Release Notes

RATIONAL ROSE® REALTIME

VERSION: 2002.05.20

PART NUMBER: 800-025113-000

WINDOWS/UNIX

Rational®
the software development company

support@rational.com
http://www.rational.com

Part Number: 800-025113-000

Version Number: 2002.05.20

Contents

Preface

Contents

This manual provides the information on the following key topics:

This chapter is organized as follows:

Audience

This guide is intended for all readers, including managers, project leaders, analysts, developers, and testers.

Other Resources

- Online Help is available for Rational Rose RealTime.

  Select an option from the **Help** menu.

  All manuals are available online, either in HTML or PDF format. To access the online manuals, click **Programs > Rational Rose RealTime > Rational Rose RealTime Documentation** from the **Start** menu.

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

Contacting Rational Technical Publications

To send feedback about documentation for Rational products, please send e-mail to our Technical Documentation Department at techpubs@rational.com.

Contacting Rational Customer Support

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

Your Location	Telephone	Fax	E-mail
North America	(800) 433-5444 (toll free) (408) 863-4000 Cupertino, CA	(781) 676-2460 Lexington, MA	support@rational.com
Europe, Middle East, Africa	+31 (0) 20-4546-200 Netherlands	+31 (0) 20-4546-202 Netherlands	support@europe.rational.com
Asia Pacific	+61-2-9419-0111 Australia	+61-2-9419-0123 Australia	support@apac.rational.com

Note: When you contact Rational Technical Support, please be prepared to supply the following information:

- Your name, telephone number, and company name
- Your computer's make and model
- Your computer's operating system and version number
- Product release number and serial number
- Your Service Request number (if you are following up on a previously-reported problem, for example, [SR# 111222333])

Introduction 1

Contents

This chapter is organized as follows:

Overview

Thank you for selecting **Rational Rose RealTime**, the real-time industry's leading environment that unifies software teams by integrating with best-in-class Rational products including Rational ClearCase, Rational RequisitePro, and Rational SoDA.

These release notes describe:

Please read these Release Notes before you install or use Rational Rose RealTime.

Note: In some cases, you may note discrepancies between the printed documentation and the online documentation. In these cases, refer to the online documentation; however, the Readme will contain any additional updates and should be followed.

Please visit the Rational Web site for the latest Release Notes, patches, and additional information:

http://www.rational.com/support

If you encounter any problems while installing or running Rational Rose RealTime, please check here first to see if you encountered a known problem. If you find a problem that is not yet documented, please call Rational Technical Support so we can investigate it, provide you with a workaround, and track the problem for further action.

Contact information for this release is in *Technical Support* on page 67.

PLEASE READ FIRST

Please read the following notes before you install Rational Rose RealTime.

You Must Install License Keys to Run this Product

With your software shipment, you received an Welcome letter containing your Start-up License Key Certificates for this product. You need these keys in order to run your software. The start-up keys will expire a few weeks after shipment. Ensure that you request your permanent license keys as soon as they are available. The availability date for your permanent keys is indicated in the top section of your Start-up Licence Key Certificate.

Please check the *Rational Rose RealTime Installation Guide* for detailed instructions on how to install your license keys and how to request permanent license keys.

Note: If you are installing license keys on a UNIX platform, please refer to the *Installation Guide* for instructions. **Do not** follow the instructions on the Rational Start-up License Certificate or on the envelope in which the certificate is delivered to you.

Notice to Customers Running Earlier Rational Rose RealTime Releases on Windows

You cannot run earlier versions of Rational Rose RealTime on Windows NT, Windows 2000, and Windows XP Pro with the latest version of Rational Rose RealTime at the same time.

Notice to UNIX - HPUX 10.20 Users

To obtain the Rational Rose RealTime Companion CD for the Rational Rose RealTime Professional Edition software, see Contacting Rational Customer Support.

What's New

Welcome to Rational Rose RealTime Version 2002.05.02.305.000. Based on extensive customer consultation and feedback, this service release contains a wide range of updates and corrections designed to streamline user workflows and enhance developer productivity. Listed below are some of the more visible changes that you will discover in this release. We hope that you find the following enhancements helpful and we look forward to serving your needs in future releases:

New Features to Enhance Model Navigation and Simplify User Workflow

- Find/Show references: On Specification dialogs and in the **Model View** tab in the browser, you can now search the model and code for references to the selected capsule, class, attribute, operation, protocol, or signal. The results appear in the **Find** window.
- Resizable show usage window.
- Moving model elements feature to simply moving capsules, protocols, classes, and packages.
- Smart resizing of Specification dialog list columns.
- C, C++ and Java language-specific Attribute, Operation and Aggregation tools to simplify UML adoption. Fine-grained computation and control over class dependencies.
- Simplified Java modeling with import Java tool, called Add External Java, that imports class, operation and attribute descriptors into UML model directly from class and jar files.
- New **Diagrams** tab on Specification dialogs for faster navigation to, and manipulation of, model element diagrams.
- Faster model navigation with referenced class hyperlinks on Specification dialogs supporting operation return type, argument type, classes, operations, port protocol class, capsule role capsule class, message operation, message signal, and more.

- New shortcut menus to navigate directly to Class specifications from operation, attribute, argument, and role types.
- Full operation signature display on Specification dialogs and Code window.
- Additional protection with optional prompt for: closing Specification dialogs; deleting model elements; quitting the toolset.
- Inheritance browser now supports rearrangement of inheritance hierarchies without diagrams.
- Specification History keeps track of the last Specification dialogs you visited. Cycle through the specification sheets with hot keys. Open selected, close selected, Specification dialogs. Lock frequently visited Specification dialogs in the history.
- Diagram placement improvements and a new larger minimum size for diagram graphics.
- Better display of inherited operations.
- More keyboard shortcuts and improved visibility of shortcuts
- Optional line wrapping on internal editor.
- New menu items, **Help > Email Technical Support > Problem Report**, **Help > Email Technical Support > Feature Request**, and **Help > Email Technical Support > Support Request**, to simplify communication and feedback between users and the Rational Team.

RQA-RT Enhancements

- Performance improvements. Suppress display of Sequence diagrams and Trace during test execution.
- Improved test results overview with date and time stamp and label generated test harnesses, test results and purloined test run pass and fail diagrams.

Build and Target Enhancements

- Automatically save the output on the **Build Log** tab in the **Output** window to a file.
- Filtering build errors and sorting information by column.
- Show/Hide build warnings.
- The TargetRTS Wizard simplifies customization and adaptation to operating systems and compilers.
- New target reference configurations for ITRON 3 and targets without operating systems (NoRTOS).

- Improved C target support - target observability for single-threaded targets.
- Microsoft Visual Studio 7.0 source debugger integration.
- Reuse target build environment when building models using target setup script, setup.pl, and vssetup.pl to get the Microsoft Visual Studio environment directly from the registry.

Documentation Improvements

- New extensibility API to extend existing build, model, configuration management APIs through to target control and source debugger integration.
- New example models and animated demonstrations to accelerate learning.
- Comprehensive keyboard shortcut overview for quick reference.
- A more comprehensive index including troubleshooting topics in the online help.

New Rational Developer Network Menu Option

The Rational Developer Network provides guidance to implement and deepen your knowledge of Rational tools and best practices. It includes immediate access to white papers, artifacts, code, discussions, training, and documentation. For more information, see http://www.rational.net.

From Rational Rose RealTime, click
Help > Rational on the Web > Rational Developer Network to open the Rational Developer Network.

Referenced Configurations and Toolchain Requirements

2

Contents

This chapter is organized as follows:

This chapter describes the referenced configuration and toolchain requirements for running Rational Rose RealTime.

Note: Rational Rose RealTime is not supported on Windows 98, or Windows XP Home.

Referenced Configurations

Rational Rose RealTime references the following:

Windows NT

The minimum supported configuration for running Rational Rose RealTime on Windows NT is:

- Windows NT 4.0 with service pack 6a
- Minimum Pentium 150 MHz. We recommend 500 MHz or faster CPU
- Minimum 128 MB of RAM. We recommend 256 MB or more of RAM
- Minimum 325 MB of disk space for the Rational Rose RealTime installation
- Minimum display 1024 X 768. We recommend 1280 X 1024 or higher
- Postscript printer for printing
- Browser requirement - Internet Explorer 5.01 (SP2) or 5.5 or Netscape Navigator 4.7 or 6.0. We recommend Internet Explorer 5.5

Windows 2000

The minimum supported configuration for running Rational Rose RealTime on Windows 2000 is:

- Windows 2000 Professional, with service pack 1 and service pack 2
- Minimum Pentium 150 MHz. We recommend 500 MHz or faster CPU
- Minimum 128 MB of RAM. We recommend 256 MB or more of RAM
- Minimum 325 MB of disk space for the Rational Rose RealTime installation
- Minimum display 1024 X 768. We recommend 1280 X 1024 or higher
- Postscript printer for printing
- Browser requirement - Internet Explorer 5.01 (SP2) or 5.5 or Netscape Navigator 4.7 or 6.0. We recommend Internet Explorer 5.5

Windows XP Pro

The minimum supported configuration for running Rational Rose RealTime on Windows XP Pro is:

- Minimum Pentium 300 MHz. We recommend 500 MHz or faster CPU
- Minimum 128 MB of RAM. We recommend 256 MB or more of RAM
- Minimum 325 MB of disk space for the Rational Rose RealTime installation
- Minimum display 1024 X 768. We recommend 1280 X 1024 or higher
- Postscript printer for printing
- Browser requirement - Internet Explorer 5.01 (SP2) or 5.5 or Netscape Navigator 4.7 or 6.0. We recommend Internet Explorer 5.5

UNIX

The minimum supported configuration for running Rational Rose RealTime on UNIX is:

- Solaris 2.6, Solaris 2.7, Solaris 2.8, or HPUX 10.20
  - For Solaris operation, the minimum workstation is an UltraSparc 10. We recommend an UltraSparc 60 with 512 MB of RAM. We recommend the Solaris 2.8 operating system.
  - For HPUX operation, we support installation of the HP 700 series architecture
  - See the Rational Rose RealTime web site (http://www.rational.com/support) for a list of the required UNIX patches applicable to your operating system.
- Minimum 256 MB of RAM. We recommend 512 MB of RAM with approximately three times this amount in swap space
- Minimum 370 MB of disk space for the Rational Rose RealTime installation

Toolchain Requirements

For Toolchain requirements see the following:

Help Viewer (Windows Only)

The Help Viewer requires that Microsoft Internet Explorer (version 3.02 or later) be configured on a user's computer. It is not necessary for Internet Explorer to be the system's default browser, or that the Internet Explorer icon be visible on the user's desktop.

If you choose not to have Internet Explorer as the default browser, you will need to run Hhupd.exe. This file is the distribution executable that installs the run-time components needed for an HTML Help Project, such as Hh.exe, Hhctrl.ocx, Itss.dll, and Itircl.dll. Hhupd.exe is in the HTML Help Workshop/Redist folder.

Compiler

You must have a C++ compiler installed on your system to make use of the code generation and execution capabilities for Rational Rose RealTime. Different compilers are required for host workstation and for embedded system targets. The list of supported compilers and targets is provided in the *Rational Rose RealTime Installation Guide*.

Real-Time Operating System

If you deploy your model on a real-time operating system, your operating system, hardware, and tool line-up must be one of the referenced configurations listed in the *Rational Rose RealTime Installation Guide*. If you do not have one of these referenced configurations, you may be able to obtain support for your configuration from a Rational RoseLink partner, or by customizing the Rational Rose RealTime Services Library for your target. See the *C++ Reference* or *C Reference* for instructions on customizing the Services Library and compiling for new target configurations.

Note: If there is no RTOS available on the target, or if the application will exist in a single thread, you can use a NoRTOS configuration. This means that you can configure the Rational Rose RealTime run-time libraries to build Rational Rose RealTime applications that run without an operating system. The resulting application that is generated will be a "main" program; you can build and run a main program on the target.

Changes to Referenced Configurations

Table 1 shows the referenced host configurations and targets for Rational Rose RealTime.

Table 1 Host configurations

Toolset Host
Solaris 2.6
Solaris 2.7
Solaris 2.8
Windows NT 4.0 (Service Pack SP6a)
Windows 2000 (Service Packs SP1 and SP2)
Windows XP Pro
HPUX 10.20

Note: If you use a referenced configuration other than that tested by Rational and listed in the topic "*Referenced Configurations and Toolchain Requirements*", in the *Rational Rose RealTime Installation Guide*, standard support will cover problems encountered by customers only to the extent that the problem is reproducible on the configuration listed in the *Rational Rose RealTime Installation Guide*.

Table 2 shows the referenced configurations and targets.

Table 2 Referenced configurations and targets

Host Configuration(s)	Target RTOS	Compiler/Processor	RTS Library	Connexis DCS Library
Solaris	Same	Gnu 2.95.1, SPARC	C++	C++
		Gnu 2.8.1, SPARC	C & C++	C++
		Gnu 2.7.2.3, SPARC	C++	-
		Sun C++ 5.0, SPARC	C++	C++
		Sun C 5.0, SPARC	C	-
HPUX	Same	Gnu 2.8.1, HPPA	C & C++	C++
		HP C++ 10.11, HPPA	C++	-
Windows	Same	Visual C++ 6.0, x86	C & C++	C++
		Visual C++ 7.0, x86	C & C++	-

Table 2 Referenced configurations and targets

Host Configuration(s)	Target RTOS	Compiler/Processor	RTS Library	Connexis DCS Library
Solaris Windows	pSOS 2.5	Diab 4.2b, ppc	C++	C++
Solaris, HPUX	VRTX 4.AB	Microtec 1.3C, ppc	C++	n/a
Windows	VRTX 4.Baa	Microtec 1.4, ppc	C++	-
Solaris, Windows	OSE 4.1.1	Diab 4.3f, ppc GreenHills 1.8.9, ppc GreenHills 2.0, ppc	C & C++ C C	C++ - -
Solaris	OSE 4.1.1 SoftKernel	Gnu 2.95.1, SPARC	C & C++	-
Windows	OSE 4.1.1 SoftKernel	Visual C++ 6.0, x86	C	-
Solaris, Windows, HPUX	Tornado 2.0 (VxWorks 5.4)	Cygnus 2.7.2.960126, M68040 Cygnus 2.7.2.960126 Cygnus 2.7.2.960126, ppc GreenHills 1.8.9, x86c GreenHills 2.0, ppc	C & C++ C & C++ C & C++ C & C++ C & C++	- C++ - C++ C++
Solaris	Tornado 2.0 Sim	Cygnus 2.7.2.960126, SPARC	C++	C++
Windows NT	Tornado 2.0 Sim	egcs 2.90.29, x86	C++	C++
Solaris, Windows	LYNX 3.1.0a	gnupro-2.9-98r2, ppc	C++	C++
Solaris	LYNX 3.0.1	Cygnus 2.7.97r1, x86 Cygnus 2.7.97r1, ppc	C++ C++	C++ C++
Solaris	Chorus Classix 4.0	egcs-2.91.66, ppc	C++	-
Windows	Windows CE sh3	eMbedded Visual C++ 3.0, sh3	C++	C++
Windows	eCos ITRON	gnu 2.95.3, x86	C	-
N/C - native compilation only	AIX 4.2.1	gnu 2.8.1	C++	-
N/C - native compilation only	Nucleus 1.1	Diab 4.2b	C++	-

Table 2 Referenced configurations and targets

Host Configuration(s)	Target RTOS	Compiler/Processor	RTS Library	Connexis DCS Library
N/C - native compilation only	Red Hat Linux 6.1	Egcs 2.91.66	C++	C++
N/C - native compilation only	QNX 4.2.2	Watcom C++ 10.6	C++	-
N/C - native compilation only	UnixWare 7.0.1	SDK 3.0	C++	C++

Licensing Requirements 3

Contents

This chapter is organized as follows:

Please refer to the *Rational Rose RealTime Installation Guide* for instructions on installing a startup license, and obtaining a permanent license.

Note: If you are installing licenses on a UNIX platform, do not follow the instructions on the Rational Start-up License Certificate or on the envelope in which the certificate is delivered to you.

Checking the Validity of Your License Keys

If you upgrade to Rational Rose RealTime 6.4 from Rational Rose RealTime releases 6.0, 6.0.1, or 6.0.2, your license keys are not valid. For information on obtaining new license keys, see "Requesting License Keys" in the *Rational Rose RealTime Installation Guide*.

If you upgrade to Rational Rose RealTime 6.4 from Rational Rose RealTime releases 6.1, 6.1.1, 6.2, or 6.3, your license keys are valid.

For more information on license keys, see the topic "Installing License Keys" in the *Rational Rose RealTime Installation Guide*.

License Usage

A toolset launched manually will require a license.

A second toolset session launched manually on the same platform will require a second license.

A toolset launched as a COM Automation server does **not** require a license unless it is explicitly made visible by setting the RRTEI Application object's **Visible** property to **True**. For more information on RRTEI, see the *Extensibility Interface Reference* in the Rational Rose RealTime online Help.

A toolset launched with a compiled script passed as a command line argument will require a license because the toolset will be made visible before running the script.

A toolset launched with both the **-runScriptAndQuit** and a compiled script passed as a command line argument, will not require a license because the toolset will not be made visible when running the script.

Note: A toolset session, regardless of how it is initiated, will require a license if the toolset is made visible.

Storing Licenses

We recommend that you save your license files in a safe location.

Integration Notes 4

Contents

This chapter describes how to integrate Rational Rose RealTime with other tools. This chapter is organized as follows:

Microsoft Development Environment

We recommend that you install the latest service packs available from Microsoft for Visual Studio or Visual C++.

Configuration for Operation with Rational ClearCase

Rational Rose RealTime 2002.05.20 works with Rational ClearCase 5.0.

For more information on this integration, refer to the *Guide to Team Development, Rational Rose RealTime*.

Integration with Rational Robot

Installing the 2002.05.20.305 release of Rational Rose RealTime will interfere with the operation of the 6.1 release of Rational Robot.

We recommend that you upgrade, at a minimum, to the 2001A.04.00 release of Rational Robot.

Rational ClearCase on a UNIX Server and Clients on Both NT and UNIX

You can access a ClearCase server on UNIX with Rational Rose RealTime clients running on both Windows and UNIX workstations. For more information on integrating these tools, refer to the *Guide to Team Development, Rational Rose RealTime*.

Rational SoDA

Both Rational SoDA and Rational Rose RealTime must be properly installed and licensed before starting. SoDA may be installed either as part of a Suite installation, or as an individual point product. For additional information on the latest updates on SoDA integration, see the following Rational Product Support web page:

http://www.rational.com/support

Note: To generate a report using SoDA, the Rational Rose RealTime model must be saved at least once. If the Rational Rose RealTime model was never saved, it will be untitled. An untitled model causes SoDA to generate errors.

Rational RequisitePro

Both Rational RequisitePro and Rational Rose RealTime must be properly installed and licensed before starting. Rational RequisitePro may be installed either as part of a Suite installation, or as an individual point product. See "RequisitePro" in the *Rational Rose RealTime Installation Guide.*

For the latest updates on Rational RequisitePro integration, see the following Rational Product Support page:

http://www.rational.com/support

Note: The Rational Rose RealTime RequisitePro integration does not support the association of a Rational Rose RealTime package with a Rational RequisitePro project. Use Case and Model association is supported.

Adding Options to Purify on UNIX

The toolset looks for an installation of Rational Purify by checking for an environment variable named **PURE_HOME**. This environment variable is not automatically configured by installing Rational Purify. You must set this environment variable manually. The variable does not have to point to a directory containing Rational Purify, nor is it required to point to a directory. The variable may contain anything, but it must be set.

Note: If Rational Purify is not installed, the **Purify** output window and the **Run with Purify** menu option will continue to appear in the tool. If you select **Run with Purify** when Rational Purify is not installed, there will be a delay, then a message will appear informing you that it was unable to process and it could not find the target.

Occasionally, you may need to add options during a Purify'd build on UNIX. For example, Rational Purify on HP needs to know the name of the linker or collector used by **Gnu g++**.

Options can be added by changing **PURIFY_OPTIONS** in the **CompilationMakeInsert** field of the executable component. The default value of **PURIFY_OPTIONS** (generated in the **makefile** by the code generator) is:

PURIFY_OPTIONS = -log-file=$(BUILD_TARGET).txt -windows=no

To accommodate using **g++** on HP, you can add the following, or similar:

PURIFY_OPTIONS = -log-file=$(BUILD_TARGET).txt -windows=no -collector=/usr/lib/gcc-ld -g++=yes

Where the path of to the collector, **gcc-ld** in most cases, should be the path specific to your environment.

For proper integration of Rational Purify when running the Purify'd executable from the toolset, you should preserve the default options.

For an explanation of Rational Purify options, see *Running a component instance with Purify* in the *Toolset Guide.*

Transition Notes 5

Contents

This chapter is organized as follows:

Migrating from Rational Rose and ObjecTime Developer

To migrate models into Rational Rose RealTime from either Rational Rose or ObjecTime Developer where models were previously stored in a configuration management system, the model must be loaded into Rational Rose or ObjecTime Developer and written out to a single file. For additional information, see the chapter called *Migration* in the *Rational Rose RealTime Installation Guide*.

When importing a model from Rational Rose into Rational Rose RealTime, you are encouraged to resolve any model errors in Rational Rose (**Tools > Check Model**) and fix any unresolved references. In general, Rational Rose is not concerned with unresolved references; however, they are very important in Rational Rose RealTime as they can result in incomplete code generation and compilation errors.

To export an ObjecTime model in a format readable by Rational Rose RealTime, a patch must be applied to the 5.2 or 5.2.1 toolset. This patch will format the model file into a single linear form file with all the required information. The patch is available from Rational Customer Support for the ObjecTime Developer 5.2 and 5.2.1 product releases. For additional information, contact Rational Customer Support.

After the model is imported into Rational Rose RealTime, it can then be stored in a configuration management system.

Using the New Installation

When installing Rational Rose RealTime, review the following:

Installation Process

You will find that the installation of Rational Suite DevelopmentStudio is more involved than the Rational Rose RealTime installation; however, it simplifies the configuration of the Rational Suite and Licensing controls, as well as getting you started quickly with Rational Rose RealTime.

Installation on Windows

Updating Batch Files

If you use a batch file to start Rational Rose RealTime, after you install, you must modify the environment variables to use the new mapped drive. To successfully launch Rational Rose RealTime, you must specify a fully qualified path, including the drive letter. For example, you want to update the ROSERT_HOME variable, as well as the launch command path:

```
set ROSERT_HOME=C:\Program Files\Rational\Rose RealTime

set ROSERT_HOST=win32

set ROSERT_LICENSE_FILE=%ROSERT_HOME%\license\license.dat

set path=%ROSERT_HOME%\bin\%ROSERT_HOST%;c:\Program Files\Microsoft
Visual Studio\Common\VSS\win32;c:\DevStudio\VSS\win32;%PATH%

"C:\Program Files\Rational\Rose RealTime\bin\win32\RoseRT"
```

Additional Settings

For additional environment variables and startup options, see the *Rational Rose RealTime Installation Guide*.

Installation on UNIX

The installation process creates setup files that create all necessary links, mapping, and environment variables for Rational Rose RealTime. These setup files should be added to the client .csh or .sh setup files. Administrators can view the setup files in the following locations after installation:

source <*rational_install_directory*>/config/rosert_setup.csh

or

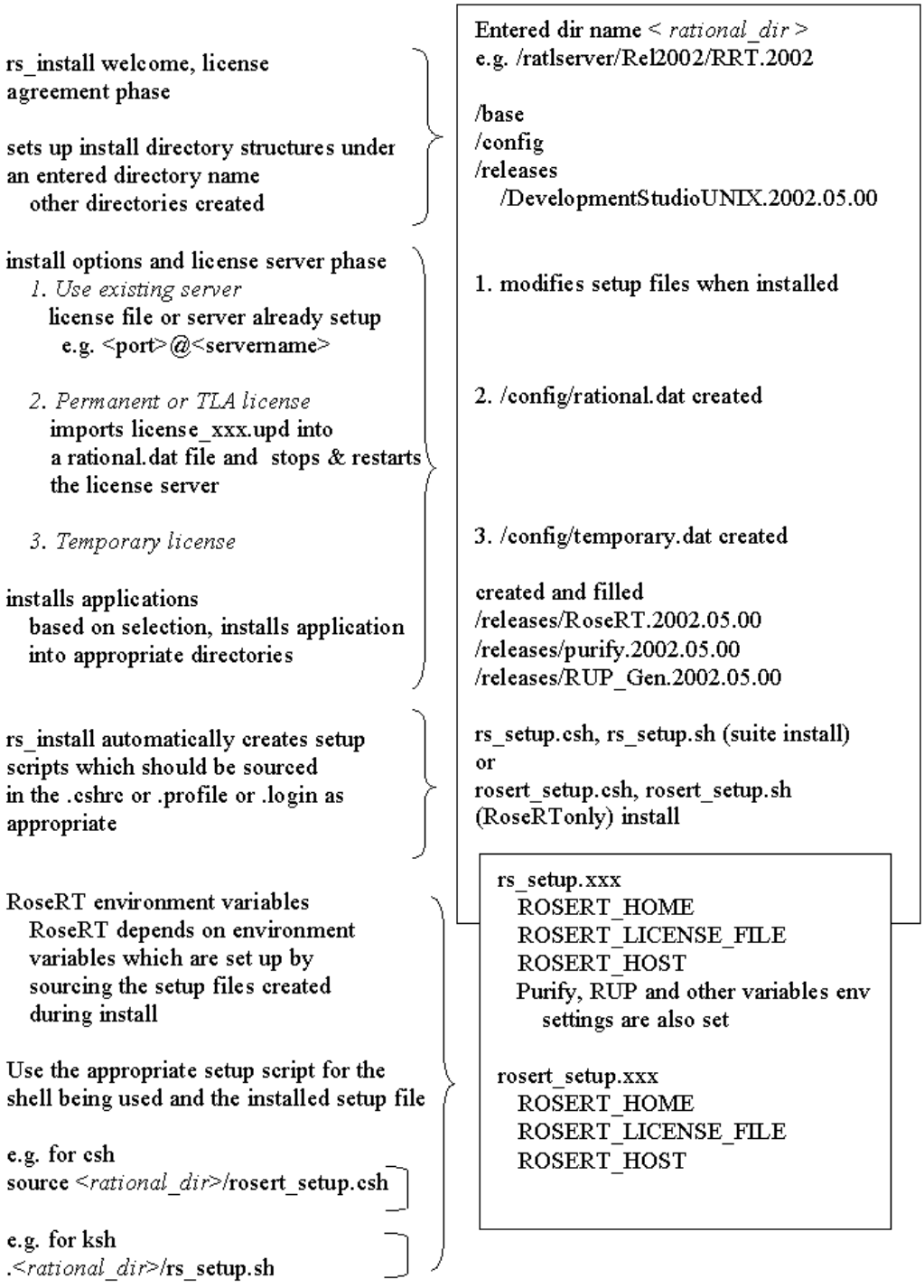
. <*rational_install_directory*>/config/rosert_setup.sh

Note: In the setup files, the install path is a hard-coded path. Typically, Rational products which build databases use hard-coded UNC paths based on installed directories, registry items, or environment variables.

Installation Overview

The following graphic provides an overview of the installation process and shows the installed UNIX directories and files.

Note: Directory and file names are for example purposes only.


rs_install welcome, license agreement phase
sets up install directory structures under an entered directory name
other directories created
Entered dir name < rational_dir >
e.g. /ratlserver/Rel2002/RRT.2002
/base
/config
/releases
/DevelopmentStudioUNIX.2002.05.00
install options and license server phase
1. Use existing server
license file or server already setup
e.g. <port>@<servername>
2. Permanent or TLA license
imports license_xxx.upd into a rational.dat file and stops & restarts the license server
3. Temporary license
installs applications
based on selection, installs application into appropriate directories
1. modifies setup files when installed
2. /config/rational.dat created
3. /config/temporary.dat created
created and filled
/releases/RoseRT.2002.05.00
/releases/purify.2002.05.00
/releases/RUP_Gen.2002.05.00
rs_install automatically creates setup scripts which should be sourced in the .cshrc or .profile or .login as appropriate
rs_setup.csh, rs_setup.sh (suite install)
or
rosert_setup.csh, rosert_setup.sh
(RoseRTonly) install
RoseRT environment variables
RoseRT depends on environment variables which are set up by sourcing the setup files created during install
Use the appropriate setup script for the shell being used and the installed setup file
e.g. for csh
source <rational_dir>/rosert_setup.csh
e.g. for ksh
.<rational_dir>/rs_setup.sh
rs_setup.xxx
ROSERT_HOME
ROSERT_LICENSE_FILE
ROSERT_HOST
Purify, RUP and other variables env settings are also set
rosert_setup.xxx
ROSERT_HOME
ROSERT_LICENSE_FILE
ROSERT_HOST

Licensing

The installation process uses a new version of FLEXlm (7.0). Licensing files and licensing servers are configured as part of the installation process and are stored under the Rational Suite DevelopmentStudio directories.

Installing Source Files

The Rational Suite DevelopmentStudio - RealTime UNIX suite has two types of configurations: HP-UX and Solaris. Rational Rose RealTime installs HP-UX and Solaris executables in to a separate directory. If you have a specific configuration, you must install from the appropriate source. If you have two configurations, we recommend that you install the source to different directories.

Environment Variables for UNIX

With older versions of Rational Rose RealTime, you had to specify environment variables, such as ROSERT_HOME, ROSERT_HOST, and others. Now, you only need to set the LM_LICENSE_FILE variable.

The Rational Suite DevelopmentStudio - RealTime (UNIX) installation process creates two files in <*rational_dir*>, which should be sourced to configure the environment variables for Rational Rose RealTime. Depending on whether you perform a Rational Suite or Point Product installation, the files created are:

Rational Suite:

rs_setup.csh and rs_setup.sh

Point Product:

rosert_setup.csh and rosert_setup.sh

These files automatically configure the Rational Rose RealTime environment and Rational Rose RealTime Suite environment settings.

The rosert_setup.csh or rosert_setup.sh file configures environment variables, including those for the other Rational Suite products. The sourced setup files set the following:

- ROSERT_HOME
- ROSERT_HOST
- Other products' environment variables (SodA, Reqpro, Purify environment variables are set by the setup files according to the setup specified)

You will have to set the environment variable for CONNEXIS_HOME. For additional information, see the *Rational Rose RealTime Installation Guide*.

When starting Rational Rose RealTime, it looks for the $ROSERT_LICENSE_FILE first. If it does not find the files, it then looks for the $LM_LICENSE_FILE.

Upgrading

When upgrading from an older version of Rational Rose RealTime:

- For UNIX, we recommend removing existing setups to minimize possible conflicts between a Rational Rose RealTime version 2001A.04.00 installation and the new Rational Suite DevelopmentStudio - RealTime version 2002.05.20 installation.
- You will need to create a directory for the installation path (<*rational_dir*>).

For additional information on licensing, see the *Licensing* chapter in the *Rational Rose RealTime Installation Guide*.

Upgrading to New Version Only (Deleting Earlier Versions) - UNIX

You can not install Rational Suite DevelopmentStudio - Rose RealTime to the same directory as an older installation of Rational Rose RealTime.

To uninstall an existing version, see the chapter "Uninstalling Rational Rose RealTime" for your specific platform.

Note: Ensure that you make copies of any files added to, or modified in the existing Rational Rose RealTime installation prior to uninstalling. The uninstall process may remove these files.

You can load models created in earlier versions of Rational Rose RealTime directly into 6.4 (also referred to as 2002.05.20). To convert your existing Rational Rose and ObjecTime Developer models, see the topic "Migrating from ObjecTime Developer 5.2/5.2.1" in the *Rational Rose RealTime Installation Guide*.

Note: Do not attempt to load workspaces created in earlier versions of Rational Rose RealTime, as they are not compatible with the new release.

Upgrading to Rational Rose RealTime 2002.05.20 While Maintaining Rational Rose RealTime 2001A.04.00 - UNIX

Your UNIX environment can continue to have a Rational Rose RealTime 2002.05.20 installation and an earlier release of Rational Rose RealTime that uses UNIX environment variables. Refer to the following pseudo code to configure your environment to use both releases of Rational Rose RealTime (.csh or .sh setup):

```
if you want to use an older version
   set up the following environment variables
      ROSERT_HOME
      ROSERT_HOST
   else
      source <rational_dir>/rosert_setup.xxx
         (where .xxx represents .csh or .sh)
   set up the following
      CONNEXIS_HOME to $ROSERT_HOME/Connexis
```

Reinstalling

When re-installing, the user account originally used to install Rational Rose RealTime must be used because there was a local .rs_install.<*name*> file created for the initial installation which retains any install settings. These original install settings are retained in the original user .rs_install.<*name*> file paths.

Known Problems, Limitations, and Updates 6

Contents

This chapter is organized as follows:

Installation and Startup Issues

The installation and startup issues are:

File Association for Compiled Scripts

Windows NT: Rational Rose RealTime does not install a file association for compiled scripts (.ebx). This means that they cannot be automatically run by double-clicking on the file from Windows Explorer.

Suite Objects.dll Not Found

When installing on Windows, or attempting to run Rational Rose RealTime, an error message sometimes occurs indicating that "suite objects.dll" cannot be found in the path.

Adding "C:\Program Files\Rational\common" to the path will fix this problem.

This problem only occurs on Windows NT systems that have had a previous version of Rational Rose RealTime installed.

Installing Rational Rose RealTime after Rational Suite DevelopmentStudio

If you install Rational Suite DevelopmentStudio, and then install the Rational Rose RealTime point product, you may have to select **Add-Ins > Add-In Manager** and check the appropriate boxes to select SoDA and ReqPro.

Start-up Problems

If Ration Rose RealTime has problems starting, look at the task bar and use the Task Manager to see if there are any running copies of Rational Rose RealTime. Terminating all running copies allows new copies of the tool to start properly.

Note: More than one copy of Rational Rose RealTime can run at the same time; however, if startup problems exist, find and terminate any runaway processes.

Uninstall

Occasionally, files are left behind after an uninstall. For example, if a model was saved in one of the Rational Rose RealTime subdirectories, the subdirectory and its parents will not be removed. You must remove these manually if you wish to return your system to a clean state.

General Issues

This topic describes the following issues:

Unique Ids

Unique ids are unique internal names associated with model elements. They are used internally by Rational Rose RealTime, and not all model elements require unique ids. Rational Rose RealTime includes a feature that helps Model Integrator by generating unique ids for those model elements that would otherwise not require them, for internal use. For Model Integrator, an element with a unique id is easier to merge.

For information on **how** and **when** to enable the Unique ids, see "Model Specification" in the online Help.

RRTEI users will find traceability easier when they set this option. Unique ids improve the traceability of model elements of other tool integrations that use RRTEI.

It is necessary to plan and choose when to incorporate the new unique ids into the project model since virtually all controlled units will be modified implicitly. Additionally, the generated new ids are dependent on time and location. For example, generating unique ids for a given model at different times, or on different machines, produces different ids.

Note: We strongly recommend any team involved in parallel development use this option.

Note: Setting this option creates unique ids for model elements that currently do not have them. This typically affects most of the model, so you will be prompted to check out those parts when setting this option.

When saving the model, the size of the affected file increases by approximately 20%, and the time to load the model also increases.

Note: Do not set this option in multiple streams, otherwise, objects with similar characteristics will be treated differently since their unique id's will differ.

For more information about when and how to use this feature, see the *Guide to Team Development for Rational Rose RealTime*.

Note: *If you clear this option, your merge results will not be as reliable*.

Spaces in Directory Names

For Windows, to allow the use of cross-compilers that do not allow spaces in the path names, use the **`subst`** command and map a drive to the value of %ROSERT_HOME% after the installation. For example, if you wish to use the K: drive, and your Rational Rose RealTime installation directory is:

```
ROSERT_HOME=C:\Program Files\Rational\Rose RealTime
```

you must map this directory to the drive by running the following commands from a console window:

```
subst K: "%ROSERT_HOME%"
set ROSERT_HOME=K:
```

Spaces in directory names can cause problems with the following *operating system.compiler library set.development platform*: systems:

OSE411T.ppc603-Diab-4.3f.NT4

VRTX4T.ppc603-Microtec-1.4.NT40

TORNADO2T.ppc630-GreenVX-1.8.9.NT40

TORNADO2T.ppc630-GreenVX-2.0.NT40

TORNADO2T.m68040-cygnus-2.7.2-960126.NT40

TORNADO2T.ppc-cygnus-2.7.2-960126.NT40

TORNADO2T.ppc860-cygnus-2.7.2-960126.NT40

For Tornado configurations, you should also use the substituted drive to point to the load script directory when running Target Observability.

On the detail tab of the processor specification, the `Load Script` path should not contain any spaces. If spaces are present, when attempting to load the executable in Basic or Debugger-Tornado[2] modes, the error message, **Unable to Execute** appears. This does not occur when loading the executable in manual mode.

Spaces in directory names can also cause problems with ClearMake.

Too Many Files Open

You can have a maximum of 1024 file pointers open simultaneously.

Exceptions When Using Configuration Management in Rational Rose RealTime

If you encounter an exception when working with models under source control on UNIX, your /var/tmp is likely full.

Executing Component Instances

If you receive the message "Unable to connect to target" when attempting to connect to a target (both host and embedded), change the value for the **Connect Delay** box on the Component Instance specification by increasing the value by two or more seconds.

Sequence Diagrams

There are a few conditions under which the Sequence Diagram will incorrectly draw Messages / FOCs. If a Message or FOC appears to be incorrectly drawn, select the Message (or the Message that starts the FOC) and, using the center 're-orient' handle, slightly move the Message. This will force the Diagram to recalculate the correct display values for that message.

Build Dependencies on Case-Insensitive File Systems (NT only)

During a build, Rational Rose RealTime detects and records build dependencies for comparison during subsequent builds; this is done to facilitate build-avoidance by only regenerating or recompiling targets when a build dependency changes. These build dependencies preserve the case of the filenames involved, even when the underlying file-system (for example, NTFS) is case-insensitive. This may cause problems when using names which are distinct within the toolset, and distinct on case-sensitive drives, but indistinct on case-insensitive drives. In most cases, the toolset or code-generator will identify and avoid or warn against potential case-insensitive name collisions.

However, some case-insensitive filename collisions cannot be detected. For example, if a component is renamed to something that is case-insensitive-identical (for example, renaming from "myComponent" to "MyComponent"), a build may incorrectly reuse all previous build results, since the underlying build dependencies will be indistinct according to the file-system.

Users are advised to use case-sensitive file-systems if possible. Otherwise, users should avoid case-insensitive name collisions when they create or rename classes, components or controllable units.

Using the Debugger-xxgdb Tool and Running your Component Instance

When using the xxgdb tool, run your component instance, add any breakpoints, and then restart it to enable the breakpoints in **gdb**.

Use of C and C++ Add-ins

You cannot use C and C++ Add-ins at the same time in Rational Rose RealTime.

Symbolic Links with TargetRTS

When using LynxOS 3.1.0, do not install Visual Lynx 3.1.0 for Windows NT on a network (NFS) disk. It should only be installed on a local NTFS drive; otherwise, symbolic links to some **include** directories will not work properly. This will cause compilation errors if you re-compile the TargetRTS.

If you have installed Visual Lynx 3.1.0 on a network disk, and if you see compilation errors stating that **include** directories netinet/in.h or net/if.h are not found, locate the entries **net** and **netinet** in the following directory:

$LYNX_HOME/usr/lynx/3.1.0/ppc/usr/include

If these entries are text files containing the following text: **!<symlink>bsd**

rename these files and create new symbolic links called '**net**' and '**netinet**' that both point to the directory bsd.

Virus Scanning Applications Affect Startup and Shutdown

On startup or shutdown, virus scanning applications, such as McAfee VirusScan and Trend, will scan the RoseRT.ini file many times, causing a very slow startup or shutdown. To enable a much faster startup and shutdown, exclude the INI extension from the Program File Extensions list in your virus scanning software.

Code Generator Runs Out of Memory When Generating Very Large Models

When you attempt to generate a very large model, the code generator may run out of memory. To improve memory usage when generating very large models, we recommend that you control all the controllable elements in your model as individual units.

To control elements in your model:

1. In the **Model View** tab in the browser, select the model.
2. Right-click and click either **File > Control Units** or **File > Control Child Units**.

3 When prompted to control all child units recursively, click **Yes**. Also click **Yes** on the subsequent confirmation dialog.

For details on controlled units, see the topic, "What is a controllable Element?" in the *Guide to Team Development, Rational Rose RealTime*.

Window Order Policy

When using the CDE window manager, to ensure that the proper Secondary and Transient window policy is in effect, in .**Xdefaults**, set the following environment variable:

Dtwm*secondariesOnTop: True

Setting this variable to True ensures that an opened secondary window in Rational Rose RealTime for UNIX, such as an external editing window, does not get caught behind the main primary window. Since the secondary window is the active window, you may be unable to regain focus of this secondary window.

When using CDE as your XWindow manager, the **Allow Primary Windows on Top** and **Raise Windows When Made Active** options are enabled by default. These options should be disabled when setting the **Dtwm*secondariesOnTop** option to True.

Running Model Examples

To compile some of the example models installed with Rational Rose RealTime, such as the **SocketInterfaceExample**, you must have the Rational Rose RealTime Companion Products installed.

Using Get Method and Set Method in the Attribute and Operation Tools

In the **Attribute** and **Operations** tools, you can de-select the **Get method** and **Set method** boxes. If you want to prevent deletion of modified code for the Get and Set methods, do the following:

1 Open the specification for an **Attribute** or **Operation**.

2 Click the **General** tab.

3 In the **Documentation** box, delete the following text:

//GENERATED BY ATTRIBUTE TOOL

or

//GENERATED BY OPERATION TOOL

Deleting this text prevents deletion of the modifications you made to the Get or Set methods if you de-select **Get method** and **Set method**.

Using the Frameworks Dialog

If you create a framework whose name comes alphabetically before RTC and RTC++ and the Frameworks dialog is displayed, the highlighted icon is the RTC++ icon, but the text in the **Description** tab is for the RTC framework. If you click **Open**, the RTC framework is loaded.

Workaround: Always use your mouse to select the framework to open before clicking **Open**.

Scoping Descriptors for Nested Classes

Problem:

If you have the following externally defined classes:

```
class ExternalClassA
{
    public:
    int x;
    class ExternalClassB
    {
        public:
        int y;
    }
}
```

and model them in Rational Rose RealTime by **ExternalClassA** and a nested **ExternalClassB**. For both, **generateClass** is not selected and **generateDescriptor** is selected. The type descriptor for **ExternalClassB** is not generated.

Description:

The descriptor for the nested class must be scoped within the top-level class because the nested class is not necessarily public. Because the top-level class is not generated, the descriptor for the nested class (which must be part of the declaration of top-level class), cannot be generated.

No Codesync Support for Java

Although codesync options may appear in the menus, Rational Rose RealTime does not support codesync for Java.

Using the GetSelected Functions

In Rational Rose RealTime, the browser selections only apply when the browser window has focus (Rational Rose RealTime uses a different definition of what is selected than Rational Rose). In all other cases, the selection is from the active diagram window (if any). When running a script, you are in the script window; the browser window does not have focus. To return focus to the browser window, you can add a delay to your script.

The GetSelected functions (for example, **RoseRTApp.CurrentModel.GetSelectedClasses**) only find the selected elements in the model browser if the model browser is active. This means that if you select elements in the browser, then run your script (from the script editor, or the tools menu), the model browser is no longer active (the selected elements are grey, not blue) and these functions will not find anything.

For example, if you open the following script in Rational Rose 2000, select your classes in the model browser, then run the script, it will work.

```
Sub Main ()
   Dim theSelectedClasses As ClassCollection

   Viewport.Open
   Viewport.Clear

   Print "Selected classes from model browser:"
   Set theSelectedClasses = RoseApp.CurrentModel.GetSelectedClasses()
   If theSelectedClasses.Count > 0  Then
      For i = 1 To theSelectedClasses.Count
         Print "Class name: "; theSelectedClasses.GetAt(i).Name
      Next i
   Else
      Print " No classes have been selected "
   End If
End Sub
```

However, this script will not work in Rational Rose RealTime. It will only work in Rational Rose RealTime if you make the model browser active. For example, if you add the sleep line to the following script, run the script, select the classes in the model browser, and wait 5 seconds, it will find the selected classes.

```
Sub Main ()
   Dim theSelectedClasses As RoseRT.ClassCollection

   Viewport.Open
   Viewport.Clear

   Sleep(5000)
   Print "Selected classes from model browser:"
   Set theSelectedClasses =
RoseRTApp.CurrentModel.GetSelectedClasses()
   If theSelectedClasses.Count > 0  Then
      For i = 1 To theSelectedClasses.Count
         Print "Class name: "; theSelectedClasses.GetAt(i).Name
      Next i
   Else
      Print " No classes have been selected "
   End If
End Sub
```

Specifying a Location or File Name Containing Spaces (UNIX)

To properly process a location or file name containing one or more spaces, the command line must be properly quoted. You need two levels of quotes: the first set of quotes (') wraps the second (") so that the RoseRT script will not interpret the space character; it will pass the file name with space(s) as a single argument to the RoseRT executable.

For example, given the following path and file name:

Test with Spaces/Model with spaces.rtmdl

you would invoke the toolset using the following:

RoseRT '"Test with Spaces/Model with spaces.rtmdl"'

The parameter is quoted as follows:
<open single quote><open double quote> path/filename<close double quote><close single quote>

Note: You must use the quotes as described. If double quotes are used to wrap the single quotes, the contained string is processed as a command and it will produce errors.

Limits on the Number of Open Windows

GDI handles are required to create graphic objects such as windows, menus, cursors, bitmaps. Opening windows consumes handles. Therefore, we recommend that you open a reasonable number of windows; on Windows platforms do not to exceed 200 open windows at a given time.

Setting the Stack Space Limit

Some operations with large models may require a value larger than 32MB to be set manually. If your UNIX administrator set a hard limit on the stack size, you can set a higher limit by using the limit command in **csh,** or the **ulimit** command in **sh** or **ksh**. Some operations with large models may require a value larger than 32MB to be set manually.

Limitations in the Specification History List

The following limitations apply to items in the Specification History list:

- if an RTS object is "locked", it will not be loaded in to the Specification History windoes with the workspace next time model is opened
- RTS objects are visible in the Specification History window after RTS shutdown. However, if you attempt to open this object with no RTS running, the object is removed from the list (this also occurs when using the **Refresh** menu item)

Integrations Issues

This topic describes the following issues:

Error Loading Large Models from ClearCase - Windows

If you have problems loading large models from ClearCase - that is, it takes a long time or an empty model appears - check your error messages. It may be because there are too many files open.

To resolve this problem:

1. Go to the Control Panel, and double-click on the **ClearCase** applet.

   The **ClearCase Properties** dialog appears.

2. Click the **MVFS Performance** tab.

3. Under **Mnodes**, click the **Override** boxes for both **Maximun number of mnodes to keep on the free list** and **Maximum number of mnodes to keep for cleartext free list**, and set the value for both to **800**.

4. Click **OK** to apply the changes and close the dialog.

5. Reboot the Windows computer.

RequisitePro

When the Rational RequisitePro add-in is enabled in Rational Rose RealTime, there is a known problem with **File > Save As**.

There will be numerous prompts similar to the following:

as RequisitePro attempts to save the units of RTClasses and RTCClasses. The number of prompts will be less if only one of C or C++ is enabled. Clicking **OK** to all the prompts causes no harm. Other possible workarounds are disabling the RequisitePro add-in for the **Save As** (ensure the model is saved before disabling) or removing the RT packages if they are not required at this time.

For the latest updates on RequisitePro integration, see the product support page at:

http://www.rational.com/support

SoDA

SoDA can be installed without Rational Rose 2001 and will continue to function with Rational Rose RealTime. However, when run from Rational Rose RealTime, the menu of templates presented are from Rational Rose, not Rational Rose RealTime. Some of these templates will issue errors.

Please refer to the product support page at:

http://www.rational.com/support

for the latest updates on SoDA integration.

Note: To install the Rational Rose RealTime domain for SoDA, you need to select the Full/Customer install option during your SoDA installation.

Online Help Issues

This topic describes the following issues:

Problems Accessing Rational Rose Help when Concurrently Running Rational Rose RealTime on Windows

Problem

Running both Rational Rose RealTime and Rational Rose may cause Rational Rose not to access its help files.

Explanation

To enable the help for the Rational Rose add-ins to be accessible when used from within the Rational Rose RealTime toolset, Rational Rose RealTime temporarily replaces a registry key value installed by Rational Rose. This registry key is located in [HKEY_LOCAL_MACHINE\Software\Rational Software\Rose] and is named "HelpFileDir".

When the toolset starts, it substitutes this key's value for its own Help directory. The value installed by the Rational Rose installation program is backed up and restored when the last instance of the running toolset exits.

Workaround

You can disable this behavior by setting the value of the key located in [HKEY_LOCAL_MACHINE\Software\Rational Software\Rose RealTime\6.4] whose name is "ReplaceRoseHelpDir" to "No".

Navigating through the Online Help

You can either select a topic in the **Contents** tab or click the **Next** or **Previous** buttons, and the topic appears in the main window.

Note: If you click **Previous**, although the previous topic appears in the main window, the incorrect topic may be highlighted in the **Contents** tab. However, this does not affect the navigating to the selected topic.

Viewing Animated Demonstrations

Rational Rose RealTime includes animated demonstrations of various topics.

To see the current list of animated demonstrations, from the **Help** menu, click **Contents**. The Rational Rose RealTime Online Help Start Page appears. Select Animated Demonstrations for your specific platform.

Viewlets can be viewed on a number of platforms; however, your browser must have Java 1.1 and JavaScript support to play a Viewlet. Additionally, Java must be enabled in your browser.

Note: It is normal for some larger Viewlets to take 10 to 15 seconds to load. Ensure that your browser cache is not disabled; otherwise, some Viewlets may take longer to load.

If launching a Viewlet takes a very long time, or a gray box displays instead of running a Viewlet, try running the Viewlet in your browser by doing the following:

1. In your browser, navigate the Rose RealTime /Help directory.
2. Select WatchAnimatedDemos.htm.
3. Select a Viewlet to run that demonstration.

Windows 2000 Issues

You will encounter problems running Rational Rose RealTime 2002.05.00 on Windows 2000 if you log on as a user without Administrator privileges. The types of problems include the following:

- When opening Rational Rose RealTime, you may see the error message, "Failed to Update the System Registry. Try using REGEDIT."
- The configuration of the Add-in manager cannot be restored.
- The two menu items **Add Class Dependencies** and **Component Wizard** are missing from the **Build** menu.

- When creating a component, you are not able to define the **Environment in Component Specification** for the **C++ TargetRTS**.
- When selecting **Rebuild** from the **Build** menu, or clicking the **Build** tool from the Toolbar, there is no activity.
- After selecting **Run** from the **Build** menu, then clicking **Yes** to select **Build the component**, you will receive the error message **Operation not allowed**.
- When clicking **Help > About**, there is no **Version** or **Company** information.
- You will not be allowed to set a Top Capsule in the component specification.

Note: To run Rational Rose RealTime on Windows 2000, you must log on with Administrative or Power User privileges.

UNIX Issues

This topic describes the following issues:

Executing the Current Version with a Previous Version

The current and previous toolsets do not work together by default, due to sharing the *MWRPC_ENDPOINT* and *MWSHM_KEY* environment variable settings. These settings are only calculated if the environment does not already have them set.

To execute a current and a previous toolset from the same host or shell:

1. Invoke the first toolset with the `-env` parameter (and `&` to have it run in the background)
2. Find the *MWRPC_ENDPOINT* environment variable and note the value, for example, `10633`

3 Set the *MWRPC_ENDPOINT* environment variable for the second toolset to be one greater than the previous:

```
setenv MWRPC_ENDPOINT 10634
```

MWRPC_ENDPOINT values must be between 10000 and 65000 to be valid.

4 Set the MWSHM_KEY to the same value as the MWRPC_ENDPOINT

```
setenv MWSHM_KEY 10634
```

MWSHM_KEY environment variable values must be integers less than 1048576, to be valid. We recommend that you use the RPC value for convenience.

5 Invoke the second toolset with the **`-env`** parameter to confirm environment settings.

Refresh Problems with Exceed

The screen sometimes does not refresh completely when running the UNIX version of Rational Rose RealTime and displaying it on a PC using Hummingbird Exceed.

To changing your Exceed settings:

1 Start Xconfig.

2 Open the **Performance** dialog.

3 Use the following settings:

Save Unders = No

Maximum Backing Store = When Mapped

Default Backing Store = None

Minimum Backing Store = None.

HP-UX Make Dependency Filename Restriction

On the HP-UX platform, the HP-UX Make restricts dependency filenames to 160 characters. We recommend the use of Gnu-Make to work around this problem. Alternatively, you may shorten the path names to your class files to avoid this issue.

Non-GUI-based External Editors

On UNIX, the toolset will freeze when the user specifies "/bin/vi" as the external editor and then tries to launch the external editor from the code edit pane. If you are using a non-GUI-based external editor, this should be specified using “xterm -e /bin/vi” so that the editor has a terminal (tty) to display to.

Case Sensitivity within Paths

The UNIX temporary directory name is translated to all lower case. If you set the environment variable TEMP, make sure the path name is all lower case or the directory will not be found. This will cause problems when Help is started.

Toolset Freezes on Startup

Under certain configurations, when starting Rational Rose RealTime on Solaris and displaying to an exceed X-server, the "Welcome to ..." window appears momentarily, then disappears. After this, the main application GUI is locked out and freezes.

The workaround is as follows:

1. Kill the Rational Rose RealTime process.
2. Edit the **RoseRT.ini** file in the **~/.registry.2002.05.00** directory and change **ShowStartupDialog=Yes** to **ShowStartupDialog=No.**
3. Run **RoseRT -cleanup**.
4. Run **RoseRT**.

Troubleshooting Toolset Freezes on UNIX

Occasionally the Rational Rose RealTime toolset freezes (for example, it does not respond to user input, or the window will not refresh). We have been unable to fully characterize these problems.

If you encounter a toolset freeze, please complete the following steps:

1. If on Solaris, run: /usr/proc/bin/pstack [Rose RealTime pid] > [outfile].
2. If on Solaris, run: /bin/truss -o [output file] -p [Rose RealTime pid] for 15 seconds.

3. Kill the Rational Rose RealTime process:
   a. run: kill -SEGV [Rose RealTime pid]
   b. You will be prompted:
      i. for information on how to reproduce this problem
      ii. to run a resource cleanup utility on your session
      iii. to indicate whether you want to be contacted by Rational Customer Service.

      Note: If you answer yes to "Would you like to be contacted by Rational Customer Service", a Rational Customer Support engineer will contact you shortly. You will be asked to provide the information you have gathered in steps 1 through 6.
4. Take note of your Rose RealTime build number. You can obtain this information by going to the Help menu and pulling down to "About Rational Rose RealTime…".
5. Attempt to reproduce the problem:
   a. If you can reproduce the steps that caused the freeze, take note of these steps.
   b. If you are unable to reproduce the steps that caused freeze, any information you can provide on what you were doing at the time of the freeze will be helpful to our investigation.
6. If you are using ClearCase, were you in a ClearCase view at the time of the freeze and what version of ClearCase are you using?
7. Please forward all of the above information to Rational Customer Services (for contact information, see Contacting Rational Customer Service by Email or Telephone). Ensure that you include "Rose RealTime" in the subject of your email.

Online Help

Cannot Open Some Links in the Online Help

Some links to multimedia content do not work on UNIX. Where possible, an alternative method for opening certain files is included.

Maintaining a Single Favorites List

The Rational Rose RealTime online help is modularized. Consequently, if you select **Add** on the **Favorites** tab to add current help topic to your favorites list, this entry will only appear in the favorites list for that component of the online help.

To maintain a single list of "favorite" help topics:

1. Use the **Search** or **Index** tabs to find the desired online help topic.
2. Click the **Locate** button to see where this help topic appears in the **Contents** tab.
3. Close the online help.
4. Open the online help, using **Help > Contents**.
5. From the **Contents** tab only, find the help topic.
6. Click the **Favorites** tab.
7. Click **Add**.

Error Occurs When Printing a Diagram

If you use an 8 bit, 256 color terminal, you may receive the following error message when you print a diagram from your model if the graphics use the default gray shading:

"Error Writing to print file <file_name>."

To print diagrams within your models, you must redefine the default fill color. To redefine your fill color:

1. From the **Tools** menu, select **Options**.
2. Click the **Font/Color** tab.
3. Click **Fill Color**.
4. Select another color other than the default gray.

Updates

Rational Rose RealTime includes the following updates:

Code Generator Updates

Description

The code generator will issue a warning message when the action code for an inherited transition is duplicated in child capsule code.

Background

The action of a transition or expression of a choice point becomes the body of a C++ operation with two parameters, **rtdata** and **rtport**.

For example:

```
void MyCapsule_Actor::transition2_t1( const MyData * rtdata,
                                      MyProtocol::Base * rtport )
{
// transition action code
}
```

rtdata: The **rtdata** parameter points to the data in a message. It is a pointer to the lowest common superclass of the possible data classes in all the signals (regardless of the port) that trigger the transition. The argument type must be the most common base class, or **void** * if there is none.

rtport: This represents the port on which the current message was received. It is a pointer to the common base class of all protocol roles that trigger a transition.

For example, if we have two triggers for a transition, one with portA (ProtocolA - Base) and one based on portB (ProtocolB), the type for **rtport** would be the lowest common denominator in the protocol class hierarchy, which if ProtocolA and ProtocolB were not related by inheritance would be **RTProtocol** * (the base class for the **TargetRTS**).

Result

When a transition is inherited from its parent and the action code is not overridden, in most cases, the transition function is not regenerated in the derived (child) capsule's code.

In certain cases, the transition function will be regenerated in the child capsule's code. Additional triggers can introduce new data types or protocol roles to what must be supported by the action code. The result is that the **rtdata** and **rtport** parameters may need to be more general (higher in the class inheritance hierarchy). When the

transition function is regenerated with this different signature, your code may no longer compile. The code generator will issue a warning if the inherited action code is not overridden and the context is more general. The warning is:

- For a transition:

  "A duplicate function is generated for this action because its triggers are incompatible with the inherited function."

- For a choice point:

  "A duplicate function is generated for this choice point because its triggers are incompatible with the inherited function."

For example, if the base capsule class has a transition triggered by a signal which conveys data of type **RTString**, the transition in the base class has the formal argument:

```
const RTString * rtdata
```

Suppose a derived capsule class modifies that transition so that it is also triggered by another signal which conveys data of type **RTInteger**, and the action code is inherited. If the function generated for the base class is reused in the derived class, you have a problem: the action code may depend on the data being of type **RTString**, but we cannot be sure of this type. In this example, the action is generated as the body of a new function in the derived class, and the warning is displayed.

Even if your code compiles, you may have a problem at run-time. For example, if CapsuleA contains an inherited transition from its parent which calls its parent's transition code (using SUPERMETHOD).

CapsuleB is a subclass of CapsuleA and adds a trigger to this transition. If the transition code needed to be regenerated in CapsuleB, then this would lead to the transition code in CapsuleA being called twice.

Prompting for Information When an Exception Occurs

On Windows hosted versions of Rational Rose RealTime, a dialog appears after an exception occurs allowing you to specify information about the steps leading up to the exception, your current system configuration, and whether you would like to be contacted.

On Unix hosted versions of Rational Rose RealTime, a script is executed after an exception occurs, which has the same functionality as that provided by Windows hosted versions. Whether you specify the requested information or opt not to specify

any information, the script runs **RoseRT -cleanup**. After any Rational Rose RealTime UNIX exception, you must run **RoseRT -cleanup** to ensure that any remaining crash artifacts are removed.

Note: On UNIX hosted versions, if you do not want to be prompted for information after exceptions, set an environment variable **ROSERT_NO_FEEDBACK** to **True**. Even when this variable is set, **RoseRT -cleanup** will run after an exception. Later, if you want to be prompted to provide feedback, unset the **ROSERT_NO_FEEDBACK** variable.

Classes for RTJava applications in a Single .jar File

Classes for **RTJava** applications are now bundled into a single .jar file rather than a directory. You may need to modify **CLASSPATH** in your environment, as well as other locations where Java code is compiled or run outside the toolset.

rtsetup.pl File

There is a new Perl script, called rtsetup.pl, which you can use to recreate the environment used to build the TargetRTS. The primary usage will be in the **CompilationMakeCommand**: you want to prefix the existing command with the following:

rtperl -S rtsetup.pl

For example, if the command was

$defaultMakeCommand

it becomes

rtperl -S rtsetup.pl $defaultMakeCommand

The script examines the generated makefile to determine the **TargetConfiguration** specified for the current component, executes $RTS_HOME/config/<TargetConfiguration>/setup.pl to recreate the environment required to build the TargetRTS, and then, in that environment, executes the command that follows.

It is useful for designers who must build components for different targets which depend on conflicting environment variable definitions.

RRTEI Updates

New Public Method GetLinkedDiagram()

Description

GetLinkedDiagram() is a new method for the RRTEI **NoteView** class. It retrieves a diagram linked to a note.

Syntax

```
theReturn = theNoteView.GetLinkedDiagram ( )

theReturn As RoseRT.Diagram
```

Returns NULL (Nothing) if there is no diagram linked to the note.

```
theNoteView As RoseRT.NoteView
```

Instance of the **NoteView** whose linked diagram is being retrieved.

Using the TargetRTS Configuration for the VS.net C++ compiler

The TargetRTS configuration for the VS.net C++ compiler is called **NT40T.x86-VisualC++-7.0** for both C and C++ RTSs. You can select the configuration as any other in the **TargetConfiguration** box of the Compilation (**C** or **C++**) tab of a Component Specification. To build the models your environment requires, refer to the file in vc7/bin/vcvars32.bat for your VS.net installation. If you are using rtsetup.pl, the environment will be configured for you from the registry, but **nmake** will have to be in the path.

The debugger integration with VS.net can be accessed by selecting **Debugger-VS_net** for **Operation mode** on the **Details** tab of a Component Instance Specification.

Building and debugging with VS.net works only on Windows.

Problems Addressed

The following customer originated cases have been addressed in this release:

Defect Number	Description
41991	We need more meaningful message when make is not set a component
46240	default attribute visibility shouldn't be 'private'
47309	need a reasonable error message if I try to build an external library component

Defect Number	Description
48292	Warning should be generated if dependency doesn't exist for an attribute of a RTT
48726	Fix the way external docs and URLs are handled on Unix
48874	Forward reference the type descriptor before class definition
49818	C product: flags are not initialized in TCB of the Timer capsule.
49829	Nested pathmaps don't work when building a component
50476	UNIX Exception: MenuItemsFromRes/xdr_string (starting two sessions at same time) [
50606	Cannot derive from nested base class.
50867	Codegen bug with nested classes
50934	RTJava: Building RTJava External Project gives meaningless error message
51054	Java: rtdata type should be class specified on signal, not Object.
51165	Misleading error message for data class problem when building
51240	C language: subclass (struct) needs an attribute in order to be generated
53909	Build errors when CLASSPATH has a trailing '\'
53938	rtcppgen application error when capsule role reference contains incorrect fully qualified
53942	Usability: User interface not refreshed (hangs) while Purify instruments model
53970	1cPMbcsFromUnicode6FpcipkwiInIUN_FLAGS__i_
54911	Error Parsing
55206	Compiler traceback incorrect if multiple classes have the same name
55238	Minor problem in C++\TargetRTS\libset\sh3-eMVisualC++-3.0\libset.mk "/MD" used
56559	$ROSERT_HOME/bin/s/cc/vt vtadmin & vtsetview need execute permission
58957	corrupt .rtusr and RoseRT.ini files
60884	CRTS leaks large data in timeouts
61194	rtBound transition occurs before the initial transition
61341	chdir_run.pl needs double quotes around paths with spaces

Defect Number	Description
61496	Codegen bug with junction points
62199	RoseRT crash Move attribute browser window, 1st class to 2nd class (drag-and-drop
62761	Problems with ranked publishers (locator_rank)
63464	Capsules gets deleted from the model when using DEL.on a diagram
64057	Crash when undoing delete of package, then file new
64426	RTJava: Not possible to move classes between packages
66541	RQA: no matching instance found for instance
66868	RoseRT: Second Zoom window outside screen
66883	Rosert: All text on transition not visible when typing
67039	Codesync: can't bring in Constuctor Initializer list
67390	RTJava: Can't run with non-default OutputClassDir
67656	RoseRT takes up 99% of CPU after start-up (no model loaded)
68110	OTD to RRT conversion problem with non-exiting transitions
68446	Undefined symbol delete_RTEventInfo when C RTS recompiled with RTS_CLEANUP_MECHAN
68696	C Runtime View: step command does not work properly
69092	PATCH REQUEST V6.4: TimerControlBlock in C runtime fails to compile if RTS_COMPAT
69249	Ctrl-B on detail tab of Logical view package crashes toolset
69417	Error in RTMemoryUtil_new
69834	Rose RT: C- Codegenerator aborts
69939	C Target RTS: not all functions removed when RTS_CLEANUP_MECHANISM 0
70060	Unchecking "GenerateConstructFunction" in options (F12) destroys timers in C
70270	Receiving Unresolved Reference error after doing package rearrangement
70525	Moving sequence diagram changes name
70865	Exception when exporting Use Case View to Rose ptl file

Defect Number	Description
70956	Prvide a NoRTOS configuration for the C TargetRTS
71063	The first capsule after startup woll not be created
71086	can't change the order in which nested classes are generated
71088	RoseRT generated app crashes when attaching probes
71115	rtcppgen.exe "Application Error" when building component for C++ model example
71209	Rose RealTime: Source Control ->Check-in Window
71700	Toolset cannot set a breakpoint when debugging with TC.
71727	Connexis registerSAP causing locked mutex in RTLayerConnector
71854	TimerTest Example model: Test Case #NonPeriodicTimers failure:
72035	RQA-RT: signals sometimes missing in generated sequence diagram
72412	Fix comment block size in converted OTD-FD to small
73220	Spelling mistake in dialog: Unable to obtain a license.ealTime for Unix 6.4
73529	001B:0056E7F2 RoseRT.exe - crash when importing a rose .cat file with a sequence d
73708	probes with identical names are lost
74072	Stack depth differs between *.ebs and *.ebx scripts
74076	RoseRT trace probe causes "Target Failed" error
74194	CLONE OF: Host Platform textbox lacks version details in the Email TS wizard
74257	CM_* IsSnapshotView() too vague - Should look for "View attributes: snapshot"
74339	RQART-Java: Cannot resolve symbol error when building a Java Model
74362	RTDebugger AddEvent crash in targetRTS if inject probe on model
74452	Error creating new model when current model has stored probe/monitor
74734	Problems running Verify Behavior with 80 or more Seguence diagrams, application c
74773	RQA-RT backslashes stripped out of RTStrings in messages

Defect Number	Description
74923	TO continue/shutdown dialogue causes crash if model has changed significantly
75632	Internal Error (Code Generator): mi != _capsuleRoleMap.end()
75795	Change Capsule Role - EXCEPTION_ACCESS_VIOLATION in module RoseRT.exe at 001B:008
75803	Internal Error (Code Generator): Conversion error during translation.
75846	Drag, Drop on Operations Tab causes crash
76209	new c.pty file from GA-00 does not dirty the C model on load
76427	Application Error occurs because of a composite aggregation / migrating from Rose
76526	RoseRT chrashes when syncronizing with source control. (operation spec open)
76670	C Code-generator misses #ifdef RTS_NAMES in class
76715	Bad field descriptor for array
77301	C timers not working properly, all capsules on same thread
78067	From the Help menu, click Keyboard shortcuts...only displays help start page
78243	RTCClasses does not include the RTTimerId class in a C model
78474	Inconsistency of how the TargetRTS allocates memory for RTSequenceOf
78730	RoseRT will not generate capsule code based on the package name
78782	Crash creating sequence diagram off of a trace
78881	RQART modal dialog isn't modal when source control interferes
79279	Enable TO for single threaded C Targets
79299	crash opening inherited port from connector dialog
79494	Infinite recursion possible in RTDiag::panic
79581	The frame.import function does not return if a port cardinality is exceeded
79650	crash loading model after TO session with port probe trace window open
79931	Codgen bug with RunTimeBackwardsCompatible for protocol turned on
79938	Crash when copy many nested diagram views

Defect Number	Description
80193	Purify integration with RoseRT broken
80449	RoseRT->Run with Purify doesn't function correctly if build path has space
80750	RoseRT & Purify integration appears not to work when dir path has a period
80761	Codegen bug in C version when using inheritance with class scoped attributes
81200	Remove special handling for ExternalLayerThread
81258	Sequence Diagram Painting slow
81597	RoseRT Crash after unsuccessfully connecting to target
82489	quids updated on transition code after every change.
83004	Toolset forces attribute names to be unique among both local and inherited attributes

Documentation Updates 7

For the most recent documentation updates, please visit the Product Support section of the Rational Rose RealTime web site at:

http://www.rational.com/support

Where printed documentation exists, always refer to the online Help or PDF versions of the book for the latest updates.

From the main Rational Rose RealTime online Help page, the links to the *User's Guide, Rational Connexis* and the *User's Guide, Rational Quality Architect* are only available after you install the Companion CD that contains these products.

Technical Support 8

Contents

This chapter is organized as follows:

This chapter describes how to submit problem reports, feature requests and support requests to Rational Customer Service.

Submitting Problem Reports

With Rational Rose RealTime, you can email problem reports to the Rational Software Customer Service department that services your location. When you email a problem report directly from the Rational Rose RealTime application, a wizard guides you through the process, ensuring that you provide the correct information to the Rational Customer Support team. This information includes contact and location information, and a detailed description of the problem that you are reporting.

To submit a problem report:

1. From the **Help** menu, click **Email Technical Support**.

   A submenu appears, providing you with three options.

2. Click **Problem Report**.

   The **General Information** dialog appears.

3. Type your contact and location information in the text areas provided and click **Next**.

   The **Problem Report - Additional Information** dialog appears.

4. In the **Defect Title** text area, type a detailed name for the problem that your are reporting.

5. Select the type of problem that you are reporting from the appropriate list boxes.

6 Describe the problem, using the categories provided in the **Details** area.

7 Click **Next**.

The **Email Summary** dialog appears.

8 Ensure that the information that appears in the Email Summary dialog is accurate.

Note: The email information displayed in the Technical Support Email Address is chosen based on the location information that you provided in the General Information dialog. It is not recommended that you edit the email address.

9 If you want to save or print a copy of the email, click the appropriate button.

10 Click **Send Email** to send your email.

Submitting Feature Requests

With Rational Rose RealTime, you can email feature requests to the Rational Software Customer Service department that services your location. When you email a feature request directly from the Rational Rose RealTime application, a wizard guides your through the process, ensuring that you provide the correct information to the Rational Software Customer Service department. This information includes contact and location information, and a detailed description of the feature that your are requesting.

To submit a feature request:

1 From the **Help** menu, click **Email Technical Support**.

A submenu appears, providing you with three options.

2 Click **Feature Request**.

The **General Information** dialog appears.

3 Type your contact and location information in the text areas provided and click **Next**.

The **Feature Request - Additional Information** dialog appears.

4 In the **Request Title** text area, type a detailed name for the Feature that you are requesting.

5 Select the level of urgency for the feature that you are requesting.

6 Describe the feature, using the categories provided in the **Details** area.

7 Click **Next**.

The **Email Summary** dialog appears.

8 Ensure that the information that appears in the **Email Summary** dialog is accurate.

Note: The email information displayed in the **Technical Support Email Address** is chosen based on the location information that you provided in the General Information dialog. It is not recommended that you edit the e-mail address.

9 If you want to save or print a copy of the email, click the appropriate button.

10 Click **Send Email** to send your email.

Submitting Support Requests

With Rational Rose RealTime, you can email Support requests to the Rational Software Customer Service department that services your location. When you email a Support request directly from the Rational Rose RealTime application, a wizard guides you through the process, ensuring that you provide the correct information to Rational Customer Support department. This information includes contact and location information, and a detailed description of the support request that you are submitting.

To submit a support request:

1 From the **Help** menu, click **Email Technical Support**.

A submenu appears, providing you with three options.

2 Click **Support Request**.

The **General Information** dialog appears.

3 Type your contact and location information in the text areas provided and click **Next**.

The Support Request - Additional Information dialog appears.

4 In the **Request Title** text area, type a detailed name for the request that you require.

5 Select the level of urgency for the question with which you need help.

6 Type your question in the **Question** text area.

7 Click **Next**.

The **Email Summary** dialog appears.

8 Ensure that the information that appears in the **Email Summary** dialog is accurate.

Note: The email information displayed in the Technical Support Email Address is chosen based on the location information that you provided in the General Information dialog. It is not recommended that you edit the e-mail address.

9 If you want to save or print a copy of the email, click the appropriate button.

10 Click **Send Email** to send your email.

Contacting Rational Customer Service by Email or Telephone

When contacting Rational Customer Service by email or by telephone, please be prepared to supply the following information:

- Name, telephone number, and company name
- Product name and version number
- Operating system and version number (for example, Windows NT 4.0, Windows 2000, Windows XP, Solaris 2.6/2.7/2.8, or HP-UX 10.20)
- Computer make and model
- Your case id (if you are calling about a previously reported problem)
- A summary description of the problem, related errors, and how it was made to occur

If your organization has a designated, on-site support person, please try to contact that person before contacting Rational Customer Service.

You can obtain technical assistance by sending electronic mail to the appropriate email address. Electronic mail is acknowledged immediately and is usually answered within one working day of its arrival at Rational. When sending an email place “Rational Rose RealTime” in the subject line, and in the body of your message include a description of your problem.

When sending email concerning a previously-reported problem, please include in the subject field: "`[SR# XXXXXXXXX]`", where `XXXXXXXXX` is your Service Request number. For example:

```
[SR# 111222333] Rational Rose RealTime installation issues
```

Sometimes Rational Customer Service engineers will ask you to fax information to help them diagnose problems. You can also report a technical problem by fax if you prefer. Please mark faxes "**Attention: Technical Support**" and add your fax number to the information requested above.

Telephone, fax, and email information for Rational Customer Service are Table 3. If you have problems or questions regarding licensing, please see *License Support Contact Information* on page 71.

Table 3 Telephone and fax and email information

Your Location	Telephone	Fax	E-mail
North America	(800) 433-5444 (toll free) (408) 863-4000 Cupertino, CA	(781) 676-2460 Lexington, MA	support@rational.com
Europe, Middle East, Africa	+31 (0) 20-4546-200 Netherlands	+31 (0) 20-4546-202 Netherlands	support@europe.rational.com
Asia Pacific	+61-2-9419-0111 Australia	+61-2-9419-0123 Australia	support@apac.rational.com

License Support Contact Information

If you have a problem or questions regarding the licensing of your Rational Software products, please contact the Licensing Support office nearest you.

Telephone numbers for license support are listed in the following table. Ask for, or select, **Licensing Support**.

Table 4 License Support Telephone and Fax

Region	Telephone Number	Fax Number
Americas	800-433-5444	781-676-2510
Europe, Israel, and Africa	+31 (0)20 4546 200	+31 (0)20 4546 202
North Asia Pacific (Mainland China, Hong Kong, Taiwan)	+852 2143 6382	+852 2143 6018
Korea	+82 2 556 9420	+82 2 556 9426
South Asia Pacific Australia, New Zealand, Malaysia, Singapore, Indonesia, Thailand, The Philippines, Vietnam, Guam and India	+61-2-9419-0111	+61 2 9419 0123
Japan	+81 3 5423 3611	+81 3 5423 3622

Email addresses for license support are listed in the following table.

Table 5 License Support Email

Region	Email Address
Americas	lic_americas@rational.com
Europe, Israel, and Africa	lic_europe@rational.com
North Asia Pacific Mainland China, Hong Kong, Taiwan, and Korea	lic_apac@rational.com
South Asia Pacific Australia, New Zealand, Malaysia, Singapore, Indonesia, Thailand, The Philippines, Vietnam, Guam and India	lic_apac@rational.com
Japan	lic_japan@rational.com

Index

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X