no borders. It does not change the behavior of the buttons.

   Large Buttons – Changes the size of the toolbar buttons.
3. Click **OK**.

Adding, Deleting, and Moving Toolbar Buttons

To add, delete, or move a toolbar button:

1. Click **View** → **Toolbars** → **Customize**, or right-click a toolbar and click **Customize**.
2. Click the **Commands** tab.
3. To add a button, click a menu name from the **Categories** list. Each name in the list represents a menu in the menu bar. Click a button to see its description. Drag the button to the toolbar. Make sure you release the mouse button within the toolbar.
4. To delete a button, drag it anywhere outside the toolbar.
5. To move a button, drag it to a new location.
6. Click **OK**.

Creating Your Own Toolbar

To create a custom toolbar that contains just the buttons you want:

1. Click **View** → **Toolbars** → **Customize**, or right-click a toolbar and click **Customize**.
2. Click the **Toolbars** tab.
3. Click **New**.
4. Type the name for the new toolbar and click **OK**.

5. Click the **Commands** tab.

6. Click a menu name from **Categories**.

7. Click a button to see its description. Drag the button to the new toolbar. Make sure you release the mouse button within the new toolbar.

8. Repeat steps 6 and 7 until you are finished adding buttons.

9. Click **OK**.

Resetting and Deleting Toolbars

To restore a default toolbar to its original configuration or to delete a custom toolbar:

1. Click **View** → **Toolbars** → **Customize**, or right-click a toolbar and click **Customize**.

2. In the **Toolbars** tab, do one of the following:

   – To reset a default toolbar to its original configuration, highlight the toolbar in the list and click **Reset**.

   – To delete a custom toolbar, highlight the toolbar in the list and click **Delete**.

   This command button changes depending on the type of toolbar you selected.

3. Click **OK**.

Glossary

action object – In TestFactory, an object in the application map that represents an action to which a control in the application responds. Typical actions are mouse left-click, mouse right-click, and mouse left-double-click; the corresponding action objects in the application map are LeftClick, RightClick, and LeftDoubleClick.

ActiveX control – A reusable software control that takes advantage of Object Linking and Embedding (OLE) and Component Object Modeling (COM) technologies. Developers can use ActiveX controls to add specialized functions to applications, software development tools, and Web pages. Robot can test ActiveX controls in applications.

actual results – In a functional test, the outcome of testing an object through a verification point in a GUI script. Actual results that vary from the recorded baseline results are defects or intentional changes in the application. See also *baseline results*.

Administrator – See *Rational Administrator*.

Agent computer – In LoadTest, a computer that has the Rational Agent software installed and that plays back a virtual user or GUI script. In a LoadTest schedule, you can identify the Agent computer on which to run a script. See also *Rational Agent*.

API recording – In Robot, a virtual user recording method that captures API calls between a specific client application and a server. These calls are captured on the client computer.

application map – In TestFactory, a hierarchical list of controls and actions in the application-under-test, as well as the states of the application-under-test and the transitions between those states. An application map can include UI objects and action objects, as well as TestFactory objects such as Pilots, Test Suites, and scripts.

application-under-test – The software being tested. See also *system-under-test*.

Asset Browser – A window that displays testing resources such as builds, queries, scripts, schedules, reports, report output, and logs. The Asset Browser is available in TestManager and LoadTest.

AUT – See *application-under-test*.

automated testing – A testing technique in which you use software tools to replace repetitive and error-prone manual work. Automated testing saves time and enables a reliable, predictable, and accurate testing process.

AutoPilot – In TestFactory, a tool for running scripts, Test Suites, and Pilots. The scripts and Test Suites can run on your local computer or on computers in the Test Lab. The Pilots run on your local computer, and the scripts they generate can run on your local computer or on computers in the Test Lab.

base state – In TestFactory, the known, stable state in which you expect the application-under-test to be at the start of each script segment. See also *script segment*.

baseline results – In a functional test, the outcome of testing an object through a verification point in a GUI script. The baseline results become the expected state of the object during playback of the script. Actual test results that vary from the baseline results are defects or intentional changes in the application. See also *actual results*.

best script – In TestFactory, an optimized script generated by a Pilot. A best script contains the fewest number of script segments that provide the most coverage of the source code or user interface in the application-under-test.

breakpoint – A feature of the Robot debugger. When you assign a breakpoint to a line of code, and then run the script in the debugger environment, the script stops executing at that line of code. Control returns to you, and the breakpoint line is displayed. From here you can view variables, perform other debugging activities, and continue executing the script.

build – A version of the application-under-test. Typically, developers add new features or enhancements to each incremental build. As team members test a build, they enter defects against those features that do not behave as expected. You use TestManager to define and manage builds.

built-in data test – A data test that comes with Robot and is used with the Object Data verification point. A data test uses a specific property of the object, in conjunction with other parameters, to determine the data to capture. Although built-in data tests cannot be edited, renamed, or deleted, they can be copied and then edited, and they can be viewed. See also *custom data test*.

ClearQuest – See *Rational ClearQuest*.

client/server – An architecture for cooperative processing in which the software tasks are split between server tasks and client tasks. The client computer sends requests to the server, and the server responds.

code coverage – In TestFactory, the percentage of code that is tested by a script. This percentage is based on the portion of the code that a script touches, relative to all code in the application-under-test. A Pilot can use code coverage to determine the best script for a run. See also *UI coverage*.

command ID – In LoadTest's VU language, an identifier for a command. Robot automatically assigns a unique command ID, composed of an alphanumeric prefix and a three-digit number, to each emulation command. Because command IDs appear in both the virtual user script and the LoadTest report output, they enable you to determine the relationship between an emulation command and its response times.

command ID prefix – In LoadTest, a prefix for a unique emulation command ID. The prefix defaults to the script name (up to the first seven characters). However, you can define the prefix in the Generator tab of the Virtual User Record Options dialog box.

custom data test – A customer-defined data test used with the Object Data verification point. A data test uses a specific property of the object, in conjunction with other parameters, to determine the data to capture. Custom data tests are created within your organization and are stored in the repositories that were active when they were created. They can be edited, renamed, and deleted. See also *built-in data test*.

data test – A test that captures the data of an object with the Object Data verification point. See also *built-in data test* and *custom data test*.

datapool – A source of test data that GUI scripts and virtual user scripts can draw from during playback. You can automatically generate datapools using TestManager, or you can import datapool data from other sources such as your database.

dependency – In LoadTest, a method of coordinating an object in a schedule with an event. For example, if the script Query is dependent upon the script Connect, then Connect must finish executing before Query can begin executing. See also *event*.

distributed architecture – Architecture in which computer systems work together and communicate with each other across LAN, WAN, or other types of networks. A client/server system is an example of distributed architecture.

distributed functional test – In LoadTest, a test that uses multiple Agent computers to execute multiple GUI scripts written in the SQABasic language.

dynamic load balancing selector – A type of selector in a LoadTest schedule. Items in the selector, such as scripts, are executed according to a weight that you set.

emulation commands – VU language statements or commands that emulate client activity, evaluate the server's responses, and perform communication and timing operations. LoadTest stores the results of emulation commands in a log file, which you can view from the LogViewer.

emulation functions – VU language functions that emulate client activity and evaluate the server's responses. Unlike emulation commands, emulation functions do not perform communication and timing operations, and they are not logged.

environment control commands – VU language commands that let you control a virtual user's environment by changing the VU environment variables. For example, you can set the level of detail that is logged or the number of times that virtual users attempt to connect to a server.

event – An item in a LoadTest schedule upon which another item is dependent. For example, if the script Connect sets an event and the script Query depends on this event, Connect must finish executing before Query can begin executing. See also *dependency*.

external script – A script that runs a program created with any tool. You plan and run external scripts in TestManager.

fixed user group – In LoadTest, a group that contains a scalable number of users. When you create a fixed user group, you indicate the maximum number of users that you will run in the group. Typically, you use fixed user groups in functional tests, which do not add a workload to the system.

flow control statements – In the VU and SQABasic languages, statements that let you add conditional execution structures and looping structures to a script.

functional test – A test to determine whether a system functions as intended. Functional tests are performed on GUI objects and objects such as hidden DataWindows and Visual Basic hidden controls.

Grid Comparator – The Robot component for reviewing, analyzing, and editing data files for text and numeric verification points in grid formats. The Grid Comparator displays the differences between the recorded baseline data and the actual data captured during playback.

GUI script – A type of script written in the SQABasic language. It contains GUI actions such as keystrokes and mouse clicks. Typically, a GUI script also contains verification points for testing objects over successive builds of the application-under-test.

GUI user – The type of user that is emulated when a GUI script is executed. Only one GUI user at a time can run on a computer.

hidden object – An object that is not visible through the user interface. Hidden objects include objects with a visible property of False and objects with no GUI component.

IDE – Integrated Development Environment. This environment consists of a set of integrated tools that are used to develop a software application. Examples of IDEs supported by Robot include Oracle Forms, PowerBuilder, Visual Basic, and Java.

Image Comparator – The Robot component for reviewing and analyzing bitmap image files for Region Image and Window Image verification points. The Image Comparator displays differences between the recorded baseline image and the actual image captured during playback. The Image Comparator also displays unexpected active windows that appear during playback.

instrumentation – In TestFactory, the process of inserting code coverage counters into the application-under-test. These counters record how much code is executed during a script run. See also *object code instrumentation* and *source code instrumentation*.

load – See *workload*.

load balancing – See *workload balancing*.

LoadTest – See *Rational LoadTest*.

log – A repository object that contains the record of events that occur while playing back a script or running a schedule. A log includes the results of all verification points executed as well as performance data that can be used to analyze the system's performance.

LogViewer – See *Rational LogViewer*.

low-level recording – A recording mode that uses detailed mouse movements and keyboard actions to track screen coordinates and exact timing. During playback, all actions occur in real time, exactly as recorded.

manual script – A set of testing instructions to be run by a human tester. The script can consist of steps and verification points. You create manual scripts in TestManager.

Master computer – A computer that executes LoadTest. From this computer, you create, run, and monitor schedules. When the run is finished, you use it to analyze test results.

mix-ins – See *Pilot mix-ins*.

network recording – In Robot, a virtual user recording method that records packet-level traffic. This traffic is captured on the wire.

next available selector – In LoadTest schedules, a selector that distributes each item such as a script, delay, or other selector to an available computer or virtual user. This type of selector is used in a GUI schedule. The next available selector parcels out the items sequentially, based on which computers or virtual users are available.

object – An item on a screen, such as a window, dialog box, check box, label, or command button. An object has information (properties) associated with it and actions that can be performed on it. For example, information associated with the window object includes its type and size, and actions include clicking and scrolling. In some development environments, a term other than *object* is used. For example, the Java environment uses *component*, and the HTML environment uses *element*.

object code instrumentation – In TestFactory, the process of inserting code coverage counters into the executable file of the application-under-test. These counters record how much of the program a script tests. See also *instrumentation* and *source code instrumentation*.

Object-Oriented Recording® – A script recording mode that examines objects in the application-under-test at the Windows layer. Robot uses internal object names to identify objects, instead of using mouse movements or absolute screen coordinates.

Object Properties Comparator – The Robot component that you use to review, analyze, and edit the properties of objects captured by an Object Properties verification point. The Object Properties Comparator displays differences between recorded baseline data and the actual data captured during playback.

Object Scripting commands – A set of SQABasic commands for accessing an application's objects and object properties. You add Object Scripting commands manually when editing a script.

Object Testing® – A technology used by Robot to test any object in the application-under-test, including the object's properties and data. Object Testing lets you test standard Windows objects and IDE-specific objects, whether they are visible in the interface or hidden.

OCI – Object Code Insertion. The Rational technology used in TestFactory to instrument object code and measure how much of the application-under-test a script tests. See also *code coverage* and *object code instrumentation*.

performance test – A test that determines whether a multi-client system performs within user-defined standards under varying loads. Performance tests are always run from a schedule in LoadTest.

Pilot – In TestFactory, a tool for generating scripts automatically.

Pilot mix-ins – In TestFactory, a list of Pilots that are executed on a random basis during the run of a lead Pilot. Mix-ins are useful for randomly testing multiple areas of the application-under-test. To make tests more realistic, you can combine mix-ins and scenarios.

Pilot scenario – An ordered list of Pilots that are executed during the run of a Pilot. A Pilot scenario is useful for testing UI objects that need to be exercised in a specific order. To make tests more realistic, you can combine scenarios and mix-ins.

project – A collection of data, including test assets, defects, requirements, and models, that can facilitate the development and testing of one or more software components.

proxy recording – In Robot, a virtual user recording method that captures the client/server conversation on the network wire rather than on the client computer. Proxy recording allows Robot to capture network packets that are not visible to it during network recording — for example, if the client and server are in different network segments.

query – A request for information stored in the repository. A query consists of a filter and several visible attributes — the columns of data to display, the width of the column, and the sort order.

random selector – A type of selector in a LoadTest schedule. Items in the selector, such as scripts, are randomly executed. Random selectors can be with replacement, where the odds are the same, or without replacement, where the odds change with each iteration.

Rational Administrator – The component for creating and maintaining repositories, projects, users, groups, computers, and SQL Anywhere servers.

Rational Agent – The LoadTest software that resides on a shared network drive and runs on each computer where testing occurs. The entries specified in a schedule play back on the Agent computer, which reports on their progress and status as they run. See also *Agent computer*.

Rational ClearQuest – The Rational product for tracking and managing defects and change requests throughout the development process. With ClearQuest, you can manage every type of change activity associated with software development, including enhancement requests, defect reports, and documentation modifications.

Rational LoadTest – The Rational Test component for running performance, stress, scalability, multi-user, and distributed functional tests on multiple Agents connected by a network. With LoadTest, you can initiate test runs and monitor tests from a master computer that manages the test process. LoadTest is available only in Rational Suite PerformanceStudio.

Rational LogViewer – The Robot component for displaying logs, which contain the record of events that occur while playing back a script or running a schedule. Also, the component from which you start the four Comparators.

Rational PerformanceArchitect – The Rational component that lets you test the performance of COM/DCOM applications. With Rational PerformanceArchitect, you can create a Rose sequence or collaboration diagram, convert it to a virtual user script, and then use Rational Suite PerformanceStudio to edit the script and run the performance tests.

Rational repository – A database that stores application testing information, such as test requirements, scripts, and logs. All Rational Suite TestStudio and Rational Suite PerformanceStudio products and components on your computer update and retrieve data from the same connected repository. A repository can contain either a Microsoft Access or a Sybase SQL Anywhere database.

Rational RequisitePro – The Rational product for organizing, managing, and tracking the changing requirements of your system.

Rational Robot – The Rational product for recording, playing back, debugging, and editing scripts.

Rational SiteCheck – The Robot component for managing your intranet or World Wide Web site. You can use SiteCheck to visualize the structure of your Web site, and you can use it with Robot to automate Web site testing.

Rational Synchronizer – The Rational tool that ensures the consistency of data across several Rational products.

Rational TestAccelerator – An agent application that executes scripts. TestFactory uses computers running TestAccelerator as remote machines on which to run automated distributed tests.

Rational TestFactory – The Rational Test component for mapping an application-under-test and generating scripts automatically. TestFactory is available in Rational Suite TestStudio and Rational Suite PerformanceStudio.

Rational TestManager – The Robot component for managing the overall testing effort. You use it to define and store information about test documents, requirements, scripts, schedules, and sessions.

Report Layout Editor – The TestManager component for customizing the layout of reports.

repository – See *Rational repository*.

RequisitePro – See *Rational RequisitePro*.

Robot – See *Rational Robot*.

scalable user group – In LoadTest, a group that contains a varying number of users. When you create a scalable user group, you assign it a percentage of the total workload. Assume you have a scalable user group that is 50 percent of the workload. If you run a test with 10 users, the group will contain 5 users. If you run a test with 100 users, the group will contain 50 users.

scenario – In LoadTest, a modular group of scripts and other items in a schedule that is used by more than one user group. A scenario can contain scripts, delays, and synchronization points.

scenario – See *Pilot scenario*.

schedule – In LoadTest, structure that you create to specify how scripts should be played back. A schedule can contain GUI scripts and virtual user scripts, and can indicate the number of times to repeat a script and the computer on which the script will run. In performance testing, a schedule is used to create a workload. In distributed functional testing, a schedule is used to distribute scripts among various computers.

script – A set of instructions used to navigate through and test an application. You can generate scripts in a variety of ways. You can use Robot to record scripts used in functional testing and performance testing. You can also use TestManager to create and manage manual scripts, and to manage external scripts created with a third-party testing tool. A script can have properties associated with it, such as the purpose of the script and requirements for the script. See also *external script*, *GUI script*, *manual script*, and *virtual user script*.

script outline – In TestFactory, the readable version of a script. A script outline contains a description of the actions that Robot performs while running the script.

script segment – In TestFactory, a section of a script that tests a particular element of product functionality. A Pilot generates a script segment by starting the application-under-test in a base state, navigating through the part of the product that you are testing, and returning the application-under-test to the base state. See also *base state*.

seed – An initial number fed to a random number generator. Using the same seed produces the same series of random numbers. In LoadTest, you use seeds to generate think times.

selector – An item that you insert in a LoadTest schedule to indicate how often and in what order to run scripts.

sequential selector – In a LoadTest schedule, a type of selector that executes each script, delay, or other item in the same order in which it appears in the schedule.

session – In virtual user recording, one or more scripts that you record from the time you begin recording until the time you stop recording. Typically, the scripts in a session represent a logical flow of tasks for a particular user, with each script representing one task. For example, a session could be made up of three scripts: *login*, *testing*, and *logout*. In TestFactory, a session is the period of time that the TestFactory application or a window is open.

shared variable – An integer variable that multiple scripts and multiple virtual users can read and write to. You can see the value of a shared variable while monitoring a LoadTest schedule. For example, you can set a shared variable as a flag to end a playback session. Each script can check the flag to see if the session should end. When that flag is set, exit tasks can be performed.

shell script – A script that calls or groups several other GUI scripts and plays them back in sequence. Shell scripts provide the ability to create comprehensive tests and then store the results in a single log.

SiteCheck – See *Rational SiteCheck*.

source code instrumentation – In TestFactory, the process of inserting code into the source code of the application-under-test. This code measures how much of the source code a script tests. See also *instrumentation* and *object code instrumentation*.

SQABasic – The Robot scripting language for recording GUI actions and verifying GUI objects. SQABasic contains most of the syntax rules and core commands that are contained in the Microsoft Basic language. In addition, SQABasic has commands that are specifically designed for automated testing. See also *VU*.

stable load – In LoadTest, a condition that occurs when a specified number of virtual users have logged on to the system-under-test and are active. When the stable load criterion is met, LoadTest begins measuring the load.

streak – When running a virtual user schedule in LoadTest, a series of successes or failures for emulation commands. You can see a streak while monitoring a schedule.

structural test – A test to determine whether the structure of a Web site is consistent and complete. A structural test ensures that an application's interdependent objects are properly linked together. You perform a structural test using SiteCheck.

synchronization point – In LoadTest, a place where emulated virtual users stop and wait until all other synchronized users reach that point. When all users reach the synchronization point, they are released and continue executing.

Synchronizer – See *Rational Synchronizer*.

system tuning – In LoadTest, the process of optimizing a system's performance by changing hardware resources and software configuration parameters while using a constant workload.

system-under-test – The system being tested. This includes the computers and any software that can generate a load on the system, networks, user interfaces, CPUs, and memory. See also *application-under-test*.

test assets – The resources that facilitate the planning or development phases of the testing effort. Examples of test assets include scripts, schedules, sessions, test documents, and test requirements.

test development – The process of developing tests to verify the operation of a software application. This includes creating scripts that verify that the application-under-test functions properly. Test development lets you establish the baseline of expected behavior for the application-under-test.

test documents – Test plans, project schedules, resource requirements, and any other documents that are important to your project. You develop your test documents using your own word processing or scheduling program; you then reference the name and location of the document in TestManager. This lets members of the test and development team locate documents quickly.

Test Lab – A collection of computers on which TestAccelerator is running. In TestFactory, you can distribute the scripts associated with a Pilot, a Test Suite, or the AutoPilot to run on computers in the Test Lab. See also *Rational TestAccelerator*.

Test Suite – In TestFactory, a tool for running a collection of scripts as a group.

TestAccelerator – See *Rational TestAccelerator*.

TestFactory – See *Rational TestFactory*.

TestManager – See *Rational TestManager*.

Text Comparator – The Robot component for reviewing, analyzing, and editing data files for text and numeric verification points in any format except grids. The Text Comparator displays the differences between the recorded baseline results and the actual results.

think time – In virtual user and GUI scripts, think times are delays that simulate a user's pauses to type or think while using an application. With virtual user scripts, LoadTest calculates the think time at runtime, based on think time VU environment variables that are set in the script. You can set a maximum think time in Robot. With GUI scripts, Robot uses the actual delays captured between keystrokes, menu choices, and other actions.

transaction – In LoadTest, a logical unit of work performed against a server. For example, submitting a search query or submitting a completed form to a Web server are both transactions.

transaction rate – In LoadTest, the playback speed calculated as a function of number of transactions per unit of time. For example, if a script contains one transaction, and each script is started at half-second intervals, your transaction rate would be 2 per second.

transactor – In LoadTest, an item that you insert in a LoadTest schedule to indicate the number of user-defined transactions that a virtual user performs in a given time period.

UI coverage – In TestFactory, the percentage of objects in the application map that are tested by a Pilot-generated script. This percentage is the proportion of UI objects that the script touches, relative to all UI objects available to the Pilot. A Pilot can use UI coverage to determine the best script for a run. See also *code coverage*.

UI object properties – Attributes of object classes and UI objects that TestFactory uses to map applications and generate scripts.

unexpected active window – A window that appears during script playback that interrupts the script playback process and prevents the expected window from being active. For example, an error message generated by the application-under-test is an unexpected active window. You can view unexpected active windows in the Image Comparator.

user group – In LoadTest, a collection of users that execute similar tasks and generate the same basic workload. Accountants and data entry operators are examples of user groups.

verification – The process of comparing the test results from the current build of the software to its baseline results.

verification point – A point in an SQABasic script that confirms the state of one or more objects. During recording, a verification point captures object information from the application-under-test and stores it as the baseline. During playback, a verification point recaptures the object information and compares it to the baseline. In a manual script, a verification point is a question about the state of the application-under-test.

virtual user – In LoadTest, a type of user that is emulated when a virtual user script is executed. A computer can run multiple virtual users simultaneously.

virtual user script – A type of script written in the VU language. Virtual user scripts contain client/server requests and responses as well as user think times.

VU – The Robot scripting language for recording a client's requests to a server. VU provides most of the syntax rules and core commands available in the C programming language. In addition, VU has emulation commands and functions that are specifically designed for automated performance testing. See also *SQABasic*.

wait state – A delay or timing condition that handles time-dependent activities.

workload – In LoadTest, the set of all activities that users perform in an actual production setting of the system-under-test. You can use LoadTest to emulate a workload.

workload balancing – In LoadTest, the act of distributing activities so no one system or device becomes a bottleneck.

workload model – In LoadTest, the workload model is represented as a schedule. You can play back this schedule and analyze the response times.

‣‣‣ Index