

Rational Environment Release Information

Release D_12_7_3

Copyright © 1992 by Rational

Part Number: 508-003207-007

November 1992 (Software Release D_12_7_3)

IBM is a registered trademark and RISC System/6000 is a trademark of International Business Machines Corporation.

OSF/Motif is a trademark of Open Software Foundation, Inc.

PostScript is a registered trademark of Adobe Systems Incorporated.

Rational and R1000 are registered trademarks and Rational Environment is a trademark of Rational.

Sun Workstation is a registered trademark and OpenLook, NeWS, and SunOS are trademarks of Sun Microsystems, Inc.

UNIX is a registered trademark of UNIX System Laboratories, Inc.

VMS is a trademark of Digital Equipment Corporation.

X Window System is a trademark of MIT.

Rational
3320 Scott Boulevard
Santa Clara, California 95054-3197

Contents

1. Overview	1
2. Supported Configurations and Upgrades	1
3. Compatibility	2
4. Upgrade Impact	3
4.1. Impact of Specification Changes	4
4.1.1. Upgrading from D_12_6_5 or Later	4
4.1.2. Upgrading from D_12_5_0	4
4.1.3. Upgrading from D_12_2_4	4
4.1.4. Upgrading from D_12_1_1	5
4.2. Impact of Implementation Changes	5
4.2.1. Upgrading from D_12_6_5 or Later	5
4.2.2. Upgrading from D_12_5_0, D_12_2_4, or D_12_1_1	5
5. Known Problems	5
6. New Environment Features	6
6.1. Network Installation	6
6.2. Crash Notification	6
6.3. New Procedures in Package Dfs	6
6.3.1. Reboot_On_Failure	7
6.3.2. Reboot_On_Failure_Interval	7
6.3.3. Quiesce_Reboot_On_Failure	7
6.3.4. Reboot_On_Failure_Settings	7
6.4. New Procedures in System_Maintenance	7
6.4.1. Show_Elaborated_Configuration	7
6.4.2. Show_Session_Of_Job	8
6.5. Rational X Interface Enhancements	8
6.6. New World Network_Public_Archive_Server_Sessions	8
7. Changes from D_12_6_5	8
7.1. Changes to Package Common	9
7.1.1. Selection in Environment Menus	9
7.1.2. Locks and Program_Errors	9
7.2. Changes to Package What	9
7.3. Changes to Archive	10
7.3.1. Archiving into Subsystems	10
7.3.2. Archiving Loaded Main Programs	10
7.3.3. Archiving Searchlists	11
7.3.4. Archiving Ventures	11
7.3.5. Archiving to Tape Across a Network	11
7.3.6. Archive Server and Session Tokens	11
7.4. Compilation Changes	13
7.4.1. Installing Units	13
7.4.2. Coding Units	15
7.4.3. Loading and Executing Units	15
7.4.4. Incremental Compilation	15
7.5. Miscellaneous Library Management Changes	16
7.6. Changes to Crash Analysis	16
7.6.1. Failure Reboot	16
7.6.2. Automatic Notification	17
7.7. Miscellaneous Changes to System Management	18
7.8. Changes to CMVC	18
7.8.1. Make_Controlled and the State Directory	18

7.8.2. Destroy_View and Frozen Objects	18
7.8.3. Naming Conflicts when Copying or Building Views	19
7.8.4. Default Parameter Values for Make_Spec_View	19
7.8.5. Consecutive Calls to Release	19
7.8.6. Cmvvc_Access_Control	19
7.8.7. Log Messages	20
7.8.8. Work Orders	20
7.9. Changes to Activity_Implementation	20
7.10. Networking Changes	20
7.11. LRM Interface Changes	21
7.11.1. Discriminated Type Function Changes	21
7.11.2. New Aggregate Range Function	22
7.11.3. Declarations.Kind Change	22
7.11.4. New Associations Package	22
7.11.5. Lexical Typing of Iterators	23
7.11.6. Compilation Unit Pragmas	24
7.11.7. Operations on Generic Instantiations	24
7.11.8. Functions and Enumeration Literal Renames	25
7.11.9. Constants versus Variables	26
7.11.10. Task Entries versus Subprograms	26
7.11.11. Comments in Declaration Names	26
7.11.12. New Labels Function	26
7.11.13. Limited Private Generic Formal Parameters	26
7.11.14. Change to Renaming Declarations Function	27
7.11.15. Renamed Declarations	27
7.11.16. Subunits Without Bodies	27
7.11.17. Subunits Without Specs	27
7.11.18. Declarations.Is_Initialized Change	27
7.11.19. New Function for Incomplete Types	28
7.11.20. Implementation-Dependent Attributes and Pragmas	28
7.11.21. Return Type of Function Instantiations	30
7.11.22. Block and Loop Names	30
7.11.23. Enumeration Literal Queries	30
7.11.24. Record Discriminant and Component Declarations	30
8. Documentation	31
8.1. Printed Documentation	31
8.1.1. Correction to <i>Quick Reference for Parameter-Value Conventions</i>	31
8.2. Online Documentation	31
9. Training	32
A Problem Reports Closed in D_12_7_3	33

1. Overview

The D_12_7_3 release of the Rational Environment™ is primarily a maintenance release that:

- Supports installation of Environment releases over a network (section 6.1).
- Simplifies the process of crash analysis and recovery (sections 6.2, 6.3, 7.6).
- Introduces a new release of the Rational X Interface (RXI) (section 6.5).
- Introduces Rational Access, an OSF/Motif™-based graphical user interface for the Environment. Access runs as an X Window System™ application on a UNIX® workstation. It features:
 - The OSF/Motif mouse, menu, and dialog box paradigm.
 - Many new “full service” commands, which combine several lower-functionality Environment operations.
 - A comprehensive, menu-based online help facility.
 - Platform-independent key and mouse bindings.
 - Full support for existing Environment features such as “item-operation” combinations and command windows.

For complete information, see the *Rational Access User's Guide* and *Rational Access Release Information* delivered with this release.

- Fixes a large number of problems from previous releases, enabling existing commands to work correctly. Some of these fixes introduce new features (see section 6); other fixes involve changed features (see section 7).
- Introduces a new, up-to-date *System Manager's Guide*, containing combined information for Series 200, 300, and 400 system (see section 8.1).
- Introduces new online help for each of the packages described in the Debugging book (Volume 3) and the Project Management book (Volume 11) of the *Rational Environment Reference Manual* (see section 8.2).

Note: This release information can be found online in the !Machine.Release.Release_Notes world in line-printer (Environment_Release12_7_3_Lpt) and PostScript® (Environment_Release12_7_3_Ps) formats.

2. Supported Configurations and Upgrades

D_12_7_3 supports the following configurations of the R1000®:

- Series 200 (Models 10, 20, and 40)
- Series 300 System (300S)
- Series 300 Coprocessor (300C) for the IBM® RISC System/6000™ and Sun Workstation® servers
- Series 400 System (400S)
- Series 400 Coprocessor (400C) for the IBM RISC System/6000 and Sun Workstation servers

D_12_7_3 supports two kinds of tape drive:

- The 9-track tape drive, which is standard on the Series 100 and 200, and an optional upgrade to the Series 300S
- The 8-millimeter tape drive, which is standard on the Series 300S, 300C, 400S, and 400C, and an optional upgrade to the Series 200

D_12_7_3 also supports the optional expansion of memory from 32 megabytes to 64 megabytes to improve system performance. This upgrade applies to all series except the Series 100. The combinations of configurations and upgrades supported by D_12_7_3 are shown in Table 1.

Table 1 Configurations and Upgrades Supported by D_12_7_3

Configuration	8-mm Tape Drive	9-Track Tape Drive	32-Mb Memory	64-Mb Memory
Series 200	Upgrade	Standard	Standard	Upgrade
Series 300S	Standard	Upgrade	Standard	Upgrade
Series 300C	Standard	N/A	Standard	Upgrade
Series 400S	Standard	N/A	Standard	Upgrade
Series 400C	Standard	N/A	Standard	Upgrade

3. Compatibility

D_12_7_3 is fully compatible with all production versions of Rational layered-software products (Table 2).

Table 2 Compatibility with D_12_7_3

Rational Product	Compatible Release
CDF: Mc68020_Bare	5_1_2 or later
CDF: Mc68020_Os2000	6_1_3 or later
CDF: Mc68020_Hp_Unix	6_2_4 or later
Design Facility: 2167	6_0_7 or later*
Design Facility: 2167A	6_2_5 or later*
Design Tools	10_2_9 or later
Documentation Tools	10_2_9 or later
Insight	1_3_0
Mail	11_4_5 or later
Rational Access	1_0_1
Rational Publishing Interface	1_0_2 or later

Table 2 Compatibility with D_12_7_3 (continued)

Rational Product	Compatible Release
Rational Teamwork Interface	2_1_2 or later
Rational X Interface	10_5_2 or later
Rational Windows Interface	10_1_1 or later
Remote Compilation Facility	1_1_1 or later
RCF: Ibm_Rs6000_Aix	1_1_0 or later
RPC	1_0_2 or later
Software Analysis Workstation	6_0_0 or later
Target Build Utility	10_0_3 or later
TestMate	2_0

* As delivered, D_12_7_3 is compatible only with Design Facility:2167A release 7_2_1. However, other releases and customizations can be made compatible by recompiling them with the new LRM Interface (see section 7.11).

4. Upgrade Impact

The Environment can be upgraded from to D_12_7_3 without forcing you to Archive.Save and Archive.Restore your applications. You will not have to modify or recompile any of your own tools, with the possible exception of:

- Tools, Rational Design Facility releases, or Rational Design Facility customizations written against the LRM Interface. In particular, changes made to the Declaration_Kinds, Attribute_Designator_Kinds, or Pragma_Kinds enumerations may result in compilation or runtime errors. See section 7.11 for details.

The procedure Find_Lrm_Change_Candidates, located in the Pdl_Commands subsystem, can be used to assist in identifying potential compatibility problems that are not caught by the compiler.

- Tools written against the unit specifications listed in "Impact of Specification Changes," below.
- Customizations of unit bodies listed in "Impact of Implementation Changes," below.

The new declarations listed in section 6 are all installed upward-compatibly and therefore have no impact on user-written tools.

Rational does not provide tools for reverting to previous releases of the Environment. You will be able to revert to the previous release, however, if you make a complete backup of the Environment, including the Diagnostic File System (DFS), before you upgrade.

4.1. Impact of Specification Changes

The installation process for D_12_7_3 changes several Environment specifications. The installation process attempts to make all changes incrementally; however, if incremental changes fail, the installation process overwrites the necessary specifications. Overwriting these specifications will cause the demotion of any customer-created tools written against them. The installation process tries to recompile such tools automatically; however, depending on the nature of the tools, some may require modification before they can be recompiled. Units that cannot be recompiled during installation are listed in the installation log.

The unit specifications that are overwritten depend on the release from which you are upgrading.

4.1.1. Upgrading from D_12_6_5 or Later

The following unit specifications are changed when you upgrade from D_12_6_5, D_12_6_6, or D_12_6_7 to D_12_7_3:

- !Implementation.Activity_Implementation'Spec
- !Commands.Cmvc'Spec

4.1.2. Upgrading from D_12_5_0

In addition to the unit specifications overwritten when upgrading from D_12_6_5 (or later), upgrading from D_12_5_0 overwrites the following unit specifications:

- !Commands.Library'Spec
- !Commands.System_Backup'Spec
- !Tools.Access_List_Tools'Spec
- !Tools.Code_Segment_Object_Editor'Spec
- !Tools.Library_Object_Editor'Spec

If you are upgrading from D_12_5_0, you should also read the release information for D_12_6_5; release information is located online in !Machine.Release.Release_Notes.

4.1.3. Upgrading from D_12_2_4

In addition to the unit specifications overwritten when upgrading from D_12_6_5 (or later) and D_12_5_0, upgrading from D_12_2_4 overwrites the following unit specifications:

- !Commands.Abbreviations.Print'Spec
- !Commands.Access_List'Spec
- !Commands.Archive'Spec
- !Commands.Queue'Spec
- !Commands.Remote_Passwords'Spec
- !Tools.Networking.Tcp_Ip_Boot'Spec

- !Tools.Networking.Transport'Spec
- !Tools.Networking.Transport_Defs'Spec
- !Tools.Networking.Transport_Name'Spec

Furthermore, the units in !Machine.Initialize@ are either demoted or moved to other locations to accommodate the D_12_5_0 mechanisms for initializing an R1000; for complete details, see Appendix E, "Machine Initialization," in the *System Manager's Guide* delivered with this release or the online information in !Machine.Initialization.Guide_To_Machine_Initialization.

If you are upgrading from D_12_2_4, you should also read the release information for D_12_5_0 and D_12_6_5; release information is located online in !Machine.Release.Release_Notes.

4.1.4. Upgrading from D_12_1_1

In addition to the unit specifications overwritten when upgrading from D_12_6_5 (or later), D_12_5_0, and D_12_2_4, upgrading from D_12_1_1 overwrites the following unit specification:

- !Implementation.Work_Order_Implementation'Spec

If you are upgrading from D_12_1_1, you should also read the release information for D_12_2_4, D_12_5_0, and D_12_6_5; release information is located online in !Machine.Release.Release_Notes.

4.2. Impact of Implementation Changes

Because unit bodies may contain user customizations, an overwritten unit body's contents are saved in a text file in the same library as the overwritten body. The name of the file is of the form *Unit_Name_Vnn*, where *nn* is the unit's default version number. These customizations then can be transferred to the new implementation.

The unit bodies that are overwritten by the D_12_7_3 installation depends on the release from which you are upgrading.

4.2.1. Upgrading from D_12_6_5 or Later

No unit bodies are overwritten when you upgrade from D_12_6_5, D_12_6_6, or D_12_6_7 to D_12_7_3.

4.2.2. Upgrading from D_12_5_0, D_12_2_4, or D_12_1_1

Upgrading from D_12_5_0, D_12_2_4, or D_12_1_1 overwrites the following unit body:

- !Commands.Abbreviations.Print'Body

5. Known Problems

There are no known problems introduced by this D_12_7_3 release.

6. New Environment Features

D_12_7_3 provides new interfaces and features, including:

- New ability to install Environment releases over the network (see section 6.1).
- New ability to specify who, if anyone, the system is to notify after a crash (section 6.2).
- New procedures in package Dfs for setting Failure Reboot options (section 6.3).
- New procedures in the System_Maintenance subsystem for displaying configuration information and displaying the session associated with a job (section 6.4).
- A new release of the Rational X Interface (RXI) (section 6.5).
- New world containing additional Network_Public sessions to be used by the Archive Server to alleviate token problems (section 6.6).

6.1. Network Installation

The D_12_7_3 release is distributed with a set of network installation tools. This feature is only of interest to those sites that have multiple machines connected via a network, and only at the time the D_12_7_3 release is installed.

In previous releases, the installation tapes for a new Environment release had to be separately loaded on each machine that was to be upgraded. With the D_12_7_3 release, the installation tapes only have to be loaded on one machine on the network. All of the other machines can be upgraded over the network.

Refer to the D_12_7_3 *Installation Procedure* for details.

6.2. Crash Notification

Since the introduction of tombstone files in the D_12_5_0 release, the system has produced an analysis of the previous outage; this analysis is written to the system error log. In D_12_7_3, system managers can specify additional distribution of the analysis output in a new control file. The pathname of this file is:

```
!Machine.Initialiation.Local.Previous_Outage_Configuration
```

This file contains information about who to notify by mail after the system reboots and where to put or send information about the failure. For more information about this file and other changes made to the process of crash analysis and recovery, see section 7.6 or Appendix B, "Diagnostic Crash Procedures for Release D_12_7_3" in the *System Manager's Guide* delivered with this release.

6.3. New Procedures in Package Dfs

In D_12_7_3, the system can be configured to reboot following a failure as long as no hard failure has been detected. This feature is known as *Failure Reboot*.

New procedures in package !Machine.Dfs'Spec_View.Units.Dfs control the Failure Reboot feature. They are described in the following subsections.

For more information about the new Failure Reboot feature, see section 7.6 or Appendix B, "Diagnostic Crash Procedures for Release D_12_7_3" in the *System Manager's Guide* distributed with this release.

6.3.1. Reboot_On_Failure

procedure Reboot_On_Failure (Enabled : Boolean := True)

Sets whether the R1000 will automatically reboot after a failure. If True, the system attempts to automatically reboot after a system failure.

6.3.2. Reboot_On_Failure_Interval

procedure Reboot_On_Failure_Interval (Number_Of_Days : Integer := 7)

Sets the number of days that must pass between two crashes for the system to allow automatic reboot.

6.3.3. Quiesce_Reboot_On_Failure

procedure Quiesce_Reboot_On_Failure

Temporarily disables the Reboot_On_Failure feature for the next crash. The option is reset to enabled after the next crash or if Reboot_On_Failure is used to reenable the option.

It is possible to quiesce only if Failure Reboot is currently enabled. If Failure Reboot is disabled, quiescing has no effect.

6.3.4. Reboot_On_Failure_Settings

procedure Reboot_On_Failure_Settings

Displays the current settings for Failure Reboot options. The value of Failure Reboot is shown as **ENABLED**, **DISABLED**, or **QUIESCED**. For example:

```
The current value for the reboot interval is 1
The reboot feature is currently: ENABLED
```

6.4. New Procedures in System_Maintenance

D_12_7_3 includes two new procedures in the !Commands.System_Maintenance subsystem.

6.4.1. Show_Elaborated_Configuration

procedure Show_Elaborated_Configuration;

Displays the configuration currently elaborated on this R1000. The configuration is the versions on the operating system subsystems that make up an Environment release. This procedure

displays the name of the current configuration, the subsystems and their versions that the configuration contains, and the microcode version that is running.

This information may be helpful to Rational personnel when they are investigating a problem on the machine. Running this command will not cause any harm to the R1000, but the output will have very little meaning to the end users.

6.4.2. Show_Session_Of_Job

```
procedure Show_Session_Of_Job (Job : Machine.Job_Id := Default.Process);
```

Displays the Session Id of the specified job. If no value is specified for the Job parameter, then the job that is calling this procedure is used. This procedure displays the job number and name, the user identity that the job is running under, the base session name that the job is running under, and the full session name that the job is running under. This procedure is useful in determining who started a given job, or under what identity a given job is executing.

6.5. Rational X Interface Enhancements

D_12_7_3 includes a new release (10_10_1) of the Rational X Interface (RXI). This release includes the following enhancements:

- A terminal type and key binding procedures have been added for XSun101 and XNews101 to support the Sun 101 keyboard. If you use these terminal types, you should use the IBM RS/6000 RXI supplement and keyboard overlays.
- An upgrade to SunOS™ 4.1.2, VMS™ 5.4, Sun NeWS™/OpenLook™ 3.0, and MIT X Window System X11R5.
- A new directory containing PostScript copies of all RXI keyboard overlays. This directory is !Machine.Editor_Data.Keyboard_Overlays. To print a keyboard overlay from the Environment, specify the Original_Raw option to the Print command.

The basic behavior and features of RXI have not been changed.

6.6. New World Network_Public_Archive_Server_Sessions

The new world !Machine.Network_Public_Archive_Server_Sessions contains objects for seven new sessions to be used by the Archive Server. These sessions are called Network_Public_Session_1 through Network_Public_Session_7. For more information about these sessions and changes to the Archive Server, see section 7.3.6.

7. Changes from D_12_6_5

This section describes the changes, enhancements, and user-visible problem fixes that D_12_7_3 makes to existing features of the Environment. The information in this section is presented in roughly the same order in which it would be found in the *Rational Environment Reference Manual* (ERM).

- Basic editing operations such as selection and definition (EI and EST) (section 7.1)

- Session and job management (SJM)—specifically, package What (section 7.2)
- Library-management (LM), including changes to archive (section 7.3), compilation (7.4), and other library-management facilities such as access control (7.5)
- System-management (SMU), including changes to crash analysis (section 7.6, printing, and messages (7.7)
- Subsystems, CMVC, and activities (PM) (sections 7.8 and 7.9)
- Networking facilities such as Telnet and FTP (section 7.10)
- LRM Interface (section 7.11)

7.1. Changes to Package Common

7.1.1. Selection in Environment Menus

Selection of entire Environment menus has been fixed. In previous releases, selecting something in a menu would fail if:

- You had two menus displayed on your screen, and
- One menu was selected completely, and
- You attempted to select anything in the other menu.

For example, if an obsolescence menu and an xref menu were displayed and all of the entries in the obsolescence menu were selected, you could not select anything in the xref menu. In D_12_7_3, the new selection occurs and the other menu is deselected. PRS numbers 2115837-Etoi-Phl and 8995225-Shei-Jst.

7.1.2. Locks and Program_Errors

The Environment no longer generates a Program_Error message in the following two cases:

- If an object is locked and being written to by another job or user and you attempt to execute a search command in the image of that object. PRS number 5098655-Mago-Sdj.
- If you execute Common.Definition on a large text file and attempt to perform an operation in that image before it is completely displayed. PRS number 8957642-Cook-Swb.

In both of the above cases the Environment now generates an appropriate lock message.

7.2. Changes to Package What

When What.Locks is applied to a deleted unit, it no longer fails with an undecipherable message. The new error message correctly identifies the problem. PRS numbers 10464-Star and 5476876-Voya-Phil.

When What.Object is executed on a window that does not correspond to a library object (such as an I/O window or a mail message), it now fails with an appropriate message: `What.Object failed - <IMAGE> does not refer to a directory object`. PRS number 12728-Star.

What.Object no longer fails when:

- It is executed on a window that contains the image of an associated file (indicated in the library display by pointy brackets). PRS number 167263-Clem-Marl.
- It is executed in a text file and the cursor is located inside a selection within that file. PRS number 4317752-Clem-Marl.

The output from the What.Users command now contains two or more entries for the identity Network_Public. One entry contains all of the jobs that were traditionally run under the session !Machine.Network_Public_Session, such as the Print Queue Server and the Mail Distribute Server. The additional entries each contain only one job, an Archive Sever job. These additional Network_Public entries are the result of changes to the Archive Server (see section 7.3.6). In order to determine the session under which a particular job (such an Archive Server job) is running, see the new Show_Session_Of_Job command (section 6.4).

7.3. Changes to Archive

7.3.1. Archiving into Subsystems

The Archive.Copy and Archive.Restore commands no longer allow you to create:

- A subsystem within an existing subsystem
- A world within a subsystem view

If an attempt is made to Copy or Restore a subsystem within a subsystem, the command is terminated with the error **Bad Subsystem Structure**. Note that when copying a subsystem from one R1000 to another across the network, if the restore part on the target fails with this bad subsystem structure error, then the restore part of the archive terminates — however, the sending part of the Archive operation continues sending units until it realizes that the receiver has terminated. At that point, the sender terminates with a **connection broken** error. You must look earlier in the log to see the “real” error message indicating that you are not allowed to create subsystems within subsystems.

If an attempt is made to Copy or Restore a world into a subsystem view, the command now converts the world into a directory before copying or restoring it. PRS number 0-0298-0.

7.3.2. Archiving Loaded Main Programs

In previous releases, the Archive.Copy command failed if you attempted to copy a loaded main program (Load_Proc or Load_Func) to another location or another machine where:

- A coded older version of that loaded main program already existed, and
- The archive job did not have full access to replace the existing loaded main program. but had enough access to demote the existing loaded main program

In such a case, the copy would fail but the existing version of the loaded main program would be left in archived state and in some cases could not be promoted back to coded state.

In D_12_7_3, the Archive job checks the target where a loaded main program is to be restored and makes sure that the identity of the archive job has enough access to the enclosing world and the existing object to create a new version of the object, *before* it demotes the existing object to archived state. If the job does not have appropriate access then the Copy command fails with a detailed error message explaining the problem. PRS number 9123690-0158-2 and CSR number 5079.

7.3.3. Archiving Searchlists

In previous releases, the Archive.Copy command failed if you attempted to copy a searchlist to another machine on the network and the entries in the searchlist could not be resolved on the target machine. The Copy command failed with a misleading error message saying that the subclass of the restored object could not be set to searchlist.

In D_12_7_3, the Archive.Copy command writes a message saying that the searchlist could not be properly restored on the target machine because it contains entries that are not resolvable on the target machine.

The searchlist is copied as a text file onto the target machine. You can then log into the target machine and edit the text file so that all the entries are resolvable and then execute the command Search_List.Revert to make the text file into a valid search list. PRS number 9123690-0183-5 and CSR number 6426.

7.3.4. Archiving Ventures

When used on ventures that contain string fields, the Archive.Copy command no longer inserts extra quotation marks into those strings. PRS number 1005010-Etoi-Ksch.

7.3.5. Archiving to Tape Across a Network

In a recent release of the Environment the capability to do an Archive.Save to a remote machine's tape drive across the network was added. However, there was a bug in this capability such that the tape would not be unloaded from the remote tape drive when the Archive.Save was finished. This has been fixed and the tape is unloaded unless otherwise specified. PRS number 747500-Gato-Rjg.

7.3.6. Archive Server and Session Tokens

The Archive Server is a job that is started each time the system boots. This job exists to process requests from other R1000's that are attempting to Archive.Copy or Archive.Restore objects onto the current R1000 across the network. The Archive Server also processes Remote.Run jobs that are sent from other R1000's on the network. The Archive Server works in the following way:

1. It receives an incoming request to process.
2. It spawns another Archive.Server job, which waits for another request from a remote R1000.
3. It (the original Archive Server job) takes the initial remote request and processes it.

In previous releases, the Archive Server jobs all ran under the same session (!Machine.Network_Public_Session). This caused a problem because some remote requests require that the Archive Server consume a layered-product token to accomplish the request. Tokens are session-based and are not released until the session is logged out or terminated. Unfortunately, since there is always a Archive Server job running, the session never terminates, so the token was never released.

In this D_12_7_3 release, the Archive Server jobs cycle through seven new sessions. When an Archive Server job finishes, the session terminates and the token is released to be scavenged when it is needed.

The problem described above was only a problem on token-based R1000's but the changes will appear on all R1000s so you should not be surprised when you see this different behavior. The new behavior you will notice is as follows:

- The output from the What.Users command contains two or more entries for the identity Network_Public:
 - One entry contains all of the jobs that were traditionally run under the session !Machine.Network_Public_Session, such as the Print Queue Server and the Mail Distribute Server.
 - The additional entries for the identity Network_Public each contain only one job, an Archive Sever job.

To determine the session under which a particular job (such as an Archive Server job) is running, see the new Show_Session_Of_Job command (section 6.4).

- The new world !Machine.Network_Public_Archive_Server_Sessions contains objects for seven new sessions to be used by the Archive Server. These sessions are called Network_Public_Session_1 through Network_Public_Session_7.
- The first time the system is booted after D_12_7_3 has been installed, you will see messages that indicate that the new sessions have been created. It will look something like the following:

```

19:01:29 !!! Predefined_Users User_create_failed : Public: User
              already exists
19:01:30 !!! Predefined_Users Prohibit_login_failed Delete status
              for Public S_1 : NAME_ERROR
19:01:30 !!! Predefined_Users User_create_failed : Network_Public:
              User already exists
19:01:30 !!! Predefined_Users Prohibit_login_failed Delete status
              for Network_Public.S_1 : NAME_ERROR
19:01:30 !!! Predefined_Users User_create_failed : Operator: User
              already exists
19:01:30 --- Predefined_Users Created_group : Privileged
19:01:30 --- Predefined_Users Created_group : Mailer
19:01:31 --- Predefined_Users Created_group : Spooler
19:01:31 !!! Predefined_Users User_create_failed : Rational: User
              already exists
19:01:31 --- Boot Running "!Machine.Initialization".Start
19:01:31 --- Predefined_Users Created_group : System
19:01:31 --- Boot Running Destroy deleted Ada/Link objects
19:01:33 +++ Predefined_Users Created_Session
              !Machine.Network_Public_Session
    
```

```

19:01:33 +++ Predefined_Users Created_Session
!Machine.Public_Session
19:02:55 +++ Predefined_Users Created_World !Machine.
Network_Public_Archive_Server_Sessions
19:02:55 +++ Predefined_Users Attempting_to_create_session !Machine.
Network_Public_Archive_Server_Sessions.Network_Public_Session_1
19:02:56 +++ Predefined_Users Created_Session !Machine.
Network_Public_Archive_Server_Sessions.Network_Public_Session_1
19:02:56 +++ Predefined_Users Attempting_to_create_session !Machine.
Network_Public_Archive_Server_Sessions.Network_Public_Session_2
19:02:58 +++ Predefined_Users Created_Session !Machine.
Network_Public_Archive_Server_Sessions.Network_Public_Session_2
19:02:58 +++ Predefined_Users Attempting_to_create_session !Machine.
Network_Public_Archive_Server_Sessions.Network_Public_Session_3
19:02:59 +++ Predefined_Users Created_Session !Machine.
Network_Public_Archive_Server_Sessions.Network_Public_Session_3
19:02:59 +++ Predefined_Users Attempting_to_create_session !Machine.
Network_Public_Archive_Server_Sessions.Network_Public_Session_4
19:03:00 +++ Predefined_Users Created_Session !Machine.
Network_Public_Archive_Server_Sessions.Network_Public_Session_4
19:03:00 +++ Predefined_Users Attempting_to_create_session !Machine.
Network_Public_Archive_Server_Sessions.Network_Public_Session_5
19:03:01 +++ Predefined_Users Created_Session !Machine.
Network_Public_Archive_Server_Sessions.Network_Public_Session_5
19:03:01 +++ Predefined_Users Attempting_to_create_session !Machine.
Network_Public_Archive_Server_Sessions.Network_Public_Session_6
19:03:02 +++ Predefined_Users Created_Session !Machine.
Network_Public_Archive_Server_Sessions.Network_Public_Session_6
19:03:03 +++ Predefined_Users Attempting_to_create_session !Machine.
Network_Public_Archive_Server_Sessions.Network_Public_Session_7
19:03:03 +++ Predefined_Users Created_Session !Machine.
Network_Public_Archive_Server_Sessions.Network_Public_Session_7

```

7.4. Compilation Changes

In previous releases, status messages produced during the compilation of objects contained the version of the object being compiled (e.g., !Users.Geb.Bug_Regressions.Pass-Deferred_Private'Body'V(1).). The 'V(n)' part was determined to be extraneous and sometimes confusing. In D_12_7_3, compilation messages contain only the full path name of the object (e.g., !Users.Geb.Bub_Regressions.Pass_Deferred_Private'Body.). PRS number 2911238-Jude-Geb.

7.4.1. Installing Units

In previous releases, if a package contained both a *with* clause referencing package System and a variable named System, then the variable declaration would hide the *with* to package System and constructs such as representation specifications would not code, saying that there was no visibility to package System. In D_12_7_3, the R1000 compilation system locates the "with System" clause directly, without relying on visibility. PRS number 3078226-Gato-Mboy.

Renaming a field in a constant no longer fails with an error indicating that something was not fully constrained. In D_12_7_3, constants are fully constrained and fully renamable. PRS

number 3108823-Sier-Geb.

The semantic checker now flags as ambiguous a series of derived types and procedures that use those types as parameters. PRS number 4711901-Rati-Pbk.

In previous releases, if the completion type of a forward declaration was in a package body and that type was a discriminant record type, the semantic checker did not always find semantic errors in the completion record type. In D_12_7_3, two-part records are semantically checked in the same manner regardless of whether they exist in separate declaration contexts or the same declaration context. PRS number 5844095-Aria-Sbur.

In accordance with LRM 7.4(4), the semantic checker no longer allows the declaration of a deferred constant for a private type that has been declared in another unit. (LRM 7.4(4) states that "a deferred constant declaration and the declaration of the corresponding private type must both be declarative items of the visible part of the same package.") PRS number 6370093-Etoi-Phl.

In D_12_7_3, the semantic checker now flags as ambiguous a function call that returns an array type when:

- The function has a single integer parameter, and
- The declaration of the integer parameter includes a default value (such as `function x (Int : Integer => 6) return array`), and
- That integer parameter is specified without an explicit assignment. For example, `x(2)` is ambiguous, but `x(Int => 2)` is not.

`x(2)` should be flagged as ambiguous because it does not specify whether the argument is to be the parameter passed into the function or an index into the returned array type. In previous releases, the semantic checker assumed that it was an index into the return array and the value for the parameter was given its default value. PRS number 7762494-Rati-Pbk.

The semantic checker now handles representation specifications for Long_Integer types. In previous releases, the semantic checker would sometimes fail with a Numeric_Error (Overflow). PRS number 8391944-Gato-Gbd.

In accordance with Ada Issue AI-00438/09, the semantic checker now allows functions that rename enumeration literals, rather than flagging them as not static. (Ada Issue AI-00438/09 states: "A function call whose function name denotes an enumeration literal, a predefined operator, or a function that is a language-defined attribute of a static subtype is allowed as a primary in a static expression if the value of each parameter is given by a static expression.") PRS number 8480408-Gato-Gbd.

The semantic checker no longer flags as erroneous the following situation:

- A generic package contains a type declaration and a subtype declaration of that type, and
- There is an instantiation of the generic, and
- There is a procedure that *withs* that instantiation and contains declared variables of both the type and the subtype, and
- There is an attempt to assign those variables to one another.

PRS number 9123690-0184-6 and CSR number 6625.

Improved Messages:

In D_12_7_3, the semanticist no longer generates a garbled message with the real error explanation when a semantic error occurs due to an invalid representation specification. Only the real error message is shown. PRS number 12862-Star.

In previous releases, if a record type contained a field that was not semantically correct or not defined and that type had a representation specification, then the rep spec would be flagged with a non-meaningful error message. In version, this has been fixed and the actual problem is flagged and described correctly. PRS number 3496838-Gato-Rjg.

In D_12_5_0 and later releases, the error message was formatted badly when a semantic error was flagged indicating that a call was made to a nonexistent procedure or function. The format has now been fixed. PRS number 9072739-Shei-Swb.

7.4.2. Coding Units

The R1000 code generation now generates correct code when a type completion of a forward declaration is a derived type. PRS number 3032909-Cook-Swb.

In previous releases, the R1000 code generator would fail to code a subprogram containing an enumerated type of characters that had a representation specification when a string literal of a string type was declared with this enumeration type of character. Instead, the code generator raised the internal error **Unimplemented: Element type of string literal has non-static size**. This problem has been fixed and the correct code is generated. PRS number 6969167-Gato-Gbd.

7.4.3. Loading and Executing Units

In previous releases, a program would raise a **Type_Error** when executing a **Return** statement if the function executing the **Return** contained a variable that was code-optimized incorrectly. The locally declared variable that was being returned would get popped from the control stack when the **Return** was executed, and its type was lost. This has been fixed and the type of the variable being returned is preserved until the return is complete. PRS number 4290910-Sier-Geb.

Unhandled exceptions from the Environment that result from errors in user programs now contain the string name of the exception. In the D_12_5_0 release, there were some cases in which the Environment displayed the internal numeric representation for the exception rather than the string name. PRS number 9031863-Gato-Gbd.

7.4.4. Incremental Compilation

Incrementally adding a type completion now produces the same DIANA as recompiling the object in batch mode. In previous releases, if a type completion of a forward declaration was incrementally inserted into a package, the DIANA was left in an inconsistent state and did not indicate that the forward declaration had been completed. PRS number 426364-Shei-Jst.

7.5. Miscellaneous Library Management Changes

The Library.Space command no longer fails with a `Numeric_Error (Overflow)` message. PRS number 12612-Star.

The logs generated by the `Access_List.Set` and `Access_List.Set_Default` commands now echo the value of the Response parameter. PRS number 523289-Sier-Geb.

Delete access is no longer required to the destination world to copy or move an Ada package specification and body in a single operation. This change fixes a problem introduced in D_12_5_0 in which attempts to copy or move Ada unit bodies and specifications to a world for which the user did not have Delete access copied (or moved) the body but failed to copy (or move) the specification, saying that Write access was required. PRS number 9123690-0194-1.

7.6. Changes to Crash Analysis

The D_12_7_3 release of the Environment includes several major changes to crash analysis. In particular:

- Much of the process has been automated so that the system captures as much information as possible and allows system managers to set the system to reboot automatically after a failure.
- The system now automatically performs the basic crash analysis and tests that previously required user intervention. The system writes the results of these tests to files that can be sent or faxed to Rational after the system reboots. When necessary, the system stops at a menu and offers a recommended action as the default.
- The system now automatically notifies users after a system failure and reboot, using electronic mail and a list of users in a new reboot configuration file.

For information about how to respond to the crash of a D_12_7_3 system, see Appendix B, "Diagnostic Crash Procedures for Release D_12_7_3" in the *System Manager's Guide* delivered with this release. The following sections describe the Failure Reboot and Automatic Notification features. Descriptions of these features are also found in the new *System Manager's Guide*.

7.6.1. Failure Reboot

In previous releases, the machine, when properly configured, would automatically reboot following normal shutdowns. If the machine crashed due to a hardware or software error, it would require the operator to respond to prompts for the machine to boot.

In D_12_7_3, the system can be configured to reboot following a failure as long as no hard failure has been detected. This feature is known as *Failure Reboot*.

As a safeguard, the Failure Reboot feature allows the specification of an interval; if two crashes occur during the specified interval, the system does not automatically reboot, but rather awaits a response from the operator. This allows single, isolated crashes to be dealt with in a timely manner, but ensures that, if more than one crash occurs during a given interval, a closer investigation can take place.

Four new procedures in package `!Machine.Dfs'Spec_View.Units.Dfs` control the Failure Reboot feature (see section 6.3 for descriptions):

- `Reboot_On_Failure`
- `Reboot_On_Failure_Interval`
- `Quiesce_Reboot_On_Failure`
- `Reboot_On_Failure_Settings`

As part of the Failure Reboot, the system, when applicable, automatically runs the confidence tests. If these tests detect a hard failure, the system does not reboot and prompts the operator to notify Rational of the failure.

In the cases where no hard failure can be detected, and `Reboot_On_Failure` is enabled, the system reboots. At the completion of the boot, the initialization procedure `Log_Previous_Outage_Start` analyzes the tombstone file produced during the most recent failure.

The system writes the output of this analysis to the system error log, as well as to the file specified by the `Analysis` option in the `Previous_Outage_Configuration` file (see the following subsection). The output of this analysis can then be reviewed and sent to Rational to determine the action, if any, to be taken. Possible actions could be to change the values of the Failure Reboot interval, disable Failure Reboot, use the `Quiesce_Reboot_On_Failure` procedure to allow the explicit creation of a crash-dump tape after the next crash, or other operations recommended by Rational.

In the D_12_7_3 release, the Failure Reboot feature is disabled by default. The system manager or operator should enable the feature with a recommended interval of 14 days. With these settings, the system automatically reboots on the first crash, producing an analysis of the crash after the machine boots. If any subsequent crash occurs within 14 days, the system suspends rebooting and, where applicable, requests the creation of a crash-dump tape for a more complete analysis.

7.6.2. Automatic Notification

After a system failure occurs, the system produces an analysis of the failure and writes it to the system error log. In addition, system managers can specify additional distribution of the analysis output in a new control file. The pathname of this file is:

`!Machine.Initialization.Local.Previous_Outage_Configuration`

The file contains five fields:

- **`Message => <<Pathname for message file>>`**
Specifies a pathname to a file that will have a brief description of the previous outage written to it. The default for this field is `!Machine.Error_Logs.Crash-_Message`. Only one pathname can be specified, or the field can be left blank.
- **`Analysis => <<Pathname for analysis file>>`**
Specifies a pathname to a file that will have a copy of the full analysis of the previous outage written to it. The default for this field is `!Machine.Error_Logs.Crash-_Analysis`. Only one pathname can be specified, or the field can be left blank.

- **Boots => <<list of mail users>>**

Specifies a list of users, separated by commas and optionally on multiple lines, who will be sent a copy of a brief description of the previous outage via R1000 mail. Mail is sent for all outages, including normal shutdowns. By default, this field is blank.

- **Crashes => <<list of mail users>>**

Specifies a list of users, separated by commas and optionally on multiple lines, who will be sent a copy of a brief description of the previous outage via R1000 mail. Mail is *not* sent for normal shutdowns. By default, this field is blank.

- **Full => <<list of mail users>>**

Specifies a list of users, separated by commas and optionally on multiple lines, who will be sent a copy of the full analysis of the previous outage via R1000 mail. Mail is not sent for normal shutdowns. If the address for Rational support (support@rational.com) is specified with the proper network access, a copy of the full analysis is sent to Rational. By default, this field is blank.

7.7. Miscellaneous Changes to System Management

The Message.Send command no longer sends unnecessary error messages to the sender when wildcard characters are used in the Who parameter. In previous releases, if the Who parameter contained wildcards, an error message would be generated to the sender, but the message would be sent properly. PRS number 9123690-0180-9 and CSR number 6106.

The Abbreviations.Print command now deletes the temporary file it creates when it is used to print a mail message or other image that does not correspond to an Environment object. These temporary files are located in !Machine.Temporary, and in previous releases, were not deleted automatically. PRS number 1033878-Gilg-Cmd.

7.8. Changes to CMVC

7.8.1. Make_Controlled and the State Directory

The Cmvc.Make_Controlled command now works properly if the State directory of a view and/or the State directory's contents are specified. In particular:

- Attempts to control the State directory now fail with the message **The State directory cannot be controlled.**
- Attempts to control objects in the State directory succeed without generating error messages.

PRS number 6289037-Gilg-Cmd.

7.8.2. Destroy_View and Frozen Objects

The Cmvc.Destroy_View now unfreezes access-control files in the Configurations.Configuration_Name_State directory of the view, if necessary. PRS number 6626937-Flor-Jst.

7.8.3. Naming Conflicts when Copying or Building Views

In D_12_7_3, if you attempt to use `Cmvc.Copy` (or a similar CMVC command) to copy a view in which there is a naming conflict between a unit in the view and a unit in the view's imports, the Copy command maintains the internal link to the local unit and generates a warning message describing the naming conflict. In previous releases, if the view to be copied contained such a naming conflict, the Copy command failed, saying that the imported link conflicted with the existing internal link. All copying to that point was be lost. PRS number 2778001-Sier-Geb.

Similarly, in D_12_7_3, if you attempt to build a view from a configuration in which there is a naming conflict between a unit in the configuration and a unit in the new view's imports, the command maintains the internal link to the local unit and generates a warning message describing the naming conflict. The build completes successfully. PRS number 9496712-Wood-Ken.

7.8.4. Default Parameter Values for Make_Spec_View

The default value of the Level parameter of the `Cmvc.Make_Spec_View` procedure has been changed from zero (0) to one (1). The new specification is:

```

procedure Make_Spec_View
(From_Path      : String           := "<CURSOR>";
 Spec_View_Prefix : String           := ">>PREFIX<<";
 Level          : Natural           := 1;
 View_To_Modify  : String           := "";
 View_To_Import  : String           := "<INHERIT_IMPORTS>";
 Only_Change_Imports : Boolean       := True;
 Remake_Demoted_Units : Boolean      := True;
 Goal            : Compilation.Unit_State := Compilation.Coded;
 Comments        : String           := "";
 Work_Order      : String           := "<DEFAULT>";
 Volume          : Natural           := 0;
 Response        : String           := "<PROFILE>");

```

7.8.5. Consecutive Calls to Release

In previous releases, consecutive calls to the `Cmvc.Release` command produced non-sequential release numbering. This has been fixed and release numbers are generated sequentially. CSR number 7334.

7.8.6. Cmvc_Access_Control

The `Cmvc_Access_Control.Check` command no longer fails if it is used on and attempts to update an entry in a full access list (one that contains seven entries). In previous releases, the Check command failed because it tried to add the correct entries before removing the incorrect entries; thus, it would report that there were too many entries and would fail. The Check command now removes the incorrect entries before added the correct entries, and, thus, works successfully. PRS number 3461738-Gilg-Cmd.

When the name of a subsystem is specified to the `View_Or_Subsystem` parameter of the `Cmvc_Access_Control.Check` command, the command now checks the subsystem itself and all

the views in that subsystem. In previous releases, the command would only check the subsystem. PRS number 7842145-Gilg-Geb.

Users that belong to a group with Developer access to a subsystem can now create new objects within that subsystem. In previous releases, only members of groups with Owner access were permitted to create objects within a subsystem. PRS number 8819874-Capi-Rati.

7.8.7. Log Messages

The Cmvcl.Replace_Model command has been made more robust and error messages have been improved. In previous releases, the Replace_Model command would sometimes fail if the View.State.Last_Release_Name file was not formatted correctly and could not be parsed. In such cases, the error messages generated were not helpful. The error messages have been rewritten and the command now handles many minor errors that would have caused it to fail in the past. PRS number 9123690-0153-5 and CSR number 4843.

The messages generated by the Cmvcl.Maintenance.Check_Consistency command now include the actual CMVC database object, rather than just the subsystem, when the command fails because the database for a subsystem cannot be opened. PRS number 5641327-Sier-Geb.

7.8.8. Work Orders

In previous releases, executing Common.Definition on a very large work order failed with a Storage_Error from the work order editor. In D_12_7_3, the work order editor has been improved to handle larger work orders. PRS number 9123690-0199-1 and CSR number 7751.

7.9. Changes to Activity_Implementation

The !Implementation.Activity_Implementation package and its contents are not intended for use by customer-defined programs; they are for use by Environment-defined subprograms and packages. Information about changes to Activity_Implementation, however, are provided on the chance that some sites may have found it necessary to build tools against it.

The Default_Handle function has been removed from the specification of package Activity_Implementation. This function was never meant to be visible and generated unhandled exceptions. PRS number 6527216-Gato-Bas.

In the D_12_6_5 release, the specification of package Activity_Implementation was out of sync with the implementation, resulting in !Lrm.System.Operand_Class_Error messages being generated by some procedures. This has been fixed, and the procedures in package Activity_Implementation should work as advertised. PRS number 9123690-0174-1 and CSR number 6022.

7.10. Networking Changes

In previous releases, some Telnet operations were extremely slow on the Series 400 processor. The slow performance was caused by the CMC Ethernet Controller (new on the 400's) interpreting the bit pattern 16#FF# as a packet terminator and sending one packet for each such bit pattern. This behavior lead to much unneeded packet traffic and even timeouts on the

network. This problem has been fixed; Telnet performance is back to normal. PRS number 3443626-Suno-Ken.

Connections from an R1000 to a UNIX workstation via Ftp.Connect or Telnet.Connect have been made more reliable. In previous releases, the R1000 sometimes echoed the login request to the UNIX workstation when it should not have. The UNIX workstation could not handle this echo, so the connections used to fail. PRS number 9123690-0187-0.

In previous releases, Series 200 or 300 systems would sometimes crash with a !Lrm.System-Assertion_Error from the Tcp_Ip_Driver software. In particular, the system would crash if:

- An application called Transport.Connect to make a passive connection on the network, and
- This connection timed out before completion, and
- The application immediately tried the passive connection again.

This problem was determined to be a timing problem between the R1000 Kernel software and the Network board in the R1000. The Kernel software has been updated to handle this short timing discrepancy and the crash no longer occurs. This was never a problem with the Series 400 because it has a different Network board. PRS number 9057421-Back-Rfg.

7.11. LRM Interface Changes

The D_12_7_3 release of the Environment includes a new release of the LRM Interface. If you are already running Rational Design Facility release 7_2_1, this change will not affect you; the new LRM Interface was delivered with that release.

If you are upgrading to the LRM release included in D_12_7_3, note that changes made to the Declaration_Kinds, Attribute_Designator_Kinds, and Pragma_Kinds enumerations may result in compilation or runtime errors (see subsections 7.11.20 and 7.11.24 for details).

The procedure Find_Lrm_Change_Candidates, located in the Pdl_Commands subsystem, can be used to assist in identifying potential compatibility problems that are not caught by the compiler.

7.11.1. Discriminated Type Function Changes

Type_Information.Is_Discriminated and Type_Information.Discriminants have been changed to work on generic formal types, private types, limited private types and incomplete type. To handle incomplete types, these functions now accept type declarations as well as type definitions.

```
function Is_Discriminated
  (A_Type : Type_Definition_Or_Declaration) return Boolean;
-- This function applies to private, limited private, incomplete or
-- record types. It returns True if this type has discriminants.
-- It may be applied to type declaration to handle the case of
-- incomplete types.

function Discriminants (A_Type : Type_Definition_Or_Declaration)
  return Discriminant_Iterator;
-- Returns a list of discriminants of the type. These elements may
```

```
-- then be manipulated with the functions provided in package
-- Declarations.
```

Note that the name of the formal parameter has been changed from `Record_Type` to `A_Type`. This is because the former was a misnomer. Record types are not the only types with discriminants.

7.11.2. New Aggregate Range Function

The following function has been added to package `Names_And_Expressions`.

```
function Aggregate_Range (An_Aggregate : Expression)
    return Discrete_Range
-- For an array aggregate, returns a range specifying the bounds of
-- the aggregate.Associations
```

7.11.3. Declarations.Kind Change

`Declarations.Kind` has been changed to be consistent with the kind operations of other packages. That is, it now checks the element's major kind and returns `Not_A_Declaration` if the major kind is not a declaration.

7.11.4. New Associations Package

A new package called `Associations` is now available. This package may be used to analyze procedure, entry and function calls, generic instantiations and pragma arguments. The `Associations` package exports the following services:

```
type Association_Kinds is (Named_Association,
                           Positional_Association,
                           Defaulted,
                           Not_An_Association);
```

```
function Association_Kind
    (An_Association : Association) return Association_Kinds;
-- Returns the kind of an association.
```

```
function Formal_Parameter (An_Association : Association)
    return Identifier_Definition;
-- Returns the identifier of the formal name for the given
-- association. This function tries hard to return an identifier
-- definition. However, in the case of a pragma argument, only a
-- reference can be returned.
```

```
function Actual_Parameter
    (An_Association : Association) return Name_Expression;
-- Returns the actual name or expression for the given association.
```

7.11.5. Lexical Typing of Iterators

Lexical typing of iterators has been added. Lexical typing is the practice of using subtype name to reflect the kind of major elements that they can contain. The following declarations have been added to Ada_Program:

```
subtype Association_Iterator is Element_Iterator;
subtype Choice_Iterator is Element_Iterator;
subtype Compilation_Unit_Iterator is Element_Iterator;
subtype Context_Clause_Or_Pragma_Iterator is Element_Iterator;
subtype Declaration_Or_Context_Clause_Or_
  Representation_Clause_Or_Pragma_Iterator is Element_Iterator;
subtype Expression_Iterator is Element_Iterator;
subtype Name_Iterator is Element_Iterator;
subtype Pragma_Iterator is Element_Iterator;
subtype Representation_Clause_Iterator is Element_Iterator;
subtype Statement_Or_Pragma_Iterator is Element_Iterator;
subtype Type_Definition_Iterator is Element_Iterator;
-- Note that some of the iterators can mix items of different major
-- kinds. Their name attempts to convey this information. For
-- instance a declarative part can contain, besides declarations,
-- context clauses (viz. use clauses), representation clauses or
-- pragmas.
```

Many subprogram specifications have been changed, in all packages, to take advantage of the new names. A number of local iterator names have also been introduced to shorten the identifiers or to denote specific constructs.

In Declarations:

```
subtype Declarative_Part_Iterator is
  Ada_Program.Declaration_Or_Context_Clause_
    Or_Representation_Clause_Or_Pragma_Iterator;

subtype Subprogram_Formal_Parameter_Iterator is
  Ada_Program.Element_Iterator;

subtype Generic_Formal_Parameter_Or_Pragma_Iterator is
  Ada_Program.Element_Iterator;
```

In Names_And_Expressions:

```
subtype Aggregate_Component_Iterator is Ada_Program.Element_Iterator;
```

In Representation_Clauses:

```
subtype Record_Component_Clause_Or_Pragma_Iterator is
  Ada_Program.Element_Iterator;
```

In Statements:

```
subtype Declarative_Part_Iterator is
  Ada_Program.Declaration_Or_Context_Clause_
    Or_Representation_Clause_Or_Pragma_Iterator;
```

```

subtype Statement_Part_Iterator is
  Ada_Program.Statement_Or_Pragma_Iterator;

subtype If_Statement_Arm_Iterator is Ada_Program.Element_Iterator;

subtype Case_Statement_Alternative_Iterator is
  Ada_Program.Element_Iterator;

subtype Exception_Handler_Arm_Iterator is Ada_Program.Element_Iterator;

subtype Select_Alternative_Iterator is Ada_Program.Element_Iterator;

```

In Type_Information:

```

subtype Declarative_Part_Iterator is
  Ada_Program.Declaration_Or_Context_Clause_Or_
    Representation_Clause_Or_Pragma_Iterator;

subtype Discrete_Range_Iterator is Ada_Program.Element_Iterator;

subtype Discriminant_Association_Iterator is
  Ada_Program.Element_Iterator;

subtype Discriminant_Iterator is Ada_Program.Element_Iterator;

subtype Record_Component_Or_Pragma_Iterator is
  Ada_Program.Element_Iterator;

subtype Variant_Or_Pragma_Iterator is Ada_Program.Element_Iterator;

```

7.11.6. Compilation Unit Pragmas

The following function was added to package `Compilation_Units`:

```

function Attached_Pragmas
  (To_Compilation_Unit : Compilation_Unit) return Pragma_Iterator;
-- Returns the list of pragmas attached to a compilation unit. Only
-- those pragmas that follow the compilation unit are returned here.
-- Pragmas that precede the compilation unit are part of the context
-- clause.

```

7.11.7. Operations on Generic Instantiations

In `Declarations`, the following operations have been modified so that, when applied to a generic instantiation, they analyze the un-rooted tree to report information of the instance:

```

function Visible_Part_Declarations
  (Package_Specification : Package_Declaration)
  return Declarative_Part_Iterator;
-- Returns a list of all declarations, representation
-- specifications, and pragmas in the visible part of a package in
-- the order of appearance. When applied to a package
-- instantiation, this operation yields the instance's visible

```



```
-- declarations.
```

```
function Private_Part_Declarations
  (Package_Specification : Package_Declaration)
  return Declarative_Part_Iterator;
-- Returns a list of all declarations, representation
-- specifications, and pragmas in the private part of the package in
-- order of appearance. When applied to a package instantiation,
-- this operation yields the instance's private declarations.

function Subprogram_Parameters (Subprogram_Or_Entry : Declaration)
  return Subprogram_Formal_Parameter_Iterator;
-- Returns an ordered list of formal parameter declarations.
-- Use IS_INITIALIZED and INITIAL_VALUE to query
-- the information related to the presence of the default parameter
-- initialization, and use TYPE_MARK to obtain the parameter type mark.
-- When applied to a subprogram instantiation, this operation yields
-- the instance's parameters.

function Return_Type (Of_Function : Function_Declaration)
  return Name_Expression;
-- Returns the name expression of the return type, selectors in
-- NAMES_AND_EXPRESSIONS can then be used to extract more information.
-- When applied to a function instantiation, this operation yields
-- the instance's return type.
```

In Declarations, the following subprograms have been added or modified to improve support of generic instantiations:

```
function Generic_Unit_Declaration
  (Generic_Instantiation_Or_Unit_Declaration : Declaration)
  return Declaration;
-- Returns the declaration of the generic unit being instantiated.

function Generic_Instantiation_Parameters
  (Generic_Instantiation : Declaration) return Association_Iterator;
-- Returns an ordered list of parameter associations of a generic
-- instantiation. The operations defined in package ASSOCIATIONS
-- can be used to decompose them.

function Generic_Actual_Parameters (Generic_Instantiation : Declaration)
  return Association_Iterator
  renames Generic_Instantiation_Parameters;
-- Use of this form is discouraged.
```

7.11.8. Functions and Enumeration Literal Renames

In Names_And_Expressions, the following comments have been added to document the fact that a function call may be a renaming of an enumeration literal:

```
-- FUNCTION CALLS
-- Note that references to enumeration literals renamed as functions
-- are treated as genuine function calls
```

7.11.9. Constants versus Variables

In order to be able to distinguish between constant and variable objects, the following function was added to Names_And_Expressions:

```
function Is_Constant (A_Name : Name) return Boolean;
-- Returns True if the given name is constant. The name must be
-- of a syntactic form suitable for the left hand side of an
-- assignment (ie. not an attribute, a character, etc.).
```

7.11.10. Task Entries versus Subprograms

In consistency with the LRM 6(2), Declarations.Is_Subprogram has been changed to return False on an entry.

7.11.11. Comments in Declaration Names

Declarations.Name no longer return same line comments. It is the equivalent of applying Ada_Program.String_Name to the result of the Declarations.Identifiers.

7.11.12. New Labels Function

The following function has been added to package Statements:

```
function Labels (A_Statement : Statement) return Name_Iterator;
-- Returns an iterator on the names of the labels of a statement. A
-- statement can have several labels.Declarations
```

7.11.13. Limited Private Generic Formal Parameters

The enumeration type in package Declarations that defines generic parameter kinds was changed to handle limited private formal types. The behavior of function Generic_Parameter_Kind has been changed accordingly.

```
type Generic_Parameter_Kinds is (Subprogram,
                                Object,
                                Private_Type,
                                Limited_Private_Type,
                                Discrete_Type,
                                Integer_Type,
                                Floating_Point_Type,
                                Fixed_Point_Type,
                                Array_Type,
                                Access_Type,
                                Not_A_Generic_Parameter);
```

7.11.14. Change to Renaming Declarations Function

Declarations.Is_Renaming_Declaration has been changed so that, if given an identifier definition, it goes to the corresponding declaration. This makes it consistent with all of the Is_@ functions in this package.

7.11.15. Renamed Declarations

The first function below has been added to Declarations. The second has been documented and restricted to avoid returning meaningless information:

```
function Renamed_Name
  (A_Declaration : Declaration) return Name_Expression;
-- Returns the name of the entity being renamed. It can be a simple
-- name, an operator symbol, an indexed component, a slice, a
-- selected component or an attribute.

function Renamed_Declaration
  (A_Declaration : Declaration) return Declaration;
-- If applied to the renaming of a simple name, an operator symbol
-- or an expanded name, returns the name's declaration. Returns nil
-- element otherwise. Use of this function is discouraged.
```

7.11.16. Subunits Without Bodies

Declarations.Unit_Body now returns Ada_Program.Nil_Element when passed the specification of a subunit for which there is no body object in the library rather than raising the exception Inappropriate_Program_Element.

7.11.17. Subunits Without Specs

When Declarations.Specifications is passed a subunit stub or body identifier, it will now return the stub declaration, regardless of whether or not the subunit has a specification.

Previously, this function returned the specification, if one existed for the subunit. In cases where a the subunit had no spec, the stub was returned when passed the identifier from the Is_Separate clause but a Nil_Element was returned when passed the identifier of the subunit body.

This behavior is consistent with LRM 6.3(3). This is also the change with minimal impact, since few programs should depend on the Nil_Element. Note however that this behavior is somewhat inconsistent with Is_Spec, which returns False on a stub.Declarations.Specifications

7.11.18. Declarations.Is_Initialized Change

Declarations.Is_Initialized has been changed to test the nature of its third child, returning True if it is a true initial value, and raising Inappropriate_Program_Element if it is a renaming.

7.11.19. New Function for Incomplete Types

The following function has been added to the Declarations package.

```
function Full_Type_Declaration
  (Type_Declaration_Or_Id : Declaration) return Type_Declaration;
-- Given the declaration of an incomplete type, returns the
-- corresponding full type declaration. A nil element is returned
-- if the full type declaration is not yet compiled. NOTE: this is
-- the identity function if given a non-incomplete type declaration.
```

7.11.20. Implementation-Dependent Attributes and Pragmas

Attribute handling in Names_And_Expressions has been changed to better handle implementation specific attributes. Attribute handling now looks like:

```
type Attribute_Designator_Kinds is (
  Address_Attribute,
  Aft_Attribute,
  Base_Attribute,
  Callable_Attribute,
  Constrained_Attribute,
  Count_Attribute,
  Delta_Attribute,
  Digits_Attribute,
  Emax_Attribute,
  Epsilon_Attribute,
  First_Attribute,
  First_Bit_Attribute,
  Fore_Attribute,
  Image_Attribute,
  Large_Attribute,
  Last_Attribute,
  Last_Bit_Attribute,
  Length_Attribute,
  Machine_Emax_Attribute,
  Machine_Emin_Attribute,
  Machine_Mantissa_Attribute,
  Machine_Overflows_Attribute,
  Machine_Radix_Attribute,
  Machine_Rounds_Attribute,
  Mantissa_Attribute,
  Pos_Attribute,
  Position_Attribute,
  Pred_Attribute,
  Range_Attribute,
  Safe_Emax_Attribute,
  Safe_Large_Attribute,
  Safe_Small_Attribute,
  Size_Attribute,
  Small_Attribute,
  Storage_Size_Attribute,
  Succ_Attribute,
  Terminated_Attribute,
```

```

Val_Attribute,
Value_Attribute,
Width_Attribute,
Not_A_Predefined_Attribute);

```

```

function Attribute_Designator_Kind
  (Attribute : Name) return Attribute_Designator_Kinds;
-- Returns the kind of an attribute. If the attribute is
-- implementation-specific, Not_A_Predefined_Attribute is returned.

```

```

function Attribute_Designator_Name (Attribute : Name) return String;
-- This is the preferred way to analyze an implementation-specific
-- attribute. It returns an uppercase string for the attribute
-- simple name.

```

```

function Attribute_Designator_Name (Attribute : Name) return Name;
-- The Simple_Name returned here is only intended for use by
-- ADA_PROGRAM.STRING_NAME. Use of this function is discouraged.

```

In addition, package Pragmas has been changed to better handle implementation specific pragmas. It now looks like:

```

function Is_Predefined (A_Pragma : Pragma_Usage) return Boolean;

```

```

type Pragma_Kinds is (Controlled,
  Elaborate,
  Inline,
  Interface,
  List,
  Memory_Size,
  Optimize,
  Pack,
  Page,
  Priority,
  Shared,
  Storage_Unit,
  Suppress,
  System_Name,
  Not_A_Predefined_Pragma);

```

```

Unknown : constant Pragma_Kinds := Not_A_Predefined_Pragma;

```

```

function Kind (A_Pragma : Pragma_Usage) return Pragma_Kinds;
-- Returns the kind of a pragma. Returns Not_A_Predefined_Pragma on
-- implementation-specific pragmas.

```

```

function Name (A_Pragma : Pragma_Usage) return String;
-- Returns the uppercase simple name of any pragma. This is the way

```

-- to analyze implementation-specific pragmas.

```
function Arguments (A_Pragma : Pragma_Usage)
  return Association_Iterator;
-- Returns a list of the arguments to a pragma. Operations from
-- package ASSOCIATIONS can be used to decompose them.
```

Note that R1000-specific pragmas have been removed from the Pragma_Kinds. The names of these and other implementation specific pragmas can be used to analyze them.

7.11.21. Return Type of Function Instantiations

Declaration.Return_Type has been changed to yield the instance return type when applied to a function instantiation. Note that this return type will be a subtype renaming the true actual type.

7.11.22. Block and Loop Names

The following functions have been added to package Statements in order to query for loop and block names.

```
function Is_Named_Statement (A_Statement : Statement) return Boolean;
-- Returns true if applied to a loop or block that has a name.
```

```
function Statement_Name (A_Statement : Statement) return Name;
-- Returns the name of a block or loop. Returns Nil_Element if not
-- a block or loop, or if no name is present.
```

7.11.23. Enumeration Literal Queries

In Names_And_Expressions, Is_Literal, Position_Number and Representation_Value have been changed to work on an enumeration literal specifications. Previously, they worked only on a reference to an enumeration literal.

7.11.24. Record Discriminant and Component Declarations

The following literals have been added to type Declaration_Kinds in order to differentiate record components from variables.

```
A_Discriminant,
A_Record_Component,
```

Ada_Program.Kind and Declarations.Kind have been modified to return these new values when appropriate. Declarations of these kinds can be analyzed by Is_Initialized, Initial_Value, and Object_Type.

8. Documentation

The D_12_7_3 release of the Rational Environment is accompanied by new and updated printed and online documentation.

8.1. Printed Documentation

This D_12_7_3 release is accompanied by two printed manuals:

- An up-to-date *System Manager's Guide* for the Rational R1000 Software Development System, Series 200, 300, and 400. This document has been tested for accuracy with the D_12_6_5 release, reorganized, and presented in a new, easier-to-read format with red tabs marking emergency procedures. The *System Manager's Guide* is product number 4000-00142.
- The *Rational Access User's Guide*, for use with the Rational Access user interface. This document is product number 4000-00722.

You should receive one copy of each manual for every system under an active support contract. If you have not received your new documentation or if you would like to order additional documentation, please contact your Rational representative.

8.1.1. Correction to *Quick Reference for Parameter-Value Conventions*

On the second page of Appendix F, "Quick Reference for Parameter-Value Conventions," in the *System Manager's Guide*, the table describing special names for default objects describes the special name "<ACTIVITY>" as resolving to the "default activity for the current session." This is incorrect. "<ACTIVITY>" resolves to the current activity, which is the activity named in the job response profile for the current job. Unless you have explicitly set the activity for the job, the activity is the same as the default activity for the current session.

This correction also applies to the versions of the "Quick Reference for Parameter-Value Conventions" in the Session and Job Management, Library Management, and System Management Utilities books of the *Rational Environment Reference Manual*.

8.2. Online Documentation

This D_12_7_3 release of the Environment includes new or updated online documentation for:

- Rational's online help system. To display this information, press [Help on Help] or execute `What.Does ("Help_On_Help")`.
- The packages described in the Debugging (DEB) book of the *Rational Environment Reference Manual*:
 - Package Debug
 - Package Debug_Maintenance
 - Package Debug_Tools
- The packages described in the Project Management (PM) book:
 - Package Activity

- Package Check
- Package Cmvc
- Package Cmvc_Access_Control
- Package Cmvc_Hierarchy
- Package Cmvc_Maintenance
- Package Work_Order

An effort has been made to bring the online documentation for the Debugging and Project Management packages up-to-date with the D_12_7_3 release of the Environment. The new online help for these packages also contains parameter-level information, examples, and cross references previously not available.

This D_12_7_3 release also includes online PostScript versions of the keyboard overlays for the Rational X Interface. These overlays are located in !Machine.Editor_Data.Keyboard_Overlays. To print an overlay from the Environment, specify the print option Original_Raw.

9. Training

Rational is currently in the process of updating the standard Environment training courses to reflect this D_12_7_3 release of the Environment and to incorporate the paradigm changes introduced by the Rational Access user interface. Rational will publicize the availability of these courses when they are complete.

Appendix A

Problem Reports Closed in D_12_7_3

This appendix lists the software problem reports and customer-service requests closed by the D_12_7_3 release.

Table A-1 lists the problem reports fixed by D_12_7_3. It includes the Problem Reporting System (PRS) number of the problem, the Customer Service Request (CSR) number (if applicable), a brief description of the problem, and the section of this release information in which the problem is discussed in more detail (when applicable).

Table A-1 Software Problems Fixed by D_12_7_3

PRS Number	PRS Summary	Section
none	CSR7334: Consecutive calls to Cmvc.Release product non-sequential numbering	7.8.5
10464-Star	What.Locks fails if applied to a deleted unit	7.2
12612-Star	Library.Space gets Numeric_Error	7.5
12728-Star	What.Object does the wrong thing with non-directory objects	7.2
12862-Star	Semantics blows up when rep spec applied to incomplete type	7.4.1
0-0298-0	Archive creates subsystems and worlds within subsystems	7.3.1
167263-Clem-Marl	What.Object on pointy file fails; <IMAGE> is not defined	7.2
426364-Shei-Jst	Incremental operations give different results	7.4.4
523289-Sier-Geb	Access_List.Set@ fails to put Response parameter in log	7.5
747500-Gato-Rjg	Tape Eject behavior inconsistent	7.3.5
1005010-Etoi-Ksch	Archive.[Save,Restore] changes venture fields	7.3.4
1033878-Gilg-Cmd	Abbreviations.Print doesn't delete temporary files	7.7
2115837-Etoi-Phl	Selection sometimes fails in Environment menus	7.1.1
2778001-Sier-Geb	Cmvc.Copy halts when there is a naming conflict	7.8.3
2911238-Jude-Geb	Compilation messages should not have 'V()' in them	7.4
3032909-Cook-Swb	R1000 code bug - type completion for access types	7.4.2
3078226-Gato-Mboy	Overloading of name of package System and variable name causes confusion	7.4.1
3108823-Sier-Geb	Semantics error on renaming a field in a constant	7.4.1
3443626-Suno-Ken	Poor Telnet performance on model 400 processor	7.10
3461738-Gilg-Cmd	Cmvc_Access_Control.Check fails when there are too many extra entries	7.8.6
3496838-Gato-Rjg	Erroneous message in semantically inconsistent code	7.4.1
4290910-Sier-Geb	R1000 code generator generates code that raises Type_Error at run time	7.4.3
4317752-Clem-Marl	What.Object fails when cursor is in selection in text image	7.2

Table A-1 Software Problems Fixed by D_12_7_3 (continued)

PRS Number	PRS Summary	Section
4711901-Rati-Pbk	Ambiguity undetected by semantics	7.4.1
5098655-Mago-Sdj	Problem searching in an image another job has open	7.1.2
5476876-Voya-Phil	What.Locks fails when given a deleted object	7.2
5641327-Sier-Geb	Cmvc_Maintenance.Check_Consistency generates unhelpful message	7.8.7
5844095-Aria-Sbur	Incorrect use of discriminant not detected	7.4.1
6289037-Gilg-Cmd	Cmvc.Make_Controlled generates bogus message when controlling objects in State directory	7.8.1
6370093-Etoi-Phl	R1000 compiler allows deferred constant for private type	7.4.1
6527216-Gato-Bas	Activity_Implementation.Default_Handle blows up in D_12_1_1	7.9
6626937-Flor-Jst	Problem with Cmvc.Destroy_View and frozen access control files	7.8.2
6969167-Gato-Gbd	Unimplemented: String literal has non-static size	7.4.2
7762494-Rati-Pbk	Ambiguous expression not detected by semantic analysis	7.4.1
7842145-Gilg-Geb	Cmvc_Access_Control.Check on a subsystem does not check view	7.8.6
8391944-Gato-Gbd	Can't semanticize record rep clause with long_integer base t	7.4.1
8480408-Gato-Gbd	Renamed enumeration literals can be static	7.4.1
8819874-Capi-Rati	Cmvc_Access_Control: Developer group can't create new objects	7.8.6
8957642-Cook-Swb	Problem with operations on image before editor is done displaying image	7.1.2
8995225-Shei-Jst	Problem with selection in Environment menus	7.1.1
9031863-Gato-Gbd	Exceptions not identified when propagated to Environment	7.4.3
9057241-Back-Rfg	Transport.Connect problem with series 200 and 300	7.10
9072739-Shei-Swb	Semantic error message deterioration in new release	7.4.1
9123690-0153-5	CSR4843: Cmvc.Replace_Model causing error	7.8.7
9123690-0158-2	CSR5079: Archive.Copy corrupts existing Load_Proc on failure	7.3.2
9123690-0174-1	CSR6022: Operand_Class_Error on Activity_Implementation call	7.9
9123690-0180-9	CSR6106: Message.Send problem	7.7
9123690-0183-5	CSR6426: Problem with Archive.Copy and searchlists	7.3.3
9123690-0184-6	CSR6625: Ada compilation discrepancy	7.4.1
9123690-0187-0	Ftp.Connect/Telnet.Connect to UNIX fail due to poor handshaking	7.10
9123690-0194-1	Delete access required for creating Ada units	7.5
9123690-0199-1	CSR7751: Storage error with Definition on work orders	7.8.8

Table A-1 Software Problems Fixed by D_12_7_3 (continued)

PRS Number	PRS Summary	Section
9496712-Wood-Ken	Problem with Cmvc.Build and import conflicts	7.8.3

Table A-2 lists the problem reports that have been investigated and closed because of one of the following reasons:

- The problem could not be reproduced in D_12_7_3.
- The problem has been fixed in a previous Environment release.
- There is not enough information about the problem to proceed.

Table A-2 Not Reproducible in D_12_7_3

PRS Number	PRS Summary
6956-Star	Directory_Uilities.Get_Containing_Subsystem
7689-Star	Archive says "moved" when it means "copied"
8286-Star	Bad messages from Compilation.Demote
8817-Star	Definition on deleted object gave Version_Error
9568-Star	Tape.Write doesn't know when it hits end of tape
9651-Star	Library.Resolve gets unhandled exception
9923-Star	Common.Complete in installed unit
10136-Star	Mail.Send_Message assumes string parameters start at indice = 1
10224-Star	Common.Edit in an activity without a selection fails
10246-Star	Suggest adding more functionality to 'C naming attribute
10391-Star	Tape always unloads even if subsequent read is desired
10491-Star	Directory_Tools.Naming.Ada_Name always returns null string
10723-Star	Should be default sort order switch for window directory
10863-Star	Inconsistency in Directory.Naming
10974-Star	Common.Format fails after expunging mailbox
11048-Star	System_Report.Generate produces meaningless report
11132-Star	Job.Kill crashed machine
11306-Star	Window directory enhancement
12380-Star	Common.Object.Delete of a mailbox failed with Nonexistent_Page_Error
12387-Star	Mail_Internal_Error trying to view unread mail message
12637-Star	Expunge on a non-main mailbox fails
12673-Star	Infinite loop in simplification of 'Width
12722-Star	Syntactic completion not as smart as it should be
12766-Star	Incorrect 'Length
0-0377-0	Strange bug in R1000 compiler
2668045-Wood-Rfg	Implicit exception reported raised at unknown location
2775544-Zebr-Lore	Inappropriate warning regarding limited private types
295703-Cook-Rcp	Debugger displays incorrect values for enumeration types
320545-Voya-Jim	Debugger doesn't work well with Definition

Table A-2 Not Reproducible in D_12_7_3 (continued)

PRS Number	PRS Summary
4348467-Gato-Mboy	Error when private type address is an access type
7178409-Blut-Smp	Cmvc.Show_Image_Of_Generation profile somewhat ignored
7509601-Nati-Drk	Debuggers don't kill themselves properly
7656507-Voya-Phil	Show_Tasks gets exceptions
8757197-Shei-Swb	Problem with Common.Complete
9123690-0140-3	Deleted unit shows up as checked out in CMVC database
9123690-0158-9	CSR5226: Cmvc problem
9123690-0171-1	CSR5649: Cmvc.Import fails with incompatible target keys
9123690-0175-7	CSR6118: Compute_Recoding spews error messages
9123690-0181-1	CSR6122: Anonymous Ada units causes some CMVC operations to hang
9123690-0181-2	CSR6163: Anonymous units cause some CMVC operations to hang
935209-Mago-Trw	Definition does not accept special names

Exabyte Operations and Ordering Information

1. User/Operator Maintenance

In new system shipments or upgrades containing an Exabyte tape drive, Rational includes a blank 8mm tape cartridge and a 3-pass, approved cleaning kit.

The Exabyte unit requires low maintenance, however, it is dependent on the proper care and handling of the transport and magnetic tape. The cleaning procedures outlined in the Exabyte Cleaning Kit are brief and require only minutes of the operator's time, but cleaning must be done as explained in order to achieve continued reliability and low maintenance.

The User/Operator of the Exabyte tape drive is responsible for performing a periodic tape head/path cleaning. The cleaning of the tape head/path is accomplished with the Exabyte Cleaning Cartridge Kit. The cleaning kit contains the instructions and all items required to perform the tape head/path cleaning operation. The kit which Rational has included in your shipment is good for 3 passes. After the 3rd cleaning, the tape should be discarded. The recommended frequency for the cleaning is after 80 Gigabytes of data transfer or monthly, whichever occurs first.

Reliability

To maximize reliability, please observe the points outlined in this section. Please note that failure to follow these recommendations or requirements, may result in voiding manufacturing and/or support warranties.

- Use only Exabyte's brand 8mm tape cartridge or the Sony, video 8, metal particle, data grade tape cartridge. (recommendation)
- Maintain proper environmental conditions for the tape drive and the media. The drive or media should not be operated or stored in areas where there is excessive heat or cold or in areas where temperatures may change at a rate of more than 10% (Fahrenheit) per hour. Also, the relative humidity should be between 20% and 80% non-condensing. (recommendation)
- Use only approved cleaning tapes. (Exabyte part no. 727113 or 727114) Unapproved video store cleaning tapes may damage the heads or transport. (requirement)
- Avoid using faulty or damaged cartridges/media. (requirement)

Ordering Information

Orders for Exabyte-brand tapes and approved cleaning kits may be placed by calling Exabyte's sales outlet at 1-800-767-8273. (For outside U.S., 1-303-442-4333 or 1-303-447-7613)

Part Number	Description
180179	Exatape, 15 Meter
180180	Exatape, 54 Meter
180181	Exatape, 112 Meter
727113	Exabyte Cleaning Tape, 3-pass
727114	Exabyte Cleaning Tape, 12-pass

)


)

)

SMSE Checklist

DISK_400

PRODUCT REQUIREMENTS

	<u>PART NUMBER</u>	<u>DESCRIPTION</u>
	507-003274-004	Installation Procedure for Series 400S

SERIES 400 SYSTEM

Installation Procedure

-
- System Installation
 - Disk Upgrade
-

RATIONAL

Copyright © 1991 by Rational

PN 507-003274-004

This document is subject to change without notice.

Rational and R1000 are registered trademarks and Rational Environment is a trademark of Rational.

Rational, 3320 Scott Boulevard, Santa Clara, California 95054-3197

Table Of Contents

1 OVERVIEW	1
2 PRE-INSTALLATION	3
Prerequisites For Installation	3
3 INSTALLING THE SYSTEM	5
Series 400S System Configuration	5
Unpacking and Inspection	7
Final Installation Checklist and Precautions	8
Checking for Proper Routing and Connection of Cables	9
Preparing to Apply System Power	11
Power Checkout	13
Installing the Operator Console	15
4 SYSTEM TESTING AND OPERATIONS	17
System Checkout Procedures	17
Modem and Networking Connections	18
Configuring the System for Remote Diagnostics (optional)	18
System Peripheral Exercisers	18
Installing the Environment Terminal (optional)	21
Configuring the Rational Environment	22
Overall System Operation	23
5 OTHER SUPPORT	25
Appendix A R1000 SERIES 400S SPECIFICATIONS	27
Appendix B RATIONAL INSTALLATION CHECKLIST	29
Appendix C UPGRADE 1.2 GB DISK DRIVE	35
Appendix C Preparation	35
Appendix C Installation	35

1

OVERVIEW

These installation instructions describe the installation procedures for a Series 400 System as shipped from the factory. In addition, an appendix to these instructions describes the installation procedure to follow when upgrading a previously installed Series 400 System with additional disk drives.

2

PRE-INSTALLATION

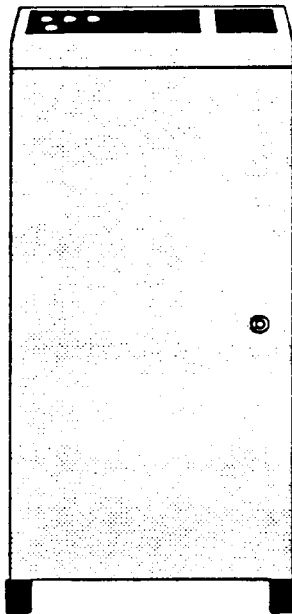


Figure 1.1 - Series 400 System

Prerequisites For Installation

Pre-installation planning is covered in the Rational "Site Planning Guide for the R1000 Series 400 Development System."

Pre-installation planning must be completed and an installation plan established prior to the system being delivered to the installation site. Pre-installation planning covers the following areas:

- Planning responsibilities and scheduling
- Site selection
- R1000 components with physical specifications and service clearances
- Environmental and electrical requirements

- Data communications line requirements
- Delivery preparation

The Implementation Plan should be completed at least *30 days* before the R1000 system delivery. Refer to the section "Implementation" in the *Field Operations Handbook* for a sample of information to be included in this implementation plan.

The implementation plan should be distributed to the following Rational representatives involved in planning, installation, project guidance, training, and post-installation support of the R1000 system:

- Field Technical Representative or Response Center support personnel responsible for the software and hardware installation, checkout, and initial training of the customer's System Manager and operations personnel.
- Marketing Representative responsible for assisting with coordination of system configuration, delivery dates and other sales-related issues.
- Field Technical Representative or home office Technical Consultant responsible for providing product training, special technical consulting in the areas of software engineering, the customer's program applications, and other specialized training to meet the customer's need.

3

INSTALLING THE SYSTEM

Having completed all of the necessary prerequisites, the Series 400S system is now ready to be installed. Refer to the diagrams as you perform the installation steps.

Table 1.1 below summarizes the activities that must be performed for a successful installation of the series 400. The activities that contain * are covered in much greater detail in the following sections.

Table 1.1 - Rational Site Preparation and Installation

Timetable	Installation Activity
Day 1	Unpack, take inventory, and perform an inspection for visible damage that may have occurred during shipment* Position the 400S cabinet over floor tiles* Install the cabinet's stabilizing brackets and lock wheels in place Install the disk drives as specified on the disk housing cover Connect the necessary terminal and network cables Perform pre-power checkout at customer power receptacle* Perform power-on and off-line checks* Execute first level diagnostics Boot system and ensure proper version of Environment software has been installed. Ensure that all Rational software products ordered, are installed. Begin overnight run of system diagnostics
Day 2	Analyze the overnight run of system testing Perform remote diagnostic link check to the Response Center (if required) Begin start-up training for customer's System Manager and operations personnel—1/2 day. (The System Manager receives two additional days of training, following three days of Rational Product training) Turn system over to customer and submit the completed copy of the support activity report to the Rational Customer Support Response Center

As you are installing the R1000, you should complete the installation checklist in the appendix on page 29, **Rational Installation Checklist** of this guide.

Series 400S System Configuration

As shown in Figure 1.1, a Series 400S consists of a single cabinet.

The 400S cabinet contains (See Figure 1.2 and Figure 1.3):

- R1000 Processor and Memory Boards
- RESHA I/O Controller board
- DC Power Supply
- System Control Panel
- 8mm Tape Drive
- Disk Housing containing 1 (std) to 4 (opt) Disk Drives
- Console and Single RS232 Comm Port
- AC Power Distribution Unit

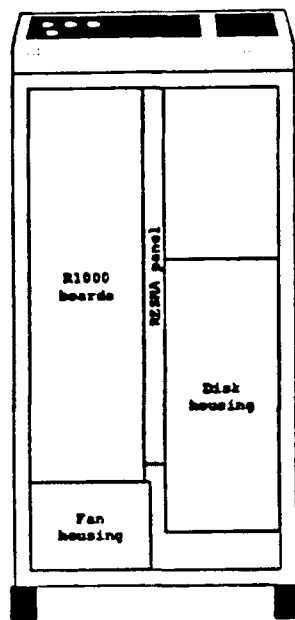


Figure 1.2 - Series 400S, Front View

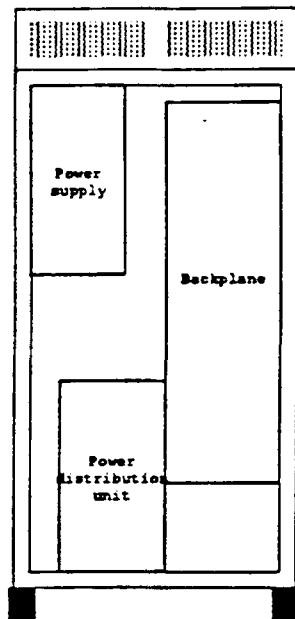


Figure 1.3 - Series 400S, Rear View

Unpacking and Inspection

Upon arrival of the R1000 system at the customer's site, the customer is responsible for moving the system from the shipping area to the installation area. The Rational representative is responsible for installing the system and should perform the following steps:

- ☐ 1. Remove all protective-packing material.
- ☐ 2. Check the shipment against the packaging list.
- ☐ 3. Check all boxes and system components for external damages.
- ☐ 4. Record discrepancies observed while checking the shipment and report them to the customer and to Production Control at Rational's home office. Production Control will help coordinate the effort to resolve discrepancies.
- ☐ 5. Document in the installation report observations relating to damaged items. The customer should be notified and Production Control contacted for instructions on how to handle a damaged item.

Final Installation Checklist and Precautions

Check through the following list of final installation preparations and precautions:

- ☐ 1. Make complete visual checks of cables, harnesses, backplane, foreplane, fans, filter pads, and all cable connectors for crushed wires, cut wires, bent or smashed pins, or any other abnormality.
- ☐ 2. Make sure you have the minimum tools required for installing the Series 400S system. These tools are contained in the basic tool kit which is issued to Rational support personnel.
- ☐ 3. If there is no raised floor, cables routed externally to the system's cabinet must be protected from the walking path of personnel. To protect these cables, the customer should use cable through or route them in overhead channels.
- ☐ 4. Make sure you have the stabilizing brackets that are required to be installed on the cabinet's wheels. The brackets must be installed to satisfy the anti-tip requirement specified by the Underwriters Laboratory. The brackets will be included with each system shipment.
- ☐ 5. Make sure that the disk drives are installed as specified on the back of the disk housing cover. Unlike previous R1000 systems, the disk drives in the series 400 are SLOT-DEPENDENT, i.e., the drive for unit 0 must be in the top slot of the disk housing, unit 1 must be in the 2nd slot from the top, etc.

Prior to shipment, the disk drive configuration is recorded on the back of the disk housing cover, using the serial number of each drive. The serial number of each drive appears on the front and back of the drive.

Check that the serial numbers specified on the list on the back of the disk housing cover match the serial numbers of the drives delivered with the system

and install them in the order specified on the list. Report any discrepancies to Production Control at the home office before proceeding.

Checking for Proper Routing and Connection of Cables

- 1. Position the 400S system cabinet in the computer area according to the layout plan established during the pre-site planning stage. (Refer to Figure 1.4.)

Note: Once the cabinet is in place, be sure to install the stabilizing brackets over each wheel as specified in the diagram that is included with the bracket package. The brackets are shipped in the crate with the system.

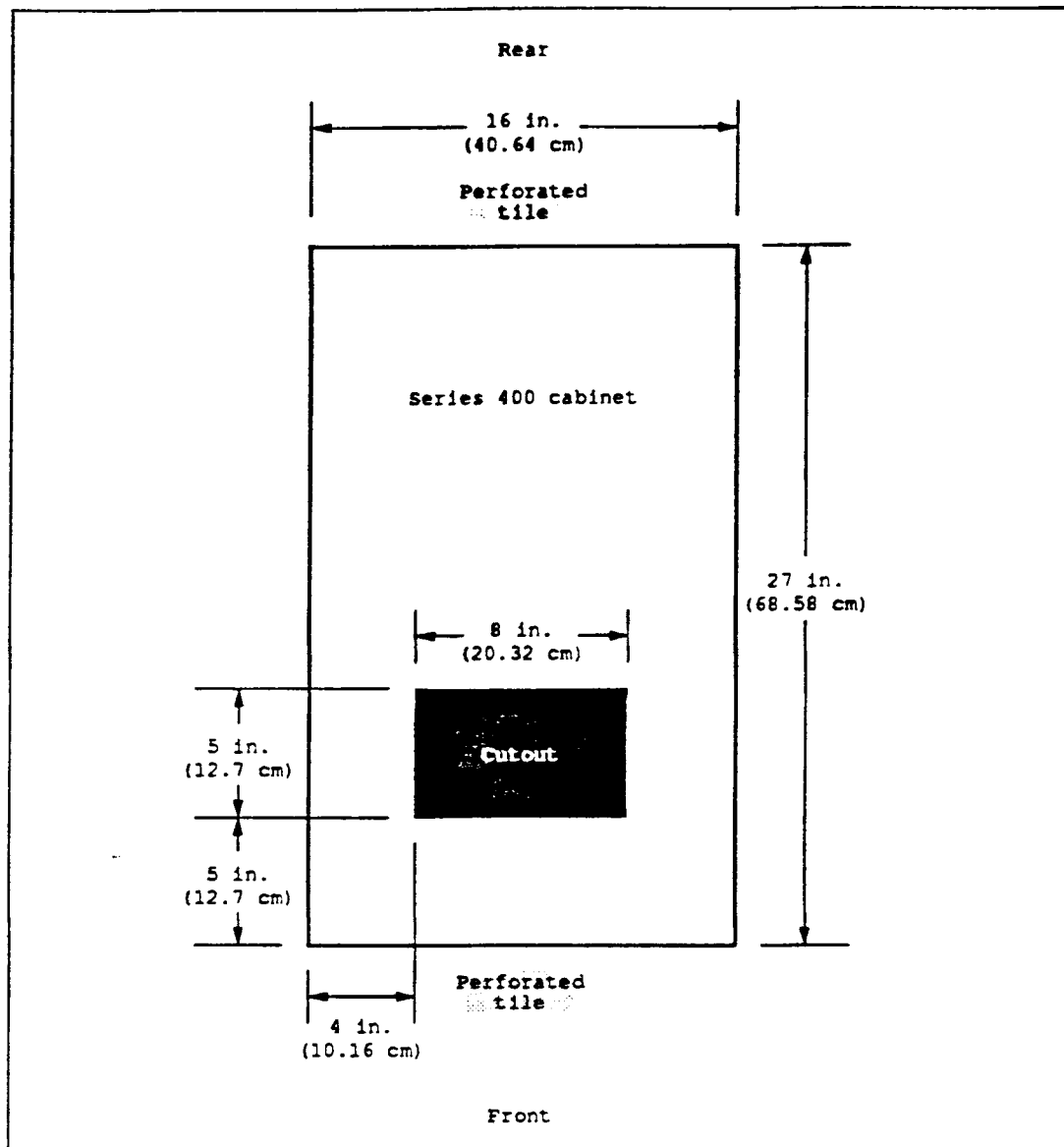


Figure 1.4 - Floor Tile Requirements for R1000 Series 400S Installation (Top View)

- 2. From the front of the 400S cabinet, ensure that the following cables are connected at the RESHA board face plate: Refer to Figure 1.5 for connector location.

- Cartridge Tape Drive
- Control Panel
- Disk Drives

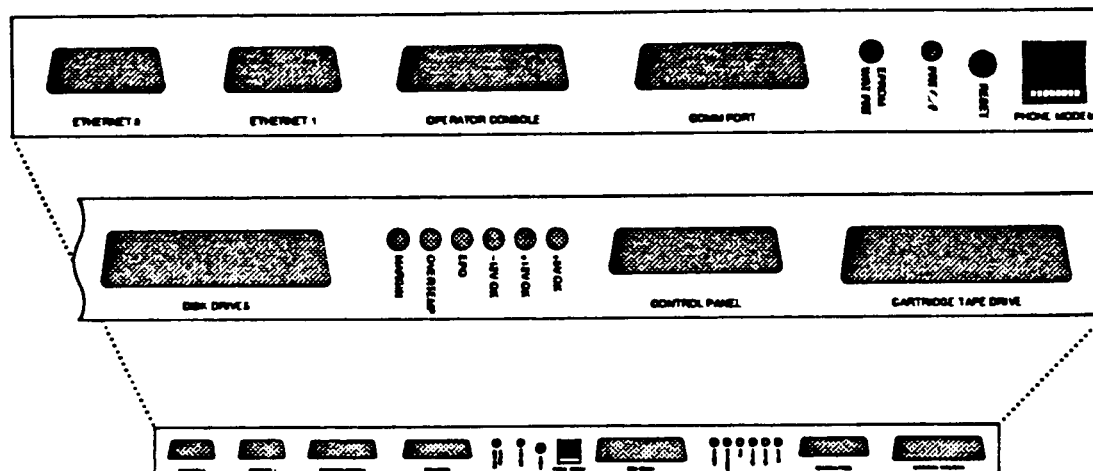


Figure 1.5 - RESHA Board Face Plate

- 3. From the rear of the 400S cabinet, check for cable connections on the DC power supply and card cage backplane:
- power cable (part no. 125-003168) on the end of power supply to PS0 power jack on the Power Distribution Unit (PDU)
 - P.S. Control cable (part no. 125-003180) from connector J9 on the upper left corner of the card cage backplane to the two molex connectors located underneath the left side of the power supply.
 - cable (part no. 125-003472) from terminal strip located on the power supply to the card cage backplane connector (J12) labeled +/- 12V IN.
 - cable (part no. 125-003473) from the molex connector on the rear of the disk cage to the card cage backplane connector (J13) labeled Disk Power.
 - cable from the card cage backplane connector labeled Exabyte Power (part no. 125-003175) and routed over the top of the card cage toward the front where the Exabyte Tape Drive is installed.
 - cable from the card cage backplane connector labeled +12 Fans (part no. 125-003174) and routed down the edge of the card cage and underneath to the fan cage located in the front of the 400S cabinet, underneath the cardcage.

Preparing to Apply System Power

Perform the following steps prior to applying AC power to the system:

- ☐ 1. Verify circuit breakers CB1 and CB2 are in the *OFF* position on the Power Distribution Unit (PDU). (Refer to Figures 1.3 and 1.7 for the location of the PDU and circuit breakers).
- ☐ 2. Verify the power keyswitch is in the *Standby* position, disk write keyswitch is *Protected* and the operator mode is in *Auto* position at the main system control panel. (Refer to Figure 1.6 for keyswitches and status indicators).

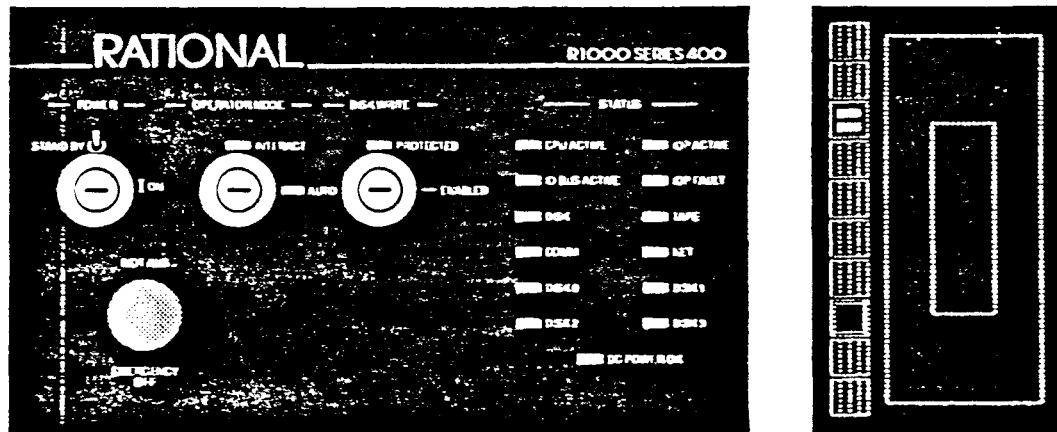


Figure 1.6 - System Control Panel with Exabyte

- ☐ 3. Verify the RESHA board is seated properly and the twist locks are secured.
- ☐ 4. Verify the R1000 processor and memory boards are seated properly--foreplane is in and the cover is installed.
- ☐ 5. Verify all disk drives are seated properly and in the order specified on the back of the disk housing cover, and the cover is installed.
- ☐ 6. Feed the main AC power cord through the cutout in the floor tile closest to the back of the PDU. (Refer to Figure 1.3 for location of PDU).
- ☐ 7. Ensure that the main AC power cord is connected at the bottom front of the PDU

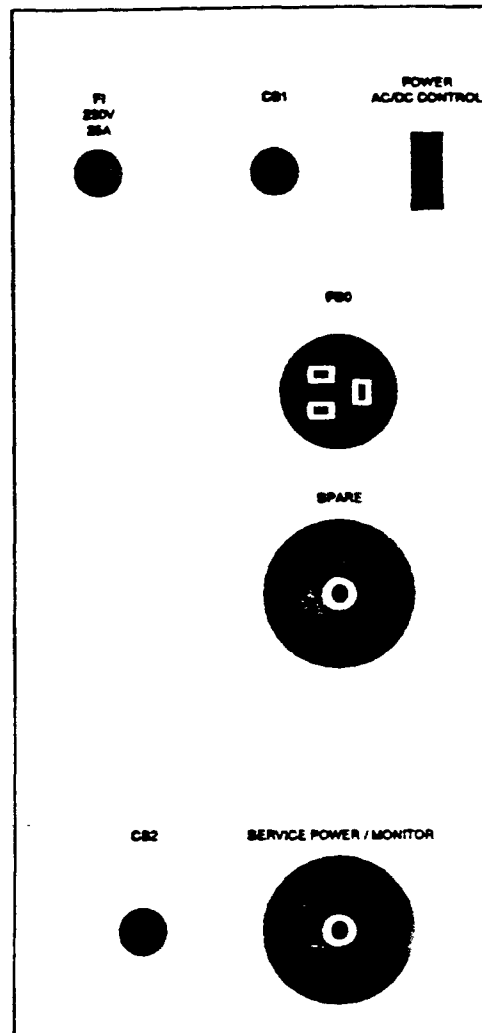


Figure 1.7 - Series 400S, Power Distribution Unit

Power Checkout

Check the manufacturer's name plate for the power requirements of the system being installed (Located at the bottom, right-hand corner of the rear door). Verify the requirements before proceeding with the following steps, referring to Figure 1.7:

- ☐ 1. Ensure that the circuit breakers and power key switch are in the *OFF* or *Standby* position for the following:
 - The Power Distribution Unit—PDU (lower rear of cabinet)
 - The main system power (key-switch on System Control Panel)
- ☐ 2. Verify the customer's power source circuit breakers are *ON*. Using a voltmeter set to AC volts, measure the AC power at the receptacle. The voltage measured

between the two hot leads should measure between a minimum value of $208 \pm 5\%$ VAC to a maximum of $240 \pm 5\%$. If these voltages are not correct, do not proceed with the installation until the facility's electrician has corrected the problem.

- ☐ 3. Connect the power cord to the mating connector originating from the customer's power source.
- ☐ 4. Turn on circuit-breaker CB2. Using a voltmeter, check the AC power for the proper input voltage. The voltage should be between a minimum value of $208 \pm 5\%$ VAC to a maximum value of $240 \pm 5\%$. Take this measurement at the power jack labeled "Service Power/Monitor" located near the lower, right-hand edge of the PDU. (Refer to Figure 1.8 for pin assignment).

Note: In Europe there is 1 phase, 1 neutral, and one ground pin.

The black cap unscrews from the power jack, providing access to the pins (see Figure

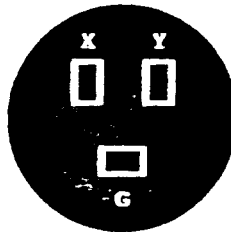


Figure 1.8 - Series 400S, PDU Plug Pinout

- ☐ 5. Turn on circuit-breaker CB1. Switch circuit-breaker CB2 to *OFF*. (CB2 should remain off during normal system operation).
- ☐ 6. Switch the DISK WRITE keyswitch on the System Control Panel to the "PROTECT" position.
- ☐ 7. Switch the OPERATOR MODE keyswitch on the System Control Panel to the "AUTO" position.
- ☐ 8. Switch the POWER keyswitch on the System Control Panel to the "ON" position—this turns on the AC power. All fans should start blowing. This also allows the DC power supplies to apply DC voltage to the CPU, controller boards and disk drives.
- ☐ 9. Switch the DISK WRITE keyswitch on the System Control Panel to the "ENABLED" position. Allow drives to run for a period of time to adjust to the room temperature.

- ☐ 10. Measure DC voltage for the DC power supply (located above the PDU) for the following values: (the measurements may be taken at the top side of the fuses located near the outer edge of the backplane)
 - a. +5 VDC
 - b. +12 VDC
 - c. -12 VDC
- ☐ 11. Verify that all fans are operational. (1 on the disk housing, 2 on the cardcage)

Installing the Operator Console

Note: At least one terminal is required to boot the Series 400S. The terminal must be an ANSI standard VT100 console and links to the Operator Console port (Refer to Figure 1.5). As Rational does not supply Rational terminals with the Series 400S, the customer should make sure an ANSI standard terminal is available at installation time.

- ☐ 1. Place the operator console in the location designated on the installation floor plan. Be sure there is a small cutout in the tile near the terminal. This cutout is for the power cord and the RS232 data cable for this terminal. The terminal requires 3-prong, 115 VAC power receptacles (domestic).
- ☐ 2. Install the operator console cable in the connector marked "Operator Console". The connector is located on the RESHA face plate. (Refer to Figure 1.5). This cable should be routed down the outside edge of the disk housing, out the bottom of the cabinet, and fed through the raised-floor tile that has been cut out for terminal and network cables. Route the cable as directly as possible to where the operator console is to be located. Feed the end of this cable back up through the tile cutout nearest the operator console and connect it to the terminal.
- ☐ 3. Set up the console's operating parameters:
 - Transmit Baud Rate: 9600
 - Receive Baud Rate : 9600
 - Parity : SPACE
 - Check Parity : OFF
 - Duplex : FULL
 - DTR Handshake : ON

After the installation of the Series 400S the operator console will be used to allow the system manager to communicate with the layers of software that support the Rational Environment™, the principal software component of the Series 400S.

4

SYSTEM TESTING AND OPERATIONS

Testing the system should begin with running the individual FRU tests and the functional diagnostics which are executed at the DFS level. Once this has been completed, the system may be booted and additional tests will be executed at the system's EEDB level.

System Checkout Procedures

Initiate the standard boot process. If the Operator Mode keyswitch is in the AUTO position, you will be asked if you want to boot the [STANDARD] configuration. Answer this question by depressing the "control" and "c" keys simultaneously. This will take you to the CLI> prompt.

At the CLI> prompt enter **x FRU** to get to the FRU Main Menu. At this point, select option 3 (Execute diagnostics) and answer the rest of the questions to execute the phase-2 tests for each board type:

- ☐ 1. IOC
- ☐ 2. VAL
- ☐ 3. TYP
- ☐ 4. SEQ
- ☐ 5. FIU
- ☐ 6. MEM 0
- ☐ 7. MEM 2 (if present)

Run the FRU confidence test (microdiagnostic) after the phase-2 tests if the phase-2 tests do not catch anything. Note that this can now be run from the FRU main menu (option 2).

Modem and Networking Connections

You may need to make final hardware connections in the following areas:

- ☐ 1. For customers with standard support, connect the customer-supplied modem to the "Comm Port"---if the port is currently being used to access the Environment, disconnect the Environment terminal and reconnect it to the "Operator Console" port (see the section "Installing the Operator Console").
- ☐ 2. If the customer has already installed networking, connect the public Ethernet cable to the R1000 at the Ethernet 0 receptacle located as the second connector from the bottom of the RESHA board face plate. (Refer to Figures 1.2 and 1.5)

Configuring the System for Remote Diagnostics (optional)

The proper use of the Comm Port is determined by the INITIOA program in the DFS. This program establishes the remote diagnostics connection to be used in case such a connection is allowed and needed. A remote diagnostic connection can be made via the modem on the RESHA board. If the system is configured to use the Modem on the RESHA board then the Comm Port port is automatically configured to the Environment port 16.

X INITIOA is used to set the Cluster Id for the machine and as the data needed for remote diagnostics. These instructions assume that the RESHA board modem will be used and the Comm Port will be used as an Environment port.

Perform the following steps:

- ☐ 1. Contact the Rational Response Center and inform them that you plan to test remote diagnostics. Verify with them that the diagnostic phone number is: 1-800-841-6400. This number has already been set in the novram of the IOC board and the number is only valid for UNITED STATES and CANADA.
- ☐ 2. Run `x INITIOA` from the `CLI>` prompt.
- ☐ 3. Verify the Cluster Id and modify it if necessary. (check the manufacturer's tag at the bottom of the rear door)
- ☐ 4. Verify the type to be used for Remote diagnostics and correct it if necessary. If no remote diagnostic connections will be allowed, make sure that the type is set to "M" for modem. This will allow the Comm Port to be used as Environment port 16.
- ☐ 5. Verify the data needed for the remote diagnostics connection and correct it if necessary.

System Peripheral Exercisers

Run the following peripheral exercisers from the CLI> prompt:

□ 1. Tape Exerciser

Enter **x tapex** at the CLI> prompt. After confirming the unit number of the tape drive to exercise, it will start running and continue to run until you explicitly stop it with a CTRL/G. Pressing any other key will give you the current status: how many bytes have been transferred, and the numbers of soft, hard, and data errors.

Let the program run for about 5 minutes or until 1 megabyte of data has been transferred. Check the status every minute or so. There should be no errors reported. Sample output:

```
CLI> x tapex
Exercise unit 0 [Y] ? y
DFS based tape exerciser, started at =>10:49:56 28-FEB-91
Type ^G to exit, any other character for status
[space bar]
Status at 10:50:08 28-FEB-91, test started at 10:49:56 28-FEB-91
Unit : 0, Bytes => 23064      , soft => 0    , hard => 0    , data => 0
[space bar]
Status at 10:51:14 28-FEB-91, test started at 10:49:56 28-FEB-91
Unit : 0, Bytes => 40388      , soft => 0    , hard => 0    , data => 0
[space bar]
Status at 10:52:12 28-FEB-91, test started at 10:49:56 28-FEB-91
Unit : 0, Bytes => 270560      , soft => 0    , hard => 0    , data => 0
[space bar]
Status at 10:53:10 28-FEB-91, test started at 10:49:56 28-FEB-91
Unit : 0, Bytes => 442468      , soft => 0    , hard => 0    , data => 0
[space bar]
Status at 10:54:10 28-FEB-91, test started at 10:49:56 28-FEB-91
Unit : 0, Bytes => 648084      , soft => 0    , hard => 0    , data => 0
[space bar]
Status at 10:55:20 28-FEB-91, test started at 10:49:56 28-FEB-91
Unit : 0, Bytes => 976868      , soft => 0    , hard => 0    , data => 0
[space bar]
Status at 10:56:14 28-FEB-91, test started at 10:49:56 28-FEB-91
Unit : 0, Bytes => 1253844      , soft => 0    , hard => 0    , data => 0
[CTRL/G] (takes several seconds to exit)
CLI>
```

□ 2. Ethernet Controller Exerciser

Enter **x netx** at the CLI> prompt. At the NETX> prompt, run all tests using the **test_all** command.

The program runs for about 5 minutes, and before it terminates, you will have to restart the machine. There should be no errors reported. Sample output:

```

CLI> x netx
NETX> test_all
    Testing RESHA STANDARD and SHORT ADDRESS registers
        Phase 1 - Short holding register wrap test
        Phase 2 - Standard holding register wrap test
        ADDRESS REGISTERS test ok
    Testing VME CONTROLS
        CONTROL test ok
[Delaying for VME_SYSFAIL assertion]
[Delaying for controller to complete self test diagnostics]
    Testing RAM
        Testing sliding 1
        Testing sliding 0
        Testing all 0
        Testing all 1
        Testing mapping (location := address)
            Initializing ram
            Verifying ram
    Memory tested successfully

CONTROL TESTS assert UNIBUS ACLO which reset the I/O controllers.
You must now crash the IOP with white button or
    BREAK KEY / option 3 / "res"
[press BREAK key]

```

```

Please enter
0 -> Restart system
1 -> Ignore break key
2 -> Redisplay recent console output
3 -> Enter debugger

```

Enter option : 3

```

Trapped into debugger
RD0 00000000 RD1 00002CA2 RD2 0000000C RD3 0000003B
RD4 00000013 RD5 00000002 RD6 00000001 RD7 00000008
RA0 0000FC40 RA1 00022105 RA2 0003FF64 RA3 00036141
RA4 00033AF0 RA5 000371F2 RA6 0003FF58 ISP 0000FAEC
PC DCF7FFFF USP DCF7FFFF ISP DCF7FFFF MSP DCF7FFFF SR 2100
VBR DCF7FFFF ICCR DCF7FFFF ICAR DCF7FFFF XSFC 7 XDPC 7
@res
R1000-400 IOC SELFTEST 1.0.0 ...
    512 KB memory ... [OK]
    Memory parity ... [OK]
    I/O bus control ... [OK]
    I/O bus map ... [OK]
    I/O bus map parity ... [OK]
    I/O bus transactions ... [OK]
    PIT ... [OK]
    Modem DUART channel ... [OK]
    Diagnostic DUART channel ... [OK]
    Clock / Calendar ... Warning: Calendar crystal out of spec! ... [OK]
Checking for RESHA board
    RESHA EEPROM Interface ... [OK]
Downloading RESHA EEPROM 0 - TEST
Downloading RESHA EEPROM 1 - LANCE
Downloading RESHA EEPROM 2 - DISK
Downloading RESHA EEPROM 3 - TAPE
    DIAGNOSTIC MODEM ... [OK]
    RESHA VME sub-tests ... [OK]
    LANCE chip Selftest ... [OK]
    RESHA DISK SCSI sub-tests ... [OK]
    RESHA TAPE SCSI sub-tests ... [OK]
    Local interrupts ... [OK]
    Illegal reference protection ... [OK]
    I/O bus parity ... [OK]
    I/O bus spurious interrupts ... [OK]
    Temperature sensors ... [OK]
    IOC diagnostic processor ... [OK]
    Power margining ... [OK]
    Clock margining ... [OK]
Selftest passed

```


Boot the system and perform the Disk Exerciser procedure described below. This procedure is executed at the EEDB prompt from the operator console:

☐ 1. Disk Exerciser (run for 5 minutes)

a. Elaborate the Disk Exerciser procedure:

```
EEDB: e disk_exerciser
DISK_EXERCISER.x.x.x ...
```

b. Specify number of minutes to execute the test:

```
---->> Disk Exerciser <<----
For how many minutes do you want this test to run? 5
Test will end at : ....

---->> Disk Tester <<----

Test choices are:
TRIVIAL_READS_WRITES
.
.
The chosen test is run concurrently on all specified disks.
.
.
```

c. Execute the test:

```
Do you have a hardcopy terminal attached? [n] n
do you want to specify test options? [n] n

Unit 0 is mapped to volume 1, HDA number ..., size = ( .... )
Unit 1 is mapped to volume 2, HDA number ..., size = ( .... )
Unit 2 is mapped to volume 3, HDA number ..., size = ( .... )
.
.
Running test SEEK_PROFILE
End of pass 1, total I/Os = ...

---->> Disk Exerciser <<----
Disk Test Done
EEDB:
```

d. Unelaborate the Disk Exerciser procedure:

```
EEDB: u disk_exerciser
Subsystem:
Unelaborated DISK_EXERCISER.x.x.x
EEDB:
```

Installing the Environment Terminal (optional)

The operator console can be disconnected from the "Operator Console" port and reconfigured as an Environment terminal using the port marked "Comm Port" (Refer to Figure 1.5).

The Environment terminal can be used to access the Environment during installation, providing a better interface for configuring the 400S for the public network. The Environment terminal also simplifies the configuration of the port to allow dial-out capability for standard support.

Note: It is also possible to configure the system from the Operator Console. However, this interface has limited editing capabilities. Contact the Rational Response Center for further details about this alternate procedure.

To configure the Environment terminal:

- ☐ 1. Disconnect the operator console from the "Operator Console" port.
- ☐ 2. Install terminal cable to the connector marked "Comm Port" (Refer to Figure 1.5).
- ☐ 3. Install the power cords and apply power to the terminal.

Configuring the Rational Environment

Follow the procedures below to configure and check such areas as network addresses, machine names, and product authorization. (see the *System Manager's Guide for the R1000 Development System* for details).

- ☐ 1. Boot the [STANDARD] configuration.
- ☐ 2. Log into the R1000 using the Operator account. This can be done by placing a terminal on the Comm Port or by logging into the console via the Command Interpreter program.
- ☐ 3. Configure the system. From the !Users.Operator world, execute:

`Do_Step ("Setup");`

You will be asked to enter the machine name and an IP address. For example:

Enter Machine Name : **Bluto**
Enter IP Address : **89.44.5.1**

When this step has executed the machine should now be on the network, enabling you to log in via telnet.

- ❑ 4. If you are planning to configure the port marked "Comm Port," for standard support (optional) execute the `!Machine.Initialize_Terminals` command to set up the proper baud rate for the modem you plan to install (see the section "Configuring the Comm Port (optional)").
- ❑ 5. Run `Product_Authorization.Register` for all products to be registered. This will include the session limits. Purchased products and session limits are usually registered before the system is shipped. A list of Authorization Codes for purchased products and session limits should have been mailed to you. Verify that these have been registered, and register any products or session limits which have not already been registered. Call the Rational Response Center to verify or request the appropriate codes.

Verify the products that have been registered by using the `Product_Authorization.Show_Registration` command in the library `!Implementation` (as described above). Use `Show_Tokens` to display the number of `Full_Session` or `Fundamental_Session` tokens (login tokens) and product tokens (RDF, RXI, etc.).

- ❑ 6. Complete installation of products which could not be fully installed at the factory. (See the installation procedures, included with each product shipment, for more details.)
- ❑ 7. Shut the system down and reboot.

Note: Shutting the system down and rebooting by using the `Schedule_Shutdown` command takes a snapshot automatically. If you use the [Break] key make sure to take a snapshot before you take down the system— otherwise the changes above will not be saved.

Overall System Operation

Complete the following steps:

- ❑ 1. Perform system tailoring (for example, creating user accounts, access control groups, and so forth).
- ❑ 2. Check the login process.
- ❑ 3. Execute an overnight run of the `Disk_Exercise` (so it runs through daemon cycle). Run this command from the EEDB.

5

OTHER SUPPORT

Complete the following steps:

- ☐ 1. The Technical Representative should review the operation of the Response Center with the customer.
- ☐ 2. The Technical Representative should provide an overview of system operations with the customer. For example, shutting the system down, booting the system, taking system backups, and handling emergencies.
- ☐ 3. The Technical Representative should review the completed "Rational Installation Checklist" with the customer.
- ☐ 4. The Technical Representative should complete a Support Activity Report (SAR) and distribute copies appropriately. Information provided should include such items as names and phone numbers of the System Manager and phone numbers for diagnostic lines, operations room, etc.

the vertical at the location designated on the map.

A

R1000 SERIES 400S SPECIFICATIONS

Physical Specification	Series 400S
CPU	Rational
Memory Capacity	32 or 64 Mbyte
Memory Transfer Bandwidth	80Mbyte/second
Logical Address Space	67 bits
Data Paths	Dual 64-bit ALUs
Disk Capacity	up to 4x1.2 Gbyte
Tape Drive, standard	8mm Cartridge
Height	38in(97cm)
Width	16in(40.6cm)
Depth	28in(71.1cm)
Weight	220 lbs. + 10 lbs./disk unit
Operating Temp	60 to 85 degrees F (15 to 29 degrees C)
Operating Humidity	20 to 80% non-condensing
Cooling	9,000 BTU's per hour (2000 watts)
Voltage	208 to 240 VAC +5%
Frequency	60 Hz (dom), 50 Hz (int)
Power Connection	3 wire
Amperage	12A
Max Service	30A
AC Power Cord	12awg 3 wire
Length	15 feet (4.57m)
Plug	NEMA L6-30P (domestic)
Receptacle	NEMA L6-30R (domestic)
Transceivers (Ethernet)	See Rational price list
Cables (tranceiver)	See Rational price list
Operator Console Connector	RS-232
Ethernet Connector	Ethernet
Telephone Jack	RJ45

B

RATIONAL INSTALLATION CHECKLIST

A Rational Technical Representative should check-off each item and show completed list to the customer.

Installation Address:

Customer Telephone _____

Customer Contact _____

Date (system arrival) _____

Date (start install) _____

R1000 ID Number: _____

Installation Task:

☐ 1. Unpacking and Inspection

- a. _____ Remove all protective-packing material.
- b. _____ Check the shipment against the packaging list.
- c. _____ Check all boxes and system components for external damages.
- d. _____ Record any discrepancies observed while checking the shipment and report them to the Response Center (document in the report any damaged items).
- e. _____ Document in the installation report any observations relating to damaged items, and notify the Response Center.

☐ 2. Final Installation Set-up Procedures

- a. _____ Position the system in the computer area according to the layout plan.
- b. _____ Make complete visual checks of cables, harnesses, backplane, foreplane, fans, filter pads and all cable connectors.
- c. _____ Make sure minimum tools required for installation are at the site.

- d. _____ Locate the cabinet stabilizing brackets.
- e. _____ Route cables externally to system's cabinet (if there is no raised floor).
- f. _____ Make sure disk drives are installed in the slots specified on the back of the disk housing cover.

☐ 3. Hardware Installation, Testing and Configuration

- a. _____ Install the cabinet stabilizing brackets over each wheel.
- b. _____ Check for proper routing and connection of cables.
- c. _____ Verify circuit breakers CB1 and CB2 are in the OFF position on the PDU.
- d. _____ Verify the power keyswitch is in the STANDBY position, disk write keyswitch is in PROTECTED and operator mode keyswitch is in AUTO position at the main system control panel.
- e. _____ Verify the RESHA board is seated properly.
- f. _____ Verify the R1000 processor and memory boards are seated properly.
- g. _____ Verify the disks are seated properly and match the configuration listed on the back of the disk housing cover.
- h. _____ Connect the AC power cord plug to the mating connector on the PDU.

☐ 4. Power Checkout.

- a. _____ Ensure the circuit breakers (located on the PDU) are in the "OFF" position and that the power keyswitch (on the system control panel) is in the "STANDBY" position.
- b. _____ Verify the customer's power source circuit breakers are ON.
- c. _____ Measure the AC power at the receptacle (using a voltmeter between the two hot leads). The voltage should measure between a minimum value of 208 VAC ($\pm 5\%$) to a maximum value of 240 VAC ($\pm 5\%$).

- d. _____ Connect the power cord to the mating connector originating from the customer's power source.
- e. _____ Place circuit breaker CB2 to the "ON" position.
- f. _____ Using a voltmeter, measure the voltage at the power jack labeled "Service Power Monitor" (located at the lower right-hand edge of the PDU). The voltage should measure between a minimum value of 208 VAC ($\pm 5\%$) to a maximum value of 240 VAC ($\pm 5\%$).
- g. _____ Place circuit breaker CB1 to the "ON" position and CB2 to the "OFF" position.
- h. _____ Switch the POWER keyswitch on the System Control Panel to the halfway (Power On) position.
- i. _____ Turn the Operator Mode Key switch to "INTERACT."
- j. _____ Check to see if any disk unit's FAULT indicator is lit on the status control panel.
- k. _____ Switch the Disk Write key switch to "ENABLED."
- l. _____ Measure DC voltage at the CPU backplane for +5 VDC ($\pm 5\%$), +12 VDC ($\pm 5\%$), and -12 VDC ($\pm 5\%$).
- m. _____ Verify all air filters are in place and clean.
- n. _____ Verify all fans are blowing and are operational.

☐ 5. Installing the Operator Console

- a. _____ Place the operator console in the location designated by the installation floor plan.
- b. _____ Verify that the operator console cable is installed in the connector marked "Operator Console" on the RESHA board face plate.
- c. _____ Set up the operator console's operating parameters.

☐ 6. System Checkout Procedures—Execute phase two FRU diagnostics for each board type:

- a. _____ IOC
- b. _____ VAL
- c. _____ TYP
- d. _____ SEQ
- e. _____ FIU
- f. _____ MEM 0 and 2
- g. _____ Run confidence test (option 2 on the FRU main menu).

☐ 7. Configuring the System for Remote Diagnostics (optional)

- a. _____ Contact the Rational Response Center and inform them that you plan to test remote diagnostics.
- b. _____ Run INITIOA from the CLI> prompt.
- c. _____ Verify the cluster ID and modify it if necessary.
- d. _____ Verify the type to be used for remote diagnostics and correct it if necessary.
- e. _____ Verify the data needed for the remote diagnostics connection and correct it if necessary.

☐ 8. System Peripheral Exercisers:

- a. _____ Tape drive
- b. _____ Ethernet controller
- c. _____ Disk drives

☐ 9. Installing the Environment Terminal (optional).

- a. _____ Place the terminal at the location designated on the installation floor plan.
- b. _____ Verify that the terminal cable is installed in the connector marked "Comm Port."

c. _____ Install the power cords and apply power to the terminal.

☐ 10. Configuring the Rational Environment

a. _____ Boot the standard configuration.

b. _____ Configure the System.

c. _____ Run Product_Authorization.Register to register any additional products or sessions.

d. _____ Shut down the system and reboot.

e. _____ Load any additional software that the customer has purchased, performing additional system configuration as needed.

☐ 11. Overall System Operation:

a. _____ Perform system tailoring (e.g. user accounts, access lists, etc.).

b. _____ Check Login process.

c. _____ Execute overnight run of the Disk_Exerciser (at EEDB).

☐ 12. Other Support:

a. _____ Customer Support Response Center: Review operation of the Customer Support Response Center with the customer's system manager.

b. _____ System operations: Provide overview of system operations.

c. _____ Installation Checklist: Review Completed Installation Checklist with System Manager.

d. _____ Support Activity Report (SAR): Complete SAR and distribute copies appropriately.

C

UPGRADE 1.2 GB DISK DRIVE

This appendix presents instructions for installing an additional 1.2 GB drive in Series 400 processor.

Preparation

- 1. Saving system data (This is performed by the customer) .
The customer must take a backup of the system before you start this installation. This should be done with no users on the system. (Note: Do not proceed until the backup has been taken.)

- a. Have all users log off the system.
- b. Log in to the system as a user who is a member of group **Privileged**.
- c. Disable all terminal ports for login to ensure that no one logs into the system.

```
Op.Limit_Login (1);
```

- d. Mount a tape to be use for an environment backup.

- e. Take a Full backup of the system.

```
Full_backup;
```

- f. Verify that the tapes are readable.

```
Verify_Backup;
```

Installation

- 1. Shutting down the system
 - a. Enter the privileged Kernel (*executing this commands will trash all the environment*)

```
Kernel: enable_priv_cmds  
Proceed [false]: true  
Password: secret
```

b. Issue the **Go_Back_In_Time** command from the privileged kernel

```
*Kernel: go_back_in_time
The purpose of this command is to trash the current state
of this machine. When the system is next booted, it will
require that you build a new virtual memory system and
recover from backup tape.
Proceed [FALSE]: true
SNAPSHOT_NUMBER [0]: <cr>
```

☐ 2. Take a DFS backup from the CLI (you should not need to use this backup; it is merely taken as a precaution):

a. After you had issued the **go_back_in_time** command, the system shutdown, and will be asked if you want to boot the [STANDARD] configuration. Answer this question by pressing "CONTROL and G" keys at the same time. This will take you to the CLI> prompt.

b. Mount the tape that is to be used for the DFS backup.

c. Perform the backup

```
CLI> backup/v
```

☐ 3. Powering Off the System

a. On the System Control Panel turn **DISK WRITE** keyswitch to **PROTECTED** position.

b. Turn **OPERATOR MODE** keyswitch to the **AUTO** position.

c. Turn **POWER** keyswitch to the **STANDBY** position.

☐ 4. Installing the Disk Drives

a. Remove the disk housing cover by turning the fastener screw counter clockwise.

b. Slide the new 1.2 GB drive in the next available slot in the disk housing, make sure that the disk drive is seated properly. Do not move the other drive(s). Unlike previous R1000 systems, the disk drives in the series 400 are **SLOT-DEPENDENT**, i.e., the drive for unit 0 must be in the top slot of the disk housing, unit 1 must be in the 2nd slot from the top, etc.

☐ 5. Powering On the System

a. Switch the **POWER** keyswitch on the System Control Panel to the "ON" position—this turns on the AC power. All fans should start blowing. This also allows the DC power supplies to apply DC voltage to the CPU,

controller boards and disk drives.

- b. Unprotect the disk drives by switching the DISK WRITE keyswitch on System Control Panel to the "ENABLED" position.
- c. Boot the STANDARD configuration. When the kernel finds out that there is no Environment, the Environment Recovery program is invoked. The environment backup tape that you had taken earlier should be mounted on the 8mm tape drive to restore the data back in the disks. The following is an example of the output from the environment recovery program.

```
Enter name of configuration to boot [STANDARD] :

.....

---->> Kernel.11.5.0 <-----
Kernel: CHANGE_GHOST_LOGGING
WANT TRACING: FALSE
WANT LOGGING: FALSE
Kernel: START_VIRTUAL_MEMORY
ALLOW PAGE FAULTS: YES

---->> ERROR_LOG <-----
13:03:02 --- TCP_IP_Driver.Worker.Finalized

---->> CONFIGURATOR <-----
starting diagnosis of configuration

**** VOLUME CONFIGURATION ****

UNIT          VOLUME    VOL    DATE OF ROOT
              NAME      NUM    WHEN ADMITTED
-----
0 volume 1          1    06-AUG-91 12:45:56
1

**** PROBLEMS WITH CURRENT VOLUME CONFIGURATION ****

unit 1 state = FORMATTED
missing volume 2
WANT TO BUILD NEW SYSTEM (YES/NO): yes
starting creation of new system
VOLUME NAME FOR unit 0: volume 1
Setting free blocks to gc footprint takes about 15 minutes per drive.
Want to set free blocks to gc footprint? no
VOLUME NAME FOR unit 1: volume 2
creating root volume: 1
adding volume 2 to virtual memory system
creation of new system is complete
starting virtual memory system
the virtual memory system is up

---->> Kernel.11.5.0 <-----
Kernel: START_NETWORK_IO
Kernel: START_ENVIRONMENT
TRACE LEVEL: INFORMATIVE
Kernel:
---->> Environment Elaborator <-----
Elaborating subsystem: ENVIRONMENT_DEBUGGER
Elaborating subsystem: ABSTRACT_TYPES
Elaborating subsystem: MISCELLANEOUS
Elaborating subsystem: OS_UTILITIES

---->> Recovery <-----
Recovery Is Needed, Should I fake it? no
```

Please tell me Volume Id for the Backup Index Tape: spacebar <cr>

====> SYSTEM & RECOVERY <====

Please Load Tape

(Kind of Tape -> CHAINED_ANSI,
Direction -> READING)
Is Tape mounted and ready to read labels? yes
Info on tape mounted on drive 0 is:
(Kind of Tape -> CHAINED_ANSI,
Writeable -> FALSE,
Volume Id -> D22OS2,
Volume Set Name -> BACKUP, 05-AUG-91 12:52:05 3)
OK to read volume? [YES] <cr>

====> Kernel.11.5.0 <====

Kernel:

====> Recovery <====

Positioning tape to Backup Index
Processing Backup Index
Processing tape to Backup Data
Processing Backup Data

====> Kernel.11.5.0 <====

Starting snapshot
Snapshot is finished
Starting snapshot
Snapshot is finished

====> Recovery <====

Recovery Is Complete
Garbage Collection can't run until the machine is rebooted
Reboot before elaborating the Environment

After recovering from the backup tapes, the system will have to be shutdown in order to enable disk collection to run. Reboot the system. (Note: In the future, when D2.2 is release the system will automatically shutdown when the environment recovery is finished.)

- ☐ 6. Fill out the SAR and return it to Rational.

Training

Installation Procedure

Release1_1_0

RATIONAL

Copyright © 1991 by Rational

PN 507-003265-002

This document is subject to change without notice.

Rational and R1000 are registered trademarks and Rational Environment is a trademark of Rational.

Rational, 3320 Scott Boulevard, Santa Clara, California 95054-3197

Table Of Contents

1 Installation	1
Overview	1
VERIFYING PREREQUISITES	1
INSTALLING THE RELEASE	2
Appendix A Procedure Do_Step	5
Appendix B Feedback	7

1

Installation

Overview

When installing the Training release, you can view the process as a series of phases. These phases must be performed in a serial manner.

- Verifying prerequisites to the installation
- Installing the release

These phases must be performed in a serial manner, as must the steps in each of the phases. This installation note is organized by phase, with each phase comprising a section of this document. Once you have performed this installation, please fill out and return the feedback form that is the last page of these instructions along with a printout of the `Do_Step_Execution_Time` found in the `Logs` library of the release archive. Your feedback is very important in helping us to improve our installation instructions.

VERIFYING PREREQUISITES

In this phase of the installation, you will verify that the prerequisites for installing this upgrade have been met. Do not proceed with the upgrade until all prerequisites have been fulfilled.

- ☐ 1. You have read the release note.
- ☐ 2. You have read this entire install note.
- ☐ 3. The machine is executing Environment release D_12_1_1 or later.
- ☐ 4. The machine is running the necessary product(s) to support the functioning of the desired training course.
- ☐ 5. If desired, request the `Training_Authorization` code for use in inhibiting product token checking, allowing the maximum number of students to be trained without restriction to the number of tokens purchased by the customer. This must be done in advance of the training date.

Note that this installation allows selective restoration of Rational training courses.

INSTALLING THE RELEASE

In this phase of the release, you will restore the tools tape which contains a release world located in `!Machine.Release.Archive.Training.Release1_1_0`. This release world contains nested archives which will then be restored to complete the installation.

- ☐ 1. Load the Training Release1_1_0 tools tape onto the tape drive.
- ☐ 2. Log in as user `Rational`, which must be a member of group *Privileged*.
- ☐ 3. **LOAD_TAPE** *[5 minutes]*
This step restores the Training Release1_1_0 tape. This step can be executed from any command window. There should be no errors.
- ☐ 4. Go to `!Machine.Release.Archive.Training.Release1_1_0`.

The following steps utilize the procedure `Do_Step` as described previously.

- ☐ 5. **RESTORE_NOTES** *[1 Minute]*
This step restores the release note `Training_Release1_0_0_LPT` to `!Machine.Release.Release_Notes`.
- ☐ 6. **RECORD_INSTALLATION** This step records release information in `!Machine.Release.Current.Products`. This is a required step and establishes that the release has successfully been installed.
- ☐ 7. Only execute the step name(s) corresponding to the course(s) you wish to install on the machine (see release note). It is assumed that the product(s) required to support installation of a course are resident and operational on the machine.
Valid names are:

ENV_FUNDAMENTALS
ENV_ADVANCED_TOPICS
ENV_TOOLSMITHING
ENVIRONMENT
NETWORKING
PROJECT_DEVELOPMENT_METHODS
CMVC
SYSTEM_MANAGEMENT
MC68020_BARE
MC68020_OS2000
RCF_FUNDAMENTALS
RDF_2167A_FUNDAMENTALS
RDF_2167A_TOOLSMITHING
RDF_2167_FUNDAMENTALS

RDF_2167_TOOLSMITHING

Once a course has been restored, follow the instructions in the release note for creating the appropriate number of training user accounts.

A

Procedure Do_Step

The majority of the steps which must be performed for this installation utilize the procedure `Do_Step` (included as part of the release tape and located in `!Commands`) to execute the required sequence of commands which implement the step. This procedure must always be executed from a command window in the release library

`!Machine.Release.Archive.Training.Release1_1_0`

Each step which utilizes procedure `Do_Step` will be of the form

<code><STEP_NAME></code>	<code>[<TIME TO EXECUTE>]</code>
	<code><DESCRIPTION></code>

where `<STEP_NAME>` is passed as the parameter to procedure `Do_Step` which performs the necessary actions to complete the step, `<TIME TO EXECUTE>` is the amount of time the step takes to execute (this may be expressed as a range, in which case the lower values is typically the minimum, and the upper value is an estimate of the maximum), and `<DESCRIPTION>` is a description of what the step does, including any action which you will need to manually perform. For example:

FOO

[5 Minutes]

This step is an example of the format used for steps which are executed using procedure `Do_Step`. To execute step **FOO**, you would go to `!Machine.Release.Archive.Training.Release1_1_0` and in a command window, execute:

```
Do_Step ("FOO");
```

which would result in automatic execution of all commands required to implement this step. If you needed to take any manual action in addition to executing `Do_Step`, it would be noted as :

☐ Perform some manual check

If any errors occur for this step, you should always fix the problem before proceeding on to the next step. Each step is defined by a fragment of Ada code which is executed by `Program.Run`. These fragments are stored in the `Steps` file

located in the `Command_Data` library of the release. In the event you want to modify a step, you can use the "PROMPT => <STEP>" form when invoking `Do_Step`. See the spec of procedure `Do_Step` (located in `!Commands`) for more detailed information.

Warning: If you interrupt the execution of `Do_Step` (by using `Job.Interrupt` such as `CONTROL-G`) it is possible that certain interactive commands executed by some steps, such as `Common.Definition`, may fail with an exception and display a message such as

Unable to read file due to Constraint_Error (Null Access)

In general, this message has no negative impact on the execution or completion of the step, and can be ignored.

B

Feedback

In order to help us improve our instructions and make installations easier to do, the following form is provided to allow you to give us feedback. Please complete and send along with a printout of the file `Do_Step_Execution_Time` located in `!Machine.Release.Archive.Training.Release1_1_0.Logs` to:

Rational
Atten: SMSE
3320 Scott Blvd.
Santa Clara, CA 95054-3197
Re: Training

- ☐ 1. How long did the installation take you?
- ☐ 2. Did the installation proceed as described in the instructions?
- ☐ 3. What could be done to improve these (or other) instructions/installations?
- ☐ 4. What did you like about this set of instructions/installation?

SMSE Checklist MEMORY_32_MB

☒ FIELD INSTALLATION REQUIREMENTS

<u>PART NUMBER</u>	<u>DESCRIPTION</u>
<input type="checkbox"/> 504-003207-007	SMSE Checklist for Environment Release 12_1_1 <input checked="" type="checkbox"/> Not Required with D_10_20_2 or later
<input type="checkbox"/> 507-003237-004	MEMORY_32_MB Installation Procedure
<input type="checkbox"/> 547-003230-001	2 Gold Wipes <input checked="" type="checkbox"/> Not Required
<input type="checkbox"/>	2 backplane jumpers <input checked="" type="checkbox"/> Not Required
<input checked="" type="checkbox"/> 505-003249-001	Support Activity Report (SAR)

X_Interface

Installation Procedure

Release10_10_1

RATIONAL

Copyright © 1992 by Rational

PN 507-003219-012

This document is subject to change without notice.

Rational and R1000 are registered trademarks and Rational Environment is a trademark of Rational.

Rational, 3320 Scott Boulevard, Santa Clara, California 95054-3197

Table Of Contents

1	Installation	1
	Overview	1
	Assesing the Impact of the Release	1
	Installation Checklist	2
	Installing the release on an R1000	3
	Install Keymap Files	3
	Installation Testing	4
	Installation Clean Up	5
	Installation using Vax VMS and DECwindows	6
	New or Changed System Files	7
	Installation using DEC Ultrix with DECwindows	9
	Installation using HP-UX and the X Window System	12
	Installation using AIX and IBM X-Windows	15
	Installation using A/UX and the X Window System	19
	Installation using SunOS with X11/NeWS	22
	Installation using SunOS with MIT X11.R5	27
2	Workstation Installation Overview	31
	R1000 Installation Overview	31
	Workstation/Fileserver Installation Overview	32
	Workstation/Fileserver Components	33
	RXI Executable	33
	RXI Default Resources File	34
	RXI Fonts	35
	RXI Man/Help File	36
	Workstation/Fileserver Installation Directories	36
	BINDIR Directory	36
	XBINDIR Directory	37
	FONTDIR Directory	38
	XAPPLOADDIR Directory	38
	MANDIR Directory	39
	OPENWIN Directory	39
3	Common Installation Questions and Problems	41
	Not using default location for RXI objects	41
	Hetrogenous networks	41
	Using links for default resources files	41
	Key Foobar doesn't work	41
	Missing resources files	42
	Questions and Answers	42
	Appendix A Procedure Do_Step	49
	Appendix B INSTALL_X_INTERFACE	51
	Appendix C Supported Configurations	53
	Appendix D Feedback	55

1

Installation

Overview

Estimate time for installation: 3 hours.

When installing the X_Interface release, you can view the upgrade process as a series of phases. These phases must be performed in a serial manner in the order specified.

- Assessing the impact of the release on the user community
- Verifying prerequisites to the installation
- Installing the release on the R1000
- Installing the release on the Workstation
- Cleaning up

On the next page, there is a checklist for installing X_Interface. Review this checklist and follow the instructions in it.

Warning: *Be sure to complete ALL of the steps in the installation exactly as written and follow the installation steps in each phase in a serial manner. Failure to adhere to these instructions will often result in the failure of the product.*

Assesing the Impact of the Release

In this phase of the installation, you will assess the impact of the release on the user community and the machine.

- ☐ 1. The installation of X_Interface may slow down the R1000 considerably, specially when running `refresh_terminal_information`.
- ☐ 2. `X?_Commands` in `!Machine.Editor_Data` will be over written.

Installation Checklist

✓	Do	Step
<input type="checkbox"/>	Read	"Do_Step Information", Appendix A, page 49, if not familiar with the Do_Step command. This command is used throughout the installation process.
<input type="checkbox"/>	Read	"Workstation Installation Overview", Chapter 2, page 31 for an overview of the Workstation RXI installation process. Common installation problems and issues are covered in this section. IF YOU HAVE NOT INSTALLED RXI BEFORE, READ THIS SECTION!
<input type="checkbox"/>	Read	"Supported Configurations", Appendix C, page 53 and pick the appropriate RXI Terminal-Type(s) to be installed. There may be more than one.
<input type="checkbox"/>	Record	Your choice of RXI Terminal-Type(s) here (from the "Supported Configurations" Appendix) _____. (There may be more than one.)
<input type="checkbox"/>	Record	The workstation-specific installation chapter here (from the "Supported Configurations" Appendix) _____. (There may be more than one.)
<input type="checkbox"/>	Read	"Supported Configurations", Appendix C, page 53 to verify that installation prerequisites for are met. The installation, or the installed product, will fail if the prerequisites aren't met.
<input type="checkbox"/>	Install	The keymaps for your chosen RXI Terminal-Type(s) on the R1000 using "Installation on R1000", Chapter 1, page 3.
<input type="checkbox"/>	Install	RXI on the workstation(s) and/or fileserver(s) using the proper workstation-specific installation section(s) recorded above.
<input type="checkbox"/>	Test	The RXI Terminal-Type(s) to verify installation was successful. See "Installation Testing", Chapter 1, page 4.
<input type="checkbox"/>	Fill	Fill out the "Feedback" form at the end of this install note.
<input type="checkbox"/>	Clean	RXI installation areas on the R1000 and workstation. See "Installation Clean Up", Chapter 1, page 5.
<input type="checkbox"/>	Done	Installation complete.

Installing the release on an R1000

In this phase of the install, you will restore the tape which contains a release world located in `!Machine.Release.Archive.X_Interface.Release10_10_1`. This release world contains nested archives which will then be restored to complete the installation.

The following steps utilize the procedure `Do_Step` as described in Appendix `Do_Step`.

- ✓ 1. Load the `X_Interface.Release10_10_1` tape into the R1000's tape drive.
- ✓ 2. Log in to the Environment as user `Rational`, which must be a member of group `Privileged`.
- ✓ 3. **LOAD_TAPE** [10 Minutes]
This step restores the `X_Interface.Release10_10_1` tape. This step can be executed from any command window. Answer the mount request at the Operators Console. There should be no errors.

Note: On machines that do not have `Do_Step`, use `Archive.Restore` (Options => "Promote, Replace"); to restore the tape.

- ✓ 4. Traverse to `!Machine.Release.Archive.X_Interface.Release10_10_1`.
- ✓ 5. **INSTALL_X_INTERFACE** [35 minutes]
This step will restore all the archives, and check for appropriate authorization codes. The section `INSTALL_X_INTERFACE` contains more information about the steps executed by this step and possible errors. The steps may be run individually if desired.

Install Keymap Files

Traverse to `!Machine.Release.X_Interface.Release10_10_1`.

Install the keymap files and update the `Terminal_Type` and `Terminal_Recognition` files on the R1000. This step must be repeated for each RXI terminal type you have selected for use with this R1000. For example, to install the files for a DEC workstation under the VMS operating system with the `UltrixConnection`, enter the following in a command window:

```
Rxi_Install.Machine_Editor_Data_Files
(Keyboard => Rxi_Install.Xdecus_Vms5_4_Ultrixconnection13a,
 Suppress_Rxi_files => False);
```

and press the [PROMOTE] key. If you are installing more than one terminal type on this R1000, you can save time by setting the `Suppress_Rxi_Files` parameter to `True` after this command has been executed once.

Enable the new keymaps. The new RXI keymaps must be enabled before they can be used. Create a Command Window and type:

```
Refresh_Terminal_Information
```

and press the [PROMOTE] key to enable the keymaps.

Installation Testing

The RXI product is keyboard and X server specific. This means that it must be uniquely and specially configured for each and every keyboard and X server that will be used. In order for RXI to work correctly, a user must log into an R1000 using an RXI terminal type name that correctly, and uniquely, specifies the keyboard-layout, the X server, and usually the keyboard-vendor and X software-vendor that is being used.

For example, a Sun Microsystems(tm) workstation using the SunOS(tm) operating system can have a Sun type-4 keyboard. If a user will be using the MIT X Window System then his terminal type is `xsun4`. However, if a user will be using the Sun OpenLook (a.k.a. OpenWindows or NeWS) then his terminal type is `xNews4`. The keyboard is the same keyboard in either case; however, the software being used will cause the keyboard to appear different to RXI and thus to the R1000.

This means that, in order to properly test an RXI installation, it is necessary to log into an R1000 from each and every keyboard type that will be used. Most sites will only use one or two keyboard types so this is not usually difficult.

After the installation is complete, log-on to the R1000 from each variety of workstation or X-based terminal keyboard. If more than one vendor's X Window System software will be used, make sure to log in once from each keyboard while using each of the various systems. Try some function keys and menus to verify that the keyboard, X Window System software, and R1000 keymap combination is working correctly.

Installation Clean Up

After the installation is complete, you may delete some objects to save some disk space.

On The R1000:

- `!Machine.Release.Archive.X_Interface.Release10_10_1`. After these archives have been restored they are of no use and may be deleted.
- `!Machine.Release.X_Interface.Release10_10_1`. If you are never going to install additional terminal types this may be deleted. Do not delete this if you think additional workstations or terminal types will be required.

On The workstation(s) and/or fileserver(s):

- Everything in the RXI source directory, the directory specified in `Rxi_Install.Workstation_Files`, may be deleted after the installation of RXI is complete.

Installation using Vax VMS and DECwindows

The steps involved in getting the rxi program running on a DEC VAXstation workstation or fileserver are:

- ☐ 1. Create a temporary source directory for the rxi program on the workstation/file- server. For example, User: [Rxi].
- ☐ 2. Traverse to !Machine.Release.X_Interface.Release10_10_1.
- ☐ 3. Run the Rxi_Install.Workstation_Files procedure; this gets the files copied to the workstation or fileserver from the R1000. Specify the main directory (eg. User: [Rxi]) as the destination of the transfer. For example:

```
Rxi_Install.Workstation_Files
  (Keyboard =>
    Rxi_Install.Xdecus_Vms5_4_Ultrixconnection13a,
    On_Machine => "My_Machine",
    Username => "MyName",
    Password => "MyPassword",
    Account => "",
    Rxi_Source_Directory => "User:[Rxi]");
```

- ☐ 4. Do a SET DEFAULT to the source directory on the workstation and execute the command @makefile restore. This unpacks rxi and makes it available for complete installation.
- ☐ 5. If this site has some or all of the optional licensed Rational products then edit the RXI_Env_Menu file. It's format is that of a C source file so it should look familiar. There is a #define line for each licensed product. For each licensed product at this site, set the corresponding #define to be equal to 1. This activates the rxi Environment menu entries for that product.
- ☐ 6. Edit the MAKEFILE.COM file in the source area. Follow the editing instructions at the front of the file. You will be asked to specify the TCP/IP library to use (Excelan MultiNet, Wollongong WIN TCP/IP or Ultrix Connection) and to specify the directories containing the library include files and the library .OLB file. This is also the place where you indicate the proper installation directories and protections for the various pieces of the RXI system.
- ☐ 7. You now have three more choices. A) Install the Rational-supplied rxi as-is, or, B) relink to use local (possibly shared) libraries, or C) completely recompile and recreate rxi using all local definitions. The supplied version of rxi should be able to execute on your system. However, it will not use any shared libraries, nor will it incorporate any local library fixes, extensions, or modifications that may exist. NOTE: The optional C compiler is required to relink or recompile rxi.

a. If you wish to just use rxi as supplied then skip this step.

- b. If you wish to simply relink rxi so that it uses local libraries then execute
`@makefile relink` command and proceed to the next step.
 - c. If you wish to recreate rxi from scratch then execute `@makefile clean` command followed by `@makefile all`.
- ☐ 8. Enable your privileges; `BYPASS` should be sufficient. Execute `@makefile install.all`; this will:
- Installs the rxi program, its helper program `rxi_detached`, and the RXI default resource file. (The command `@makefile install` performs just this step.)
 - Install the rxi help library. (The command `@makefile install.help` performs just this step.)
 - Compiles and installs the various fonts `fixed-screen-*.bdf` in the `FONTDIR` directory (defaults to `SYS$COMMON:[SYSFONT.DECW.USER_75DPI]`). These are the normal and bold-face fonts used by rxi. (The command `@makefile install.fonts` performs just this step.)
- ☐ 9. Edit the `SYS$MANAGER:SYLOGIN.COM` file and add a definition for a system-wide symbol named `RXI`. eg.
- ```
$ RXI := $ SYS$COMMON:[SYSEXEC]RXI_DETACHED.EXE
```
- This creates a system-wide foreign command that will run rxi as a detached process. You will have to logout and log back in to get it defined for yourself.
- ☐ 10. Reboot any and all DECwindows workstations (or just the servers) that need to use rxi. The reboot is necessary in order to cause the servers to recognize the new rxi fonts.
- ☐ 11. You may wish to do a `PURGE` on: `SYS$SYSTEM:`, `DECW$SYSTEM_DEFAULTS:`, and `SYS$COMMON:[SYSFONT...]`.

---

## New or Changed System Files

---

All files are installed with `OWNER=SYSTEM` and a protection more or less of `/PROTECTION=(SYSTEM:RWED,OWNER:RWED,GROUP:RE,WORLD:RE)` as appropriate. See the `Makefile.Com` file for complete details on what protection is given to what class of file.

Symbol `FONTDIR` (usually `SYS$COMMON:[SYSFONT.DECW.USER_75DPI]`)

new fixed-screen-r-11.decw\$font (various fonts)  
new fixed-screen-b-11.decw\$font  
new fixed-screen-r-12.decw\$font  
new fixed-screen-b-12.decw\$font  
new fixed-screen-r-13.decw\$font  
new fixed-screen-b-13.decw\$font  
new fixed-screen-r-14.decw\$font  
new fixed-screen-b-14.decw\$font  
new fixed-screen-wr-15.decw\$font  
new fixed-screen-wb-15.decw\$font  
new fixed-screen-wr-16.decw\$font  
new fixed-screen-wb-16.decw\$font  
new fixed-screen-wr-17.decw\$font  
new fixed-screen-wb-17.decw\$font  
new fixed-screen-wr-22.decw\$font  
new fixed-screen-wb-22.decw\$font

changed none

Symbol BINDIR (usually SYS\$COMMON:[SYSEXE])

new rxi.exe (main executable)  
new rxi\_detached.exe (utility program)

changed none

Symbol HELPLIB (usually SYS\$HELP:HELPLIB.HLB)

new none

changed HELPLIB.HLB (add RXI help data)

Symbol XAPPLOADDIR (usually SYS\$COMMON:[DECW\$DEFAULTS.SYSTEM])

new rxi.dat (system default file)  
new rxi\_env\_menu. (system menu file)

changed none

SYS\$MANAGER:SYLOGIN.COM is also changed, by whoever is performing the installation. A command of the general form 'RXI := BINDIR:RXI\_DETACHED.EXE' is added in order to make RXI generally available to users.

The system must be rebooted to make all changes take complete effect.

---

## Installation using DEC Ultrix with DECwindows

---

The steps involved in getting the rxi program running on a DEC workstation or fileserver running Ultrix are:

- ☐ 1. Create a temporary source directory for the rxi program on the workstation or fileserver. For example, if the X sources are located in a directory called `/src/x` then the suggested place for rxi would be `/src/x/rxi`. If you would prefer to put the sources into a different place, eg. `/vendor/src/rxi`, that works as well.
- ☐ 2. Traverse to `!Machine.Release.X_Interface.Release10_10_1` on the R1000.
- ☐ 3. Run the `Rxi_Install.Workstation_Files` procedure; this gets the files copied to the workstation or fileserver from the R1000. For example:

```
Rxi_Install.Workstation_Files
(Keyboard => Rxi_Install.Xultus_Vs_Ultrix4_1,
On_Machine => "nova",
Username => "sagan",
Password => "verbose",
Account => "unlimited",
Rxi_Source_Directory => "/src/X/rxi");
```

- ☐ 4. Connect to the source directory on the workstation and execute the command `make restore`. This unpacks rxi and makes it available for complete installation.
- ☐ 5. If this site has some or all of the optional licensed Rational products then edit the `RXI_Env_Menu` file. It's format is that of a C source file so it should look familiar. There is a `#define` line for each licensed product. For each licensed product at this site, set the corresponding `#define` to be equal to 1. This activates the rxi Environment menu entries for that product.
- ☐ 6. At the top of the file called `Makefile` is a list of the source, library, include, and font paths that will be used during installation. If you wish to change any of these default locations please do so now. On a DECstation, the defaults are:

```
BINDIR - where rxi executables will reside;
 usually /usr/bin/X11
XBINDIR - where X Window System executables reside;
 usually /usr/bin/X11
FONTDIR - where the server fonts reside;
 usually /usr/lib/X11/fonts/apps/75dpi
MANDIR - the man page directory for the rxi man page;
 usually /usr/man/mann
XAPPLLOADDIR - where application defaults files reside;
 usually /usr/lib/X11/app-defaults
```

On a VAXstation, the defaults are:

**BINDIR** - where rxi executables will reside;  
usually `/usr/bin`  
**XBINDIR** - where X Window System executables reside;  
usually `/usr/bin`  
**FONTDIR** - where the server fonts reside;  
usually `/usr/lib/dwf/75dpi`  
**MANDIR** - the man page directory for the rxi man page;  
usually `/usr/man/man`  
**XAPPLOADDIR** - where application defaults files reside;  
usually `/usr/lib/X11/app-defaults`

If you don't know what your system's standard **XAPPLOADDIR** is, or, if you don't want to put RXI's files there, have each user define one of the following environment variables in a manner similar to what is shown here. These examples assume that the RXI application defaults files have been placed into a directory named `/vendor/RXI/app-defaults`.

```
csh> setenv XFILESEARCHPATH /usr/lib/X11/%T/%N%S:/vendor/RXI/%T/%N%S
csh> setenv XAPPLRESDIR /usr/lib/X11:/vendor/RXI/app-defaults
```

These two environment variables accomplish basically the same task.

If you don't know what directory to use for **FONTDIR**, run the `xdumpfp` program in the RXI source area. It will give you a list of all of the font directories being used by your X server.

If you want to place the RXI fonts into a special directory (eg. so that they are not in one of the normal server directories), you will have to notify the server of their location every time that you log in. This is easily done using the `xset` program. For example, if you put the RXI fonts into `/vendor/RXI/fonts` then you would need this command in your `.xinitrc` startup file (`.xinitrc` is one of the many names for startup files, see your System Administrator or read your system documentation to determine the proper file name for you to use):

```
xset fp+ /vendor/RXI/fonts
```

This command tells the server that additional fonts can be found in this directory. If you cannot find the `xset` program, the RXI source directory contains two programs that can be used to perform this function. They are not normally installed so you will have to copy them, by hand, to an appropriate bin directory. The command that you would use to run these programs is:

```
xsetfp `xdumpfp` /vendor/RXI/fonts/
```

(Please notice the extra `'/'` on the end of the path name.) This command sets the server's font path to what it currently is, plus the new RXI font area.

- ❑ 7. You now have three choices. A) Install the Rational-supplied `rx` as-is, or, B) relink to use local (possibly shared) libraries, or C) completely recompile and recreate `rx` using all local definitions. The supplied version of `rx` should be able to execute on your system. However, it will not use any shared libraries, nor will it incorporate any local library fixes, extensions, or modifications that may exist.
  - a. If you wish to just use `rx` as supplied then skip this step.
  - b. If you wish to simply relink `rx` so that it uses local libraries then execute the `make relink` command and proceed to the next step.
  - c. If you wish to recreate `rx` from scratch then execute the `make clean` command followed by `make all`.
- ❑ 8. Switch users to "super-user" and execute `make install.all`; this will:
  - Install the `rx` program, the RXI default resource file, and any support programs needed by RXI users. (The command `make install` performs just this step.)
  - Install the `rx` man page. (The command `make install.man` performs just this step.)
  - Compiles the various fonts (`fixed-screen-*.bdf`) and places them into `$(FONTDIR)` directory so that the X server can find them. These are the normal and bold-face fonts used by `rx`. (The command `make install.fonts` performs just this step.)
- ❑ 9. On some VAXStations, the call to `xset` during the installation of `rx` causes the x server to "loose" it's fonts. If this situation exists, you will get errors about loading fonts when you run `rx`. Execute the command `xlsfonts` to list the fonts the x server knows about. If the message `Xlsfonts: pattern "*" unmatched` is displayed, you must reload the fonts. To reload the fonts, execute `xset fp default`
- ❑ 10. Install the termcap definition for the `rx` terminal type into your local termcap database. There is a file called `termcap` in the `rx` source area that contains the termcap definition for `rx`. (There is also a `terminfo` file if you are using terminfo on your system.) The `termcap` file is inserted into your local `/etc/termcap` file and the `terminfo` file is used by issuing the command `tic terminfo`.

---

## Installation using HP-UX and the X Window System

---

The steps involved in getting the rxi program running on a HP-UX workstation or fileserver running the X Window System are:

- ☐ 1. Create a temporary source directory for the rxi program on the workstation or fileserver. For example, if program sources are usually located in a directory called `/src` then the suggested place for rxi would be `/src/rxi`.
- ☐ 2. Traverse to `!Machine.Release.X_Interface.Release10_10_1` on the R1000.
- ☐ 3. Run the `Rxi_Install.Workstation_Files` procedure; this gets the files copied to the workstation or fileserver from the R1000. For example:

```
Rxi_Install.Workstation_Files
(Keyboard => Rxi_Install.Xhp46021a_hpux8_0,
 On_Machine => "my_hp",
 Username => "my_username",
 Password => "my_password",
 Account => "",
 Rxi_Source_Directory => "/src/rxi");
```

- ☐ 4. Connect to the source directory on the workstation and execute the command `make restore`. This unpacks rxi and makes it available for complete installation.
- ☐ 5. If this site has some or all of the optional licensed Rational products then edit the `RXI_Env_Menu` file. It's format is that of a C source file so it should look familiar. There is a `#define` line for each licensed product. For each licensed product at this site, set the corresponding `#define` to be equal to 1. This activates the rxi Environment menu entries for that product.
- ☐ 6. Edit the file called `Makefile`. A particular site may wish to change the places where rxi, its fonts, and/or its man page are installed. These can be changed by changing the definition of:

The defaults are:

```
BINDIR - where rxi executables will reside;
 usually /usr/bin/X11
HPBINDIR - where the HP X Window System executables reside;
 usually /usr/bin/X11
FONTDIR - where the compiled fonts reside;
 usually /usr/lib/X11/fonts/misc
MANDIR - the man page directory for the rxi man page;
 usually /usr/man/mann
```

If you don't know what directory to use for `FONTDIR`, run the `xdumpfp` program in the RXI source area. It will give you a list of all of the font directories being used by your X server.

If you want to place the RXI fonts into a special directory (eg. so that they are not in one of the normal server directories), you will have to notify the server of their location every time that you log in. This is easily done using the `xset` program. For example, if you put the RXI fonts into `/vendor/RXI/fonts` then you would need this command in your `.xinitrc` startup file (`.xinitrc` is one of the many names for startup files, see your System Administrator or read your system documentation to determine the proper file name for you to use):

```
xset fp+ /vendor/RXI/fonts
```

This command tells the server that additional fonts can be found in this directory. If you cannot find the `xset` program, the RXI source directory contains two programs that can be used to perform this function. They are not normally installed so you will have to copy them, by hand, to an appropriate bin directory. The command that you would use to run these programs is:

```
xsetfp 'xdumpfp' /vendor/RXI/fonts/
```

(Please notice the extra `'/'` on the end of the path name.) This command sets the server's font path to what it currently is, plus the new RXI font area.

The following directories may be changed but changes will only actually take effect when `rxl` is recompiled by performing "make clean" and "make all" steps.

```
XAPPLLOADDIR - where application defaults files reside;
usually /usr/lib/X11/app-defaults
```

An alternative to recompiling RXI is to have each user define one of the following environment variables in a manner similar to what is shown here. These examples assume that the RXI application defaults files have been placed into a directory named `/vendor/RXI/app-defaults`

```
csh> setenv XFILESEARCHPATH /usr/lib/X11/%T/%N%S:/vendor/RXI/%T/%N%S
csh> setenv XAPPLRESDIR /usr/lib/X11:/vendor/RXI/app-defaults
```

These two environment variables accomplish basically the same task.

- 7. You now have three more choices. A) Install the Rational-supplied `rxl` as-is, or, B) relink to use local (possibly shared) libraries, or C) completely recompile and recreate `rxl` using all local definitions. The supplied version of `rxl` should be able to execute on your system. However, it will not use any shared libraries, nor will it incorporate any local library fixes, extensions, or modifications that may exist.
  - a. If you wish to just use `rxl` as supplied then skip this step.

- b. If you wish to simply relink rxi so that it uses local libraries then execute  
`make relink` command and proceed to the next step.
  - c. If you wish to recreate rxi from scratch then execute `make clean` command followed by `make all`.
- 8. Switch users to "super-user" and execute `make install.all`; this will:
- Install the rxi program, the RXI default resource file, and any support programs needed by RXI users. (The command `make install` performs just this step.)
  - Install the rxi man page. (The command `make install.man` performs just this step.)
  - Compiles the various fonts (`fixed-screen-*.bdf`) and places them into `$(FONTDIR)` directory so that the X server can find them. These are the normal and bold-face fonts used by rxi. (The command `make install.fonts` performs just this step.)



---

## Installation using AIX and IBM X-Windows

---

The steps involved in getting the rxi program running on an IBM RS6000 workstation or fileserver are:

- ❑ 1. Create a temporary source directory for the rxi program on the workstation/fileserver. For example, if program sources are usually kept in a directory called `/src` then within this directory create a directory called `/src/rxi`.
- ❑ 2. Traverse to `!Machine.Release.X_Interface.Release10_10_1` on the R1000.
- ❑ 3. Run the `Rxi_Install.Workstaion_Files` procedure; this gets the files copied to the workstation or fileserver from the R1000. For example:

```
Rxi_Install.Workstaion_Files
(Keyboard => Rxi_Install.Xr6us_Aix3_1,
On_Machine => "My_Machine",
Username => "MyName",
Password => "MyPassword",
Account => "",
Rxi_Source_Directory => "/src/rxi");
```

- ❑ 4. Connect to the source directory on the workstation and execute the command `make restore`. This unpacks rxi and makes it available for complete installation. For example:

```
cd /src/rxi
make restore
```

- ❑ 5. If this site has some or all of the optional licensed Rational products then edit the `RXI_Env_Menu` file. It's format is that of a C source file so it should look familiar. There is a `#define` line for each licensed product. For each licensed product at this site, set the corresponding `#define` to be equal to 1. This activates the rxi Environment menu entries for that product.
- ❑ 6. Edit the file called `Makefile`. A particular site may wish to change the places where rxi, its fonts, and/or its man page are installed. These can be changed by changing the definition of:
  - `BINDIR` - where the rxi shell script goes; this is the twin to the aixterm script; usually `/usr/bin`
  - `AIXBINDIR` - where the AIX X Window System binaries currently reside; usually `/usr/bin`
  - `X11BINDIR` - where the real rxi goes; this is where most X11 executables go; usually `/usr/lpp/X11/bin`
  - `XAPPLOADDIR` - where the default resource file goes; usually `/usr/lib/X11/app-defaults`
  - `FONTDIR` - where the system-wide X11 fonts live; usually `/usr/lpp/fonts`
  - `MANDIR` - where the system-wide "new" man pages live;

usually /usr/man/mann

If you don't know what your system's standard `XAPPLLOADDIR` is, or, if you don't want to put RXI's files there, have each user define one of the following environment variables in a manner similar to what is shown here. These examples assume that the RXI application defaults files have been placed into a directory named `/vendor/RXI/app-defaults`

```
csh> setenv XFILESEARCHPATH /usr/lib/X11/%T/%N%S:/vendor/RXI/%T/%N%S
csh> setenv XAPPLRESDIR /usr/lib/X11:/vendor/RXI/app-defaults
```

These two environment variables accomplish basically the same task.

If you don't know what directory to use for `FONTDIR`, run the `xdumpfp` program in the RXI source area. It will give you a list of all of the font directories being used by your X server.

If you want to place the RXI fonts into a special directory (eg. so that they are not in one of the normal server directories), you will have to notify the server of their location every time that you log in. This is easily done using the `xset` program. For example, if you put the RXI fonts into `/vendor/RXI/fonts` then you would need this command in your `.xinitrc` startup file (`.xinitrc` is one of the many names for startup files, see your System Administrator or read your system documentation to determine the proper file name for you to use):

```
xset fp+ /vendor/RXI/fonts
```

This command tells the server that additional fonts can be found in this directory. If you cannot find the `xset` program, the RXI source directory contains two programs that can be used to perform this function. They are not normally installed so you will have to copy them, by hand, to an appropriate bin directory. The command that you would use to run these programs is:

```
xsetfp 'xdumpfp' /vendor/RXI/fonts/
```

(Please notice the extra `'/'` on the end of the path name.) This command sets the server's font path to what it currently is, plus the new RXI font area.

- ❑ 7. You now have three more choices. A) Install the Rational-supplied `rx` as-is, or, B) relink to use local (possibly shared) libraries, or C) completely recompile and recreate `rx` using all local definitions. The supplied version of `rx` should be able to execute on your system. However, it will not use any shared libraries, nor will it incorporate any local library fixes, extensions, or modifications that may exist.
  - a. If you wish to just use `rx` as supplied then skip this step.
  - b. If you wish to simply relink `rx` so that it uses local libraries then execute the `make relink` command and proceed to the next step.

- c. If you wish to recreate `rxl` from scratch then execute the `make clean` command followed by `make all`.

□ 8. Switch users to "super-user" and execute `make install.all`; this will:

- Install the `rxl` program, the RXI default resource file, and any support programs needed by RXI users. (The command `make install` performs just this step.)
- Install the `rxl` man page. (The command `make install.man` performs just this step.)
- Install the `rxl` terminfo description. This allows terminfo-using programs (such as `vi`) to operate within an `rxl` window. (The command `make install.terminfo` performs just this step.)
- Compile the various fonts `fixed-*.bdf` into the `$(FONTDIR)` directory. These are the normal and bold-face fonts used by `rxl`. (The command `make install.fonts` performs just this step.)

Some sites may not want to use the Rational supplied fonts. They can skip the `install.man` and `install.fonts` steps if they wish. They will want to edit the `RXI.Xdefaults` file to change the names of the default `rxl` fonts.

Some sites may not have the optional licensed IBM "man" product and they can skip the `install.man` step.

- 9. Run `rxl` and decide whether or not the default font is too small. Some IBM displays have a very small pixel size. This can make the default font too small for many users. RXI comes with 8 different fonts. They are:
- `fixed-screen-{r,b}-11` - default font - smallest "thin" font
  - `fixed-screen-{r,b}-12`
  - `fixed-screen-{r,b}-13`
  - `fixed-screen-{r,b}-14` - largest "thin" font - 1 pixel line thickness
  - `fixed-screen-{wr,wb}-15` - smallest "thick" font - > 1 pixel thickness
  - `fixed-screen-{wr,wb}-16`
  - `fixed-screen-{wr,wb}-17`
  - `fixed-screen-{wr,wb}-22` - largest "thick" font

You can try out the different fonts with command lines like these, restart your X server before trying this:

```
aix> rxl -fn fixed-screen-r-13 -fb fixed-screen-b-13 &
aix> rxl -fn fixed-screen-wr-15 -fb fixed-screen-wb-15 &
```

Always use matching normal (`-fn`) and bold (`-fb`) fonts or your `rxl` window may be unreadable when connected to an R1000.

If you decide to change the system-wide default font then edit the `RXI.xdefaults` file. Change the `*vt100.font:` and `*vt100.boldFont:` lines to reference the new font. You may also want to change then `*vt100.menuFont:` line if you find the menus to be unreadable as well. Then execute `make install` to install the new system-wide defaults.

---

## Installation using A/UX and the X Window System

---

The steps involved in getting the `rx`i program running on an Apple Macintosh workstation or fileserver running the X Window System are:

- ☐ 1. Create a temporary source directory for the `rx`i program on the workstation or fileserver. For example, if program sources are usually located in a directory called `/src` then the suggested place for `rx`i would be `/src/rxi`.
- ☐ 2. Traverse to `!Machine.Release.X_Interface.Release10_10_1` on the R1000.
- ☐ 3. Run the `Rxi_Install.Workstation_Files` procedure; this gets the files copied to the workstation or fileserver from the R1000. For example:

```
Rxi_Install.Workstation_Files
(Keyboard => Rxi_Install.Xapus_Aux4_0,
 On_Machine => "my_mac",
 Username => "my_username",
 Password => "my_password",
 Account => "",
 Rxi_Source_Directory => "/src/rxi");
```

- ☐ 4. Connect to the source directory on the workstation and execute the command `make restore`. This unpacks `rx`i and makes it available for complete installation.
- ☐ 5. If this site has some or all of the optional licensed Rational products then edit the `RXI_Env_Menu` file. It's format is that of a C source file so it should look familiar. There is a `#define` line for each licensed product. For each licensed product at this site, set the corresponding `#define` to be equal to 1. This activates the `rx`i Environment menu entries for that product.
- ☐ 6. Edit the file called `Makefile`. A particular site may wish to change the places where `rx`i, its fonts, and/or its man page are installed. These can be changed by changing the definition of:

The defaults are:

```
BINDIR - where rx executables will reside;
 usually /usr/bin/X11
AUXBINDIR - where the existing Apple X programs can be found;
 usually /usr/bin/X11
FONTDIR - where the compiled fonts reside;
 usually /usr/lib/X11/fonts
MANDIR - where the system-wide "new" man pages live;
 usually /usr/catman/x_man/man1
```

The following directories may be changed but changes will only actually take effect when rxi is recompiled by performing "make clean" and "make all" steps.

**XAPPLLOADDIR** - where the default resource file goes;  
usually **/usr/lib/X11/app-defaults**

An alternative to recompiling RXI is to have each user define one of the following environment variables in a manner similar to what is shown here. These examples assume that the RXI application defaults files have been placed into a directory named **/vendor/RXI/app-defaults**

```
csh> setenv XFILESEARCHPATH /usr/lib/X11/%T/%N%S:/vendor/RXI/%T/%N%S
csh> setenv XAPPLRESDIR /usr/lib/X11:/vendor/RXI/app-defaults
```

These two environment variables accomplish basically the same task.

If you don't know what directory to use for **FONTDIR**, run the **xdumpfp** program in the RXI source area. It will give you a list of all of the font directories being used by your X server.

If you want to place the RXI fonts into a special directory (eg. so that they are not in one of the normal server directories), you will have to notify the server of their location every time that you log in. This is easily done using the **xset** program. For example, if you put the RXI fonts into **/vendor/RXI/fonts** then you would need this command in your **.xinitrc** startup file (**.xinitrc** is one of the many names for startup files, see your System Administrator or read your system documentation to determine the proper file name for you to use):

```
xset fp+ /vendor/RXI/fonts
```

This command tells the server that additional fonts can be found in this directory. If you cannot find the **xset** program, the RXI source directory contains two programs that can be used to perform this function. They are not normally installed so you will have to copy them, by hand, to an appropriate bin directory. The command that you would use to run these programs is:

```
xsetfp `xdumpfp` /vendor/RXI/fonts/
```

(Please notice the extra **'/'** on the end of the path name.) This command sets the server's font path to what it currently is, plus the new RXI font area.

- 7. You now have three more choices. A) Install the Rational-supplied rxi as-is, or, B) relink to use local (possibly shared) libraries, or C) completely recompile and recreate rxi using all local definitions. The supplied version of rxi should be able to execute on your system. However, it will not use any shared libraries, nor will it incorporate any local library fixes, extensions, or modifications that may exist.

- a. If you wish to just use rxi as supplied then skip this step.

- b. If you wish to simply relink `rxl` so that it uses local libraries then execute `make relink` command and proceed to the next step.
- c. If you wish to recreate `rxl` from scratch then execute `make clean` command followed by `make all`.

□ 8. Switch users to "super-user" and execute `make install.all`; this will:

- Install the `rxl` program, the RXI default resource file, and any support programs needed by RXI users. (The command `make install` performs just this step.)
- Install the `rxl` man page. (The command `make install.man` performs just this step.)
- Compiles the various fonts (`fixed-*.bdf`) and places them into `$(FONTDIR)` directory so that the X server can find them. These are the normal and bold-face fonts used by `rxl`. (The command `make install.fonts` performs just this step.)

Some sites may not want to use the Rational supplied fonts. They can skip the `install.man` and `install.fonts` steps if they wish. They will want to edit the `RXI.xdefaults` file to change the names of the default `rxl` fonts.

---

## Installation using SunOS with X11/NeWS

---

The steps involved in getting the rxi program running on a Sun workstation or fileserver running XNews (OpenWindow) are:

- 1. Pick an RXI terminal type to install.

If all of the Sun workstations that will be using RXI to connect to R1000's have the Type 4 keyboard then install the `xNews4_v3_0_sparc` terminal type.

If all of the Sun workstations that will be using RXI to connect to R1000's have the Type 101A (IBM PC) keyboard then install the `xNews101_v3_0_sparc` terminal type.

If you will have a mixture of Type 4 and Type 101A keyboards that will all be using the same installed copy of RXI then you will have to do one of two things. Either install RXI as an `xNews4` terminal type or as an `xNews101` terminal type.

If you install RXI as an `xNews4` then:

- Type 4 keyboard users do nothing extra.
- Type 101 keyboard users should edit their `~/.xdefaults` file to contain a line that looks like this: `RXI*recognition:xnews101`.

If you install RXI as an `xNews101` then:

- Type 4 keyboard users should edit their `~/.xdefaults` file to contain a line that looks like this: `RXI*recognition:xnews4`.
- Type 101 keyboard users do nothing extra.

The line `RXI*recognition:xnews4` or `RXI*recognition:xnews101` tells RXI that when an R1000 asks the question "What type of terminal are you?", it should answer "xnews4" or "xnews101". RXI is built with a default answer that is determined by the `Rxi_Install.XNews?_v3_0` terminal type that you chose to install. This is a way to override that default.

Note: If the Type 4 keyboards are using one fileserver and the Type 101 keyboards are using another then you can simply install RXI as an `xNews4` on the one server and as an `xNews101` on the other server. Treat the two servers as two separate RXI installations as shown above. The R1000's should all have both the `xNews4` and the `xNews101` `Editor_Data` files installed.

NOTE to R1000 users using the model 4 keyboard with X11/NeWS:



Sun has seen fit to make some keys on the keyboard "identical" by default.

The keys "Help" and "F1" are defined, by the default X11/NeWS startup script, to be the same X "key symbol". This means that the R1000 cannot tell the difference between a user typing what he thinks of as "Help" and what he thinks of as "F1". This can be fixed by removing the following command from your `.xinitrc` file:

```
xmodmap -e 'keysym F1 = Help'
```

If you have already started NeWS then you can repair this on-the-fly with this command:

```
xmodmap -e 'keycode 12 = F1'
```

NOTE to R1000 users using the model 101A keyboard with X11/NeWS:

Sun has seen fit to deny 101A users a META key. Use this command to see whether or not your configuration has a META key.

```
xmodmap -pm
```

You should see a line that looks something like this in the printout:

```
mod1 Meta_L (0x7f)
```

"mod1" is the META key for all X window applications. If your mod1 key is not set, go read Sun's documentation on how to set up your 101A keyboard for "Sun Compatibility". If you have a file named `$OPENWINHOME/lib/Xmodmaprc.101A.sun` on your system, the comments in that file will tell you what to do.

Also, if you cannot find the `xmodmap` program, then the `xallkeys` program shipped with RXI can be used similarly. It can be found in the RXI source directory. It should already be compiled, just run it.

- ☐ 2. Create a temporary source directory for the rxi program on the workstation or fileserver. For example, if your various X11 application sources are located in `/src/x` then a likely place would be `/src/x/rxi`.
- ☐ 3. Traverse to `!Machine.Release.X_Interface.Release10_10_1` on the R1000.
- ☐ 4. Run the `Rxi_Install.Workstation_Files` procedure; this gets the files copied to the workstation or fileserver from the R1000. Use the `Rxi_Install.Xnews4_V3_0_Sparc` or `Rxi_Install.Xnews101_V3_0_Sparc` types. For example:

```
Rxi_Install.Workstation_Files
 (Keyboard => Rxi_Install.Xnews4_V3_0_Sparc,
 On_Machine => "nova",
 Username => "sagan",
```

```

Password => "verbose",
Account => "unlimited",
Rxi_Source_Directory => "/src/x/clients/rxi");

```

- ❑ 5. Connect to the source directory on the workstation and do the command **make restore**. This unpacks rxi and makes it available for complete installation.
- ❑ 6. If this site has some or all of the optional licensed Rational products then edit the **RXI\_Env\_Menu** file. It's format is that of a C source file so it should look familiar. There is a **#define** line for each licensed product. For each licensed product at this site, set the corresponding **#define** to be equal to 1. This activates the rxi Environment menu entries for that product.
- ❑ 7. Edit the file called **Makefile**. There are a series of "make" macros at the beginning of the file. These macros tell the installation script where the various pieces of RXI are to be installed. The default values for these macros may be correct for your site but they should be checked. The macros are:

```

OPENWIN - the "home" directory for Open Windows (X/NeWS);
 eg. /openwin
BINDIR - where rxi executables will reside;
 eg. /openwin/bin
FONTDIR - where the server fonts reside;
 eg. /openwin/lib/fonts
MANDIR - the man page directory for the rxi man page;
 eg. /usr/man/man1

```

NOTE: The **XAPPLLOADDIR** directory may be changed but changes will only actually take effect when rxi is recompiled by performing **make clean** and **make all** steps.

```

XAPPLLOADDIR - where application defaults files reside;
 eg. /openwin/etc

```

An alternative to recompiling RXI is to have each user define one of the following environment variables in a manner similar to what is shown here. These examples assume that the RXI application defaults files have been placed into a directory named **/vendor/RXI/app-defaults**.

```

csh> setenv XFILESEARCHPATH /usr/lib/X11/%T/%N%S:/vendor/RXI/%T/%N%S
csh> setenv XAPPLRESDIR /usr/lib/X11:/vendor/RXI/app-defaults

```

These two environment variables accomplish basically the same task.

If you don't know what directory to use for **FONTDIR**, run the **xdumpfp** program in the RXI source area. It will give you a list of all of the font directories being used by your X server.

If you want to place the RXI fonts into a special directory (eg. so that they are not in one of the normal server directories), you will have to notify the server of their location every time that you log in. This is easily done using the **xset** program. For example, if you put the RXI fonts into **/vendor/RXI/fonts** then you would need this command in your **.xinitrc** startup file (**.xinitrc** is one of the many names for startup files, see your System Administrator or read your system documentation to determine the proper file name for you to use):

```
xset fp+ /vendor/RXI/fonts
```

This command tells the server that additional fonts can be found in this directory. If you cannot find the **xset** program, the RXI source directory contains two programs that can be used to perform this function. They are not normally installed so you will have to copy them, by hand, to an appropriate bin directory. The command that you would use to run these programs is:

```
xsetfp 'xdumpfp' /vendor/RXI/fonts/
```

(Please notice the extra **'/'** on the end of the path name.) This command sets the server's font path to what it currently is, plus the new RXI font area.

- 8. You now have three more choices. A) Install the Rational-supplied **rx**i as-is, or, B) relink to use local (possibly shared) libraries, or C) completely recompile and recreate **rx**i using all local definitions. The supplied version of **rx**i should be able to execute on your system. However, it will not use any shared libraries, nor will it incorporate any local library fixes, extensions, or modifications that may exist.

- a. If you wish to just use **rx**i as supplied then skip this step.

- b. If you wish to simply relink **rx**i so that it uses local libraries then execute **make relink** command and proceed to the next step.

This assumes that your X libraries are in **/usr/lib**. If your X libraries are somewhere else, eg. in **/openwin/lib**, then define this environment variable first, eg. using the **cs**h shell:

```
SunOS> setenv LD_LIBRARY_PATH /openwin/lib:/usr/lib
SunOS> make relink
```

- c. If you wish to recreate **rx**i from scratch then execute the **make clean** command followed by a **make all**. Set the **LD\_LIBRARY\_PATH** environment variable if necessary (see above).

- 9. Switch users to "super-user" and execute **make install.all**; this will:

- Install the **rx**i program, the RXI default resource file, and any support programs needed by RXI users. (The command **make install** performs just this step.)

- Install the rxi man page. (The command `make install.man` performs just this step.)
  - Compile the various fonts (`fixed-screen-*.bdf`) into the `$(FONTDIR)` directory and then does a "bldfamily" there. These are the normal and bold-face fonts used by rxi. (The command `make install.fonts` performs just this step.)
- 10. Install the termcap definition for the rxi terminal type into your local termcap database. There is a file called `termcap` in the rxi source area that contains the termcap definition for rxi. (There is also a `terminfo` file if you are using terminfo on your system.) The `termcap` file is inserted into your local `/etc/termcap` file and the `terminfo` file is used by issuing the command `tic terminfo`.

---

## Installation using SunOS with MIT X11.R5

---

The installation steps for a Sun workstation or fileserver, running MIT X11.R5, are:

- ❑ 1. Create a temporary source directory for the RXI program on the workstation or fileserver. For example, if the X11 sources from MIT are located in a directory called `/src/x` then the suggested place for `rx`i would be `/src/x/clients/rxi`. If you would prefer to put the sources into a different place, eg. `/vendor/src/rxi`, then that works as well.
- ❑ 2. Traverse to `!Machine.Release.X_Interface.Release10_10_1` on the R1000.
- ❑ 3. Run the `Rxi_Install.Workstation_Files` procedure; this gets the files copied to the workstation or fileserver from the R1000. For example:

```
Rxi_Install.Workstation_Files
(Keyboard => Rxi_Install.Xsun4_X11R5_Sparc,
On_Machine => "nova",
Username => "sagan",
Password => "verbose",
Account => "unlimited",
Rxi_Source_Directory => "/src/x/clients/rxi");
```

- ❑ 4. Connect to the source directory on the workstation and execute the command **make restore**. This unpacks `rx`i and makes it available for complete installation.
- ❑ 5. If this site has some or all of the optional licensed Rational products then edit the `RXI_Env_Menu` file. It's format is that of a C source file so it should look familiar. There is a `#define` line for each licensed product. For each licensed product at this site, set the corresponding `#define` to be equal to 1. This activates the `rx`i Environment menu entries for that product.
- ❑ 6. You may wish to hand-edit the Rational-supplied **Makefile**. At the top of the **Makefile** is a list of the source, library, include, and font paths that will be used during installation. If you wish to change any of these default locations please do so now. The defaults are:

```
BINDIR - where rxi executables will reside;
 usually /usr/bin/X11
FONTDIR - where the server fonts reside;
 usually /usr/lib/X11/fonts/misc
MANDIR - the man page directory for the rxi man page;
 usually /usr/man/mann
XAPPLLOADDIR - where application defaults files reside;
 usually /usr/lib/X11/app-defaults
DESTDIR - the defaults filesystem to use, eg /usr, if you
 have installed X11.R5 somewhere special then just
 change DESTDIR, eg. DESTDIR=/vendor/X11R5
```

If you don't know what your system's standard `XAPPLLOADDIR` is, or, if you don't want to put RXI's files there, have each user define one of the following environment variables in a manner similar to what is shown here. These examples assume that the RXI application defaults files have been placed into a directory named `/vendor/RXI/app-defaults`

```
csh> setenv XFILESEARCHPATH /usr/lib/X11/%T/%N%S:/vendor/RXI/%T/%N%S
csh> setenv XAPPLRESDIR /usr/lib/X11:/vendor/RXI/app-defaults
```

These two environment variables accomplish basically the same task.

If you don't know what directory to use for `FONTDIR`, run the `xdumpfp` program in the RXI source area. It will give you a list of all of the font directories being used by your X server.

If you want to place the RXI fonts into a special directory (eg. so that they are not in one of the normal server directories), you will have to notify the server of their location every time that you log in. This is easily done using the `xset` program. For example, if you put the RXI fonts into `/vendor/RXI/fonts` then you would need this command in your `.xinitrc` startup file (`.xinitrc` is one of the many names for startup files, see your System Administrator or read your system documentation to determine the proper file name for you to use):

```
xset fp+ /vendor/RXI/fonts
```

This command tells the server that additional fonts can be found in this directory. If you cannot find the `xset` program, the RXI source directory contains two programs that can be used to perform this function. They are not normally installed so you will have to copy them, by hand, to an appropriate bin directory. The command that you would use to run these programs is:

```
xsetfp 'xdumpfp' /vendor/RXI/fonts/
```

(Please notice the extra `'` on the end of the path name.) This command sets the server's font path to what it currently is, plus the new RXI font area.

- 7. FOR NCD INSTALLS ONLY: Edit the file `Makefile` and modify the pathname for the `FONTDIR` entry. On some systems, add `/ncd` to the end of the pathname. On other systems, use the pathname `/usr/lib/X11/ncd/fonts/misc`. See the **Installation Overview** section for more information about finding the correct location for the fonts.
- 8. You now have three more choices. A) Install the Rational-supplied `rx` binary executable as-is, or, B) relink to use local (possibly shared) libraries, or C) completely recompile and recreate `rx` using all local definitions. The supplied version of `rx` should be able to execute on your system. However, it will not use any shared libraries, nor will it incorporate any local library fixes, extensions, or modifications that may exist. It will expect application resource files to exist in `/vendor/X11R5/lib/X11/app-defaults`, which may not be appropriate for your site (this can be overridden by use of the `XFILESEARCHPATH` environment

variable, see below).

- a. If you wish to just use rxi as supplied then skip this step.
- b. If you wish to simply relink rxi so that it uses local libraries then execute the `make relink` command and proceed to the next step. This assumes that your X libraries are in `/usr/lib`. If your X libraries are somewhere else, eg. in `/vendor/X11R5/lib`, then define this environment variable first, eg. using the csh shell:

```
SunOS> setenv LD_LIBRARY_PATH /vendor/X11R5/lib:/usr/lib
SunOS> make relink
```

- c. If you wish to recreate rxi from scratch then execute the `make clean` command followed by `make all`. Set the `LD_LIBRARY_PATH` environment variable if necessary (see above).

□ 9. Switch users to "super-user" and execute `make install.all`; this will:

- Install the rxi program, the RXI default resource file, and any support programs needed by RXI users. (The command `make install` performs just this step.)
- Install the rxi man page. (The command `make install.man` performs just this step.)
- Compiles the various fonts (`fixed-screen-*.bdf`) and places them into `$(FONTDIR)/bdf/misc` directory so that the X server can find them. These are the normal and bold-face fonts used by rxi. (The command `make install.fonts` performs just this step.)

- 10. Install the termcap definition for the rxi terminal type into your local termcap database. There is a file called `termcap` in the rxi source area that contains the termcap definition for rxi. (There is also a `terminfo` file if you are using terminfo on your system.) The `termcap` file is inserted into your local `/etc/termcap` file and the `terminfo` file is used by issuing the command `tic terminfo`.
- 11. If the rxi files were installed in the `/src/x/clients/rxi` hierarchy as suggested in step #2 then you will probably wish to edit the `Imakefile/Makefile` in the `/src/x/clients` directory. Add the rxi directory to the `SUBDIRS=...` list. This will cause rxi to be remade and reinstalled whenever all MIT X11 clients are remade.





# 2

---

## Workstation Installation Overview

---

There are two general classes of RXI-based terminal types and the steps in their installation may differ somewhat.

- *Real* workstations. For example, you might have a DEC workstation running the VMS operating system on your desk and that is the workstation that you would use when working with an R1000. This means that you would install one of the "real" RXI terminal types such as `Xdecus_Vms5_1_Wollongong502`.
- *Parasite* X-based terminals. For example, you might have an NCD X Terminal on your desk that keeps its font files and such on the file-system of a nearby DEC workstation running VMS. To use the NCD as your R1000 connection you would install one of the "parasite" terminal types for the NCD, such as `Xncd_Vms5_1_Wollongong502`.

Real RXI terminal types are installed using a set of terminal-type-specific keymap files on the R1000 and an RXI binary executable targeted to the real workstation.

Parasite RXI terminal types are installed using a set of terminal-type-specific keymap files on the R1000 and an RXI binary executable targeted to the *host* of the parasite terminal.

The R1000 side of things are specific to the target keyboard. The workstation/fileserver side of things are specific to the host machine of the target keyboard. Real workstations are their own hosts. Parasite terminals always have an external host machine. The external host is the machine that actually runs the RXI binary executable.

---

## R1000 Installation Overview

---

The following is intended as an overview of the RXI installation process. Please read the install note for the actual instructions for installing RXI. The general steps for installing a mythical XYZ terminal type are:

- Execute the `Rxi_Install.Machine_Editor_Data_Files` procedure to install the various XYZ keymap files on your R1000. This automatically enables any product-specific commands contained in the new keymap. Commands are enabled if the pertinent product is authorized for this R1000. The files would be:

```

!Machine.Editor_Data:
 Xyz_Commands : I Proc_Spec;
 Xyz_Commands : I Proc_Body;
 Xyz_Keys : Text;
 Xyz_Key_Names : I Pack_Spec;
 Xyz_User_Commands : Text;

```

This step also creates the appropriate XYZ entries in the **Terminal\_Types** and **Terminal\_Recognition** files in the same directory.

This step is performed for all real and parasite terminal types that will be used to talk to the R1000. It installs the keymaps for this terminal type and it tells the R1000 how to perform auto-recognition of the new terminal type when a user logs into the R1000.

This step is performed once for each RXI terminal type that your site will be using, real or parasite.

- Enable the RXI keymaps (Reboot the R1000 or run the **Refresh\_Terminal\_Information** command).
- Ship the workstation files to the workstation(s) and/or fileserver(s) where the workstation installation(s) will be performed. The **Rxi\_Install.Workstation\_Files** procedure is used to ship the files, using the R1000 FTP mechanisms.

---

## Workstation/Fileserver Installation Overview

---

The following is intended as an overview of the RXI installation process. Please read the install note for the actual instructions for installing RXI. The general steps for installing a mythical XYZ terminal type are:

- Create a temporary directory on the workstation or fileserver. This directory is used as a staging area during the installation. It can be deleted after the installation is complete. Typical installation space requirements are in the 10..20 Mb range (depending upon the target system).
- Ship the workstation files from the R1000 to this temporary directory.
- Log into the workstation/fileserver and connect to the temporary directory.
- Restore the backup file that was shipped from the R1000. (**make restore** on UNIX systems and **@makefile restore** under VMS.)
- Optionally change the default installation directories. RXI will be automatically installed in various system directories. These directories are appropriate for a large class of customers. However, some sites may have their

own, different, desired configurations. See Section 2, page 36 for details.

- Install RXI. As one step using **make install.all** on UNIX and **@makefile install.all** on VMS. Or, as several steps, **install** to install the binary executable and the system default resources file, **install.fonts** to install the RXI fonts, **install.man** to install the man page or help file, and **install.terminfo** to install a termcap description file for systems using the termcap library.
- Restart the X server(s) that will be used with RXI. They must be restarted so that they will discover the existence of the new RXI fonts. (Note: Some servers have a utility program, often called **xset**, that can be used for this purpose. These servers would not need to be restarted.)

---

## Workstation/Fileserver Components

---

There are four major components to the RXI product on the workstation, they are:

- The RXI executable(s)
- The RXI default resources file
- The RXI fonts
- The RXI man page or help file

These four components are installed, by default, on a single filesystem attached to a single workstation. However, some customers may have complex networks with scattered file servers and hundreds of workstations of different types.

Use the following to help determine where the various components must go.

### RXI Executable

The executables are installed by the **make install** (UNIX) or the **@makefile install** (VMS) command. This is one of the sub-steps performed by the **make install.all** (UNIX) or the **@makefile install.all** (VMS) command. It can be performed separately.

The RXI binary executable (a.k.a. the RXI program) goes on the filesystem(s) that serves the machine(s) that will provide the binary execution environment for RXI. These are the *host* machines discussed above. Most workstations will serve as their own hosts. X-based terminals always have an external host machine.

Most of RXI's target environments have a single RXI executable. However, some targets, such as IBM's AIX operating system, have two. The main RXI executable is an ordinary C program. The other RXI executable is a shell script whose purpose is to

a) know the exact physical location of the main RXI executable, and b) to run that executable in the background and then to exit. Under AIX, the main executable goes in one directory and the script file goes in another. See the individual target instructions for details.

The RXI executables must be placed where they are readily available to the host machines. If each host machine has its own separate and unique filesystem then, in the worst case, the RXI executables will need to be installed on each and every host machine's filesystem. However, most sites will be using one or more file servers. For those sites, the RXI executables would be installed on one or more of the file servers.

Only the customer has any way of knowing which file servers should be used for the installations. Carry out the workstation installation instructions once for each workstation or file server.

Note: It is much easier to repeat the installation instructions once for each file server than it is to attempt to install RXI once on one machine and then try to copy the installed files to each of the other file servers. It is very tempting to try to do this. It appears to be very easy. Do not fall into this trap. Perform the installation instructions once for each file server.

The executables should be placed in a directory that is visible to each user that will be running RXI (for example, `/usr/bin/x11`). If the directory is not visible (because it is protected or because it is not mounted on the user's host machine) then that user will not be able to run RXI.

Also, the executables should be placed in a directory that is on the default search path for user commands. If the RXI executables are placed in a directory which is not in the default search path, it will be necessary to modify each user's search path to gain easy access to the RXI executable.

## **RXI Default Resources File**

The default resources files are installed by the **make install** (UNIX) or the **@makefile install** (VMS) command. This is one of the sub-steps performed by the **make install.all** (UNIX) or the **@makefile install.all** (VMS) command. It can be performed separately.

The operation of the RXI program is governed to a large extent by the contents of the filesystem-wide *system defaults file*. This file is an X default resources file. Among other things, it tells RXI what its default terminal type is when logging into an R1000 and what its default window size should be. These default values govern the operation of all copies of RXI that are run using this particular filesystem. This may affect one workstation or it may affect hundreds; it depends upon how the customer network is arranged.

These files are read, by RXI, when RXI initializes itself. This means that these files must be present, on the host machine's filesystem, when RXI is run by the user. If these files are not present, RXI will still run, but it will probably not operate in quite

the expected fashion. See "Common Installation Problems", Section 3, page 41.

The RXI default resources files must be placed in the same area that other X Window System programs will be looking for their default resources files. This location is hard-wired into the Xlib library that is supplied as part of an X Window System implementation. See your vendor's documentation to find out the name of this directory.

When RXI is installed, the **RXI.xdefaults** file in the RXI source area is installed in the X application defaults area under the name **RXI**. A user can also have a local **.xdefaults** (UNIX) or **DECW\$XDEFAULTS.DAT** (VMS) file in their home directory. This local file can be used to override any or all of the resource values specified by the system defaults file.

See the discussion of **XAPPLoadDir** for more information about the X application defaults and RXI.

## RXI Fonts

The fonts are installed by the **make install.fonts** (UNIX) or the **@makefile install.fonts** (VMS) command. This is one of the sub-steps performed by the **make install.all** (UNIX) or the **@makefile install.all** (VMS) command. It can be performed separately.

The RXI fonts should be installed with the fonts that came with the customer's X Window System. These font files are read by the X server; they are not read by the RXI program.

- If a user is using a workstation directly (meaning that he will be sitting directly in front of the workstation console) then the fonts should be installed on that workstation's filesystem so that the workstation's X server can find them.
- If a user is using an X-based terminal, then the fonts are installed on the host machine, but they are installed in the font directory for the X-based terminal and not the fonts directory for the workstation.
- If a user is using one workstation (sitting in front of it), but is running RXI on some other machine on the network, then the user is using the first machine as if it were an X-based terminal and the second machine as the host. The fonts must be installed on the first machine because that machine is the one that is running the X server.

It is recommended that the RXI fonts be installed into the main font area for the X server(s). However, the RXI fonts may be installed in any location. In that case it will be necessary to inform the X server of the location of the RXI fonts before running RXI. See the discussion about **FONTDIR** below for more information on RXI font location.

## RXI Man/Help File

The man page (UNIX) or help library (VMS) is installed by the **make install.man** (UNIX) or the **@makefile install.help** (VMS) command. This is one of the sub-steps performed by the **make install.all** (UNIX) or the **@makefile install.all** (VMS) command. It can be performed separately.

The RXI man page (UNIX) should be placed into the usual man page area for the user system. Similarly, the RXI help library (VMS) should be merged with the main system help library.

The man page (help library) should be placed on the same filesystem as is used for the RXI executable(s). If a user expects to use some machine ABC to run the RXI program, then that user will also expect that machine ABC to provide help.

If RXI is being run on one machine, and the user is actually sitting in front of some other machine, do not install the man/help files on the other machine, especially if the two machines are not of the same type. The man/help files are not the same for every type of machine.

---

## Workstation/Fileserver Installation Directories

---

One step in the installation process is to review and possibly edit the **Makefile** (UNIX) or **Makefile.Com** (VMS) to change the default locations to be used when installing RXI. The Makefile lists the default names of the directories used to install RXI. If a customer is determined that RXI must be installed elsewhere, or if the X Window System software is not in the usual location, then:

- Determine where the binary executables, the resources files, the fonts, and the man/help files should be placed.
- Set **BINDIR** in the Makefile to indicate the place for the binaries.
- Set **XBINDIR** in the Makefile to indicate where the X Window System utility programs can be found (for use in installing RXI).
- Set **FONTDIR** to indicate the place for the fonts.
- Set **XAPPLOADDIR** and **XAPPLOADDIRstr** (if applicable) to be the same thing, the place for the resources files.

It is very important that you know the names of these directories before you attempt to install RXI. The following list describes the entries in the **Makefile** in greater detail. (All examples are "general case" and may not apply to your specific workstation type.)

### **BINDIR** Directory

**DESCRIPTION:** Where the RXI executable will reside.

**LOCATION:** The RXI executable may be installed in any location. It is recommended that the RXI executable be placed in a location which is visible to users in their default search paths. For example, a good location on Unix machines would be `/usr/bin` or `/usr/bin/x11`.

**ERRORS:** If the RXI executable is installed in a location that is not visible in the users' paths, or is protected, a message something like:

```
rx: Command not found.
```

or

```
Permission denied.
```

will be displayed when you try to run RXI.

## **XBINDIR Directory**

**DESCRIPTION:** Where the various X Window System utility programs reside. The installation of the RXI fonts requires the use of various X Window System utility programs usually found in this directory. These utilities compile and register the RXI fonts as appropriate for the workstation.

**LOCATION:** The location of the X Window System utilities varies from vendor to vendor. The Makefile contains an entry for the default location of the X Window System utilities for the vendor. If that directory is not present on the workstation/fileserver, look for a directory containing the programs that compile the fonts and which add the fonts to the font directory. The names of these programs are:

| Workstation   | Program Names                                    |
|---------------|--------------------------------------------------|
| Sun (MIT X11) | <code>bdftosnf</code> and <code>mkfontdir</code> |
| Sun (XNeWS)   | <code>dumpfont</code> and <code>bldfamily</code> |
| IBM RS6000    | <code>bdftosnf</code> and <code>mkfontdir</code> |
| Macintosh     | <code>bdftosnf</code> and <code>mkfontdir</code> |
| HP            | <code>xfc</code> and <code>mkfontdir</code>      |
| Dec Ultrix    | <code>bdftopcf</code> and <code>mkfontdir</code> |
| Dec VAX/VMS   | <code>font</code>                                |

**ERRORS:** An incorrect value for this entry will cause error messages to be generated during the installation. And, since the fonts will not be properly installed, this will also prevent the X server from finding the RXI fonts when RXI is run. RXI will request that the X server use its fonts and that request will fail. RXI will then print a message of the form: "X Toolkit Warning: Can not convert...". This usually indicates that the fonts were not properly installed or that the X server has not been restarted.

## FONTDIR Directory

**DESCRIPTION:** Where the RXI fonts will reside.

**LOCATION:** It is highly recommended that the RXI fonts be installed in one of same directories used for the X server fonts. Again, the location of the X server fonts varies from vendor to vendor. Also, the names of the files in the X server font directory vary with vendors. The directory containing the X server fonts will contain many files with similar names. For example, `fixed.snf`, `6x13.pcf` or even `Courier.Decw$Font`.

**ERRORS:** The installation procedure for RXI attempts to inform the current workstation's X server about the new RXI fonts. (Note: This may or may not be same the X server that normal users will be using.) If this fails, a message similar to: "X Toolkit Warning: can't convert ...." may be displayed when RXI is tested. If this happens, restart the X server to load the RXI fonts. On some systems you can shutdown then restart the X server, while on others you must log off to restart the X server. There may be other issues with restarting the X server running on a fileserver.

**CUSTOMIZATION:** It is possible to install the RXI fonts in a location other than that of the X server fonts. If the fonts are installed in another location, users must tell the X server where they were installed in order to use RXI. The method for locating the fonts in the X server varies from vendor to vendor. Refer to the reference manual for the customer's X Window System for details on how to do this. Some examples of how to add the new RXI fonts to the X server follow. Remember that these are examples only and may NOT work with your windowing system.

- Use the `xset` program to add the RXI fonts to the X server's font search path. For example, if the Fonts were installed in `/fonts/rxi`, then try the command `xset +fp /fonts/rxi`. The X server must be running for this to work. This will have to be used each time the X server is restarted.
- Put an `xset` command entry for the font directory in the user's `.xinitrc` file. (The name of this file also varies from vendor to vendor.) This file contains shell commands that are executed when the X server is started by this user.
- There may be other ways to add the RXI fonts to the X server. See the vendor's windowing system users guide for more information.

## XAPLOADDIR Directory

**DESCRIPTION:** Where the X application default files reside. The files `RXI` and `RXI_Env_Menu` will be put here.

The `RXI` file contains the various default control values that govern RXI's default behavior. Eg. the default window size and the default terminal type to use when logging into an R1000.

The `RXI_Env_Menu` file contains the Environment menu specification for use when the middle mouse button is used while logged into an R1000.



Changing the contents of either of these files will affect all users of this filesystem.

**LOCATION:** The location of the X application defaults directory varies on different workstations. Depending on what X applications are on this system, there may or may not be any files in this directory before installing RXI. Some common X applications that would have something here would be `xclock` or `twm`.

This location is not controlled by RXI. This location is hard-wired into the Xlib library that was supplied by the windowing system vendor. RXI uses this library and inherits this hard-wired location.

**ERRORS:** If the correct pathname for this directory is not used, the RXI Environment menu will be empty except for an error message about the missing file. If the message "No RXI Env Menu specified" is displayed when the middle mouse button is pressed in an RXI window while logged into an R1000, then the correct pathname was not used, the file has not been installed, the file contains garbage, the file has been deleted, or the file is protected. It will not be possible to customize RXI on a system-wide basis without these files in the correct location.

## **MANDIR Directory**

**DESCRIPTION:** Where the man pages (or help files) reside.

**LOCATION:** The location of the Man pages vary on different machines. The Makefile contains an entry for the default location of the Man pages on the system.

The man pages should be installed on the same machine/filesystem as the binary executable. If a user expects machine ABC to be able to run RXI then he will also expect that machine to provide help. RXI's man/help files differ depending upon the vendor.

**ERRORS:** If the RXI man pages were not installed in the proper location, the message "No manual entry for RXI" or "No help available for RXI" will be displayed when `man rxi` or `help rxi` is used.

## **OPENWIN Directory**

**DESCRIPTION:** Sun Microsystem OpenLook (a.k.a. OpenWindows, NeWS) systems only. Where OpenWindows resides.

**LOCATION:** `$OPENWINHOME`

The OPENWIN macro will use the current definition of the `OPENWINHOME` environment variable and RXI will install its binary, fonts, man page, and resources files directly into the OpenLook directories.

**ERRORS:** The definitions in the Makefile for OpenWindows (X/NeWS) applications assume that the installer is actually operating under the OpenLook/OpenWindows/NeWS system. This implies that you have the `$OPENWINHOME`

environment variable set. Do this command: `echo $OPENWINHOME`. If you get an error then you are not operating OpenLook.

If `FONTDIR` is not to be `$(OPENWIN)/lib/fonts` then each and every RXI user will have to tell the OpenLook server where the RXI fonts can be found. This will be done every time a user starts his OpenLook server. Read the OpenLook manuals to determine the command to use.

# 3

---

## Common Installation Questions and Problems

---

---

### Not using default location for RXI objects

---

RXI executables, RXI fonts, RXI resources, RXI man/help. See the "Workstation/Fileserver Components" section for a discussion of the location for RXI objects.

---

### Hetrogenous networks

---

RXI physically runs on machine A, user is sitting at machine B with the RXI window, RXI executables, resources, and help files go on machine A, the fonts go on machine B.

If there are lots of machine A's then they all get the fonts. If they all share a fileserver then that is good enough, put the fonts there.

If there are lots of machine B's then they all get the binary/help/defaults. If they all share a fileserver then that is good enough, put the files there.

---

### Using links for default resources files

---

Install the files wherever you want, then set up soft or hard links to those files in the app-defaults area.

---

### Key Foobar doesn't work

---

Use the Prompt-For key, then press the key that doesn't work. If the R1000 doesn't "see" the key (nothing happens), or if the R1000 gives the "wrong" name for the key, then the wrong terminal type is probably being used.

- Use the command `Io.Echo (System_Uilities.Terminal_Type);`. Does it print the expected terminal type name?
- What keyboard is being used? Check the installation instructions for RXI, is this keyboard layout supported?

- What operating system (and what version) is being used? Check the installation instructions for RXI, is this OS/version supported?
- What windowing system (and what version) is being used? Check the installation instructions for RXI, is this system/version supported?
- Use the Control-Middle-Mouse-Button menu, at the bottom, is the Rational Mode entry checked? If not, then you aren't really logged in as an RXI terminal type. Was the R1000 rebooted (or was the Refresh\_Terminal\_Information command executed) as per the instructions?
- Is there an entry for this terminal type in the !Machine.Editor\_Data.Terminal\_Type file?
- Is there an entry for this terminal type in the !Machine.Editor\_Data.Terminal\_Recognition file?

---

## Missing resources files

---

No menu, wrong default window size, wrong auto-recognition when logging in. See the "RXI Default Resources File" section and the "XAPPLOADDIR Directory" section for information about RXI default resources.

---

## Questions and Answers

---

**Q.** The X-Client and X-Server must be able to communicate with each other. It is not enough to have Vaxstations (VMS) with DECwindows as X-Server and a Sun with MIT's X11.R3 that could host an X-Client. This is because the X-Server expects clients to issue X Windows requests using the LAT protocol whereas X-Clients hosted on the Sun issue requests using the TCP/IP protocol.

Is this true? Is the X Windows protocol based on for instance TCP/IP or LAT which make X-Clients and X-Windows invisible to each other if they do not communicate on the same network protocol?

**A.** The X protocol doesn't, itself, care about the transmission medium. It is a simple stream-based protocol. A simple bi-directional stream of bytes. Things often get confusing here for the field because they often don't really understand network communications.

An X client is working with two different, and separate, protocols. First, it is communicating with an X server via some medium. That medium, today, is usually TCP/IP. TCP/IP is a communications protocol that is used to allow two machines to communicate. Neither the X server nor the X client actually deal directly with the TCP/IP protocol. Rather, they ask the operating system to open a connection to another machine and to use TCP/IP as the protocol. After that they are both dealing, in their

minds, simply with a stream of bytes. That stream of bytes has a structure, that is only known to them, that is called the X protocol.

The X protocol can be used across any communications medium that supports the transmission of bi-directional byte streams. This means that it can be sent across TCP/IP, SNA, X.25, and almost anything else you can think of.

However, the X server being used, and any X applications being used, must share a common communication protocol. Both must talk TCP/IP, or DECnet, or shared memory, or SNA, or dial-up modems, or something. Two programs that don't have a common communication protocol can't talk. This is true regardless of the type of the programs, X applications are just one example of this situation.

Folks seem to get the X protocol, and the underlying transmission method, or protocol, confused. An analogy that I sometimes use is that of two people talking on the phone. They don't care if the phones are analog or digital. They don't care if the phone company uses wire, glass fibers, or satellites. They just care that one person knows the phone number of the other and that they both talk the same language.

X programs are similar. They know a certain number of phone numbers (TCP/IP, SNA, DECnet, etc.) If they can call that number and get an answer then they can talk. (The names of the servers are like calling different area codes. You get directory assistance if you dial 555-1212 no matter what area code you use. But you get a different directory assistance operator.)

**Q.** The X-Server produces key events if the user uses the keyboard or mouse. These events are sent as numbers to RXI. The numbers depend on the used keyboard and hopefully not on the version of X Windows on the X-Server.

**A.** Sorry, they depend exactly on the X server. It is the X server that determines the symbolic name(s) of each key. Those symbolic names, encoded as 16-bit numbers, are what RXI uses to talk to R1000's. It is also those symbolic names that allow a user to rearrange his keyboard at will. He can do this slightly, such as eliminating an annoyingly placed Caps-Lock key, or he can do this radically, such as changing his keyboard into a Dvorak layout.

**Q.** X-Windows defines a big list of possible keys consisting of a logical name such as Escape, Execute, or Undo, and corresponding numbers.

**A.** They call them "symbols", or symbolic names. They are supposed to correspond to the names commonly found printed on the keys themselves. (There is very little logic in some of the names.)

**Q.** Different versions of X-Windows should send the same number to the XClient if someone presses the Undo button on a Sun4 keyboard for instance. In the other direction, Xrequests coming from the RXI X-Client are drawn on the screen in the RXI window by the X-Server.

**A.** This may or may not be true. "presses the Undo button" is ambiguous. The answer is one way if you mean "the key with UNDO printed on it" and the answer is another way if you mean "the key with the symbolic name UNDO". Different versions, or versions from different vendors, of the X server will typically send the same "key code", or numeric value to an X client if the same key is pressed.

However, whether or not a key is an "Undo" key is a function of the symbolic name that is currently attached to a key by the server. Since the user is free to (is actually encouraged to) change the symbolic names of keys, we must work with the symbolic names and not the physical key codes.

By the way, there is no guarantee that two different X servers will use the same numeric key codes for the same physical key. They will \*typically\* report to the X client, the physical hardware key number that was given to them by the hardware. This is not required or guaranteed. However, some keyboards do not report simple numbers to their hardware drivers when a key is pressed. So, for some keyboards, the so-called "physical key code" is a figment of the mind of the X server and so cannot be considered to be a constant common to all X servers for that machine.

It's like trying to get a firm hold on a ghost. You see it, but there's nothing really there to grip.

**Q.** Can different versions of X Windows X-Servers send different key numbers for a key on the same physical keyboard?

**A.** Let's use the X terminology.

The so-called physical key code (a numeric value in the 8..255 range, (0..7 are reserved by the protocol)) is usually the same across X servers. But there is no guarantee and there are known machines where, if a 2nd vendor ever appeared, the key numbers could easily change. The DEC keyboard comes to mind here; the hardware reports keyboard X/Y coordinates for keys and not simple key numbers. The key code for the X server is something constructed from a table. Tables can change.

The very-non-physical key symbol (also a numeric value, but in the 0..2\*\*16-1 range) is something that users can, and do, change at will. Key symbols are the defined way of making applications semi-independent of keyboards.

**Q.** If X-Server and X-Client have different versions of X Windows, which is relevant for the choice of the installation of the RXI application?

**A.** Both. (And isn't that annoying!)

The vendor, and version, of the X server is important because that is what determines:

- the RXI keymap that you install on the R1000

- which form of the fonts you have to install for consumption by the X server.

The vendor, and version, of the X libraries on the workstation that executes the RXI program is important because that is what will be used when RXI executes. If a vendor supplied shared-libraries then the customer will typically want to relink RXI to use those shared libraries in order to reap the runtime swap space benefits that result.

The vendor, and version, of the workstation operating system are also relevant here. RXI must be able to execute on the machine.

Also note, the version of the X libraries and the version of the operating system can be independent of each other. This means that if some X vendor has two different servers, and if they both run under two versions of some OS vendor's operating system, then potentially, we could have four different versions of RXI (!!!) that we have to support.

**Q.** The X-Client is the actual RXI application. For hosting the RXI application the following items are relevant:

- a version of X Windows
- an operating system
- the help facility
- in case of Vax VMS, the TCP/IP communication package for communication with the R1000

The RXI application is basically a terminal simulation. The user must start this RXI application on the machine on which it was installed. The usual way in which this happens is: The user uses some remote login to connect from the X-Server that he is using to the RXI host. There he starts RXI with the option to open the display on the X-Servers screen. The RXI application simulates a terminal in the RXI window and the User still has the prompt of the RXI host in this window. To connect to the R1000, the user typically uses telnet from this RXI window.

The RXI application for Vax VMS is a bit different, it builds the connection to the R1000 itself since Vax VMS does not allow one to build a terminal simulation that sets itself between the user and the operating system. Hence the dependency on the communication package for the communication between the RXI application and the R1000.

The RXI application turns keypress events coming from the X-Server into escape sequences. Normal characters in the ascii range are passed along unchanged. keynumbers corresponding to other keys are transformed into escape sequences. It will do this for any valid X Windows keynumber even if it does not belong to the keyboard type for which the RXI application was installed. This allows the user to build a telnet

connection to the R1000 from the RXI window, so the R1000 will get these escape sequences.

Does this mean that one could install an RXI application of the type Xsun4\_X11r3 on a Sun and use for instance Apollo workstations as X-Servers, as long as one installs the correct keymaps on the R1000?

**A.** Yes. RXI cares about its own personal execution environment only to the extent that it must be compiled using the correct compiler and linked with the correct libraries. It must be able to execute before it can do anything else (like talk to an R1000 or open a window on a server).

After that, it can place its window on any X server that is willing to talk to it and it can connect to any R1000 that is willing to listen. RXI literally does not care one little bit about what kind of keyboard is being used, or what kind of mouse is being used, or how they are set up. Whatever it receives from the X server it will blindly pass along to the R1000.

RXI does its very best to act like a simple piece of wire between the user's keyboard/mouse and the R1000. At most, RXI could be considered to be a protocol translation mechanism between what the X server produces and what the R1000 consumes. The translation is very simple and totally unintelligent.

By the way, terminology again:

RXI receives events that contain "key codes". These codes are looked up in a table, provided by the X server, and they are translated into "key symbols". These symbols are 16-bit numeric values. They, and any associated Shift, Control, and Meta flags that may be present, are taken, as a 19-bit number. That 19-bit number is broken up into 2..4 pieces and those pieces are encoded as printable ASCII characters. Those characters are then sent to the R1000. The phrase "escape sequence" is typically reserved for ANSI ASCII standard terminal sequences that begin with the character ESCAPE. RXI does not use ESCAPE because it is not a printing character.

**Q.** The R1000 translates escape sequences into symbolic keynames using the files in !Machine.Editor\_Data. It doesn't even know that certain installed keymaps correspond to physical terminals and other correspond to the X-Server keyboards whose keynumbers are translated to escape sequences by the RXI application. Since different keyboard cause the X-Server to send different keynumbers, a keymap must be installed for each X-Server keyboard that will be used using RXI.

Is this true? Is the combination of R1000 keymap and X-Server currently used actually independent of the RXI application and where it runs?

**A.** Yes, the keymap reflects the combination of a particular vendor/version X server and a particular vendor/version/OS keyboard. The version of RXI, and the vendor/version/OS that is being used to run it, are irrelevant. RXI tries to be just a wire between any two points.



(Note: If you get historical, then versions of RXI prior to rev10\_4 used a different encoding scheme for mouse actions. However, I don't think anyone is using such an ancient RXI anymore. It was still independent of the keyboard, but the keymap files on the R1000 would be different than more recent RXI keymaps.)

**Q.** If this is not the case, you might be forced to install two RXI application on the RXI host to support different X-Servers.

**A.** This *should* not be the case. However, some vendors might have a very-old, or even a very-new, X server. In this case it is possible that one version of RXI would be required for one and another for the other. This would be due to protocol differences.

This has never quite happened. When MIT put out X11R4 it broke RXI. Old versions of RXI would not work with the new MIT server. However, the new RXI, that works with the new server, will work just fine with any old server. Fortunately, to date, all protocol changes have been backwards compatible. That is, a client that works with the new protocol will also work with the old protocol.

So, getting a second X server might cause a customer to request a new version of RXI, but, when the new one was installed, the old one would just go away.

**Q.** Where does the mouse come in here? Isn't it the case that

- the RXI application translates mouse events in escape sequences
- mice can be different for different X-Servers (two or three buttons for instance)

hence the RXI application depends on the choice of the X-Server?

**A.** RXI turns mouse events into a series of keystrokes. When the user pushes a mouse button, the R1000 will see a series of keystrokes come in.

These keystrokes use character sequences that cannot be generated by any X protocol key event. These "mouse keys" were invented and defined in such a way that they are outside the set of character sequences generable by all possible X key events and do not conflict with them. They provide a means whereby RXI can generate fake keystrokes that are not generable by the user. This allows RXI to report mouse events to the R1000 without confusion or conflict.

The X protocol provides events that tell an X client that Button1 (or Button2, or Button3, or Button4, or Button5) has been pressed (or released). Like with keystrokes, RXI merely translates what it receives and fires this off to the R1000. RXI does not interpret.

So, if some mouse has only two keys, and if the X server does not provide any way to fake a third key, then a user with that mouse has only two keys. It is very much the

same situation of you have one user with 10 function keys and another user with 9 function keys. One user has one less function keys and that is that.

Most X servers that support 2-button mice will also provide a 3rd fake key. You press the fake 3rd key by simultaneously pressing both keys. In a case like this, the X client has no idea what is going on. It only knows that it got an event that said "Button3 goes down".

An X client has no way of knowing:

- what the keyboard looks like physically,
- how many keys the keyboard really has,
- how many buttons are on the mouse,
- whether the "mouse" is really a "light pen" or something else

All it knows, all it can know, is "key number 93 went down" and "Button2 went up" and things like that. It only knows what the X server is willing to tell it. There is no way to query about hardware configuration.

# A

---

## Procedure Do\_Step

---

The majority of the steps which must be performed for this installation utilize the procedure `Do_Step` (included as part of the release tape and located in `!Commands`) to execute the required sequence of commands which implement the step. This procedure **must always** be executed from a command window in the release library

```
!Machine.Release.Archive.X_Interface.Release10_10_1
```

Each step which utilizes procedure `Do_Step` will be of the form

```
<STEP_NAME> [<TIME TO EXECUTE>]
 <DESCRIPTION>
```

where `<STEP_NAME>` is passed as the parameter to procedure `Do_Step` which performs the necessary actions to complete the step, `<TIME TO EXECUTE>` is the amount of time the step takes to execute (this may be expressed as a range, in which case the lower values is typically the minimum, and the upper value is an estimate of the maximum), and `<DESCRIPTION>` is a description of what the step does, including any action which you will need to manually perform. For example:

FOO

*[5 Minutes]*

This step is an example of the format used for steps which are executed using procedure `Do_Step`. To execute step FOO, you would go to `!Machine.Release.Archive.X_Interface.Release10_10_1` and in a command window, execute:

```
Do_Step ("FOO");
```

which would result in automatic execution of all commands required to implement this step. If you needed to take any manual action in addition to executing `Do_Step`, it would be noted as :

□ Perform some manual check

If any errors occur for this step, you should always fix the problem before proceeding on to the next step. Each step is defined by a fragment of Ada code which is executed by `Program.Run`. These fragments are stored in the `Steps` file

located in the `Command_Data` library of the release. In the event you want to modify a step, you can use the "PROMPT => <STEP>" form when invoking `Do_Step`. See the spec of procedure `Do_Step` (located in `!Commands`) for more detailed information.

**Warning:** *If you interrupt the execution of `Do_Step` (by using `Job.Interrupt` such as `CONTROL-G`) it is possible that certain interactive commands executed by some steps, such as `Common.Definition`, may fail with an exception and display a message such as*

*Unable to read file due to Constraint\_Error (Null Access)*

*In general, this message has no negative impact on the execution or completion of the step, and can be ignored.*

**Note:** *When multiple steps are executed, you will be prompted between steps with the following:*

*Continue with <Step Name> step (Continue | Skip | Quit)? [Continue] : [input]*

*Select the action to be taken, one of `Continue`, `Skip`, or `Quit`. `Continue` (the default if `PROMOTE` is entered without typing anything) results in the step <Step Name> being executed. `Skip` will skip <Step Name> and prompt with the step following <Step Name>. `Quit` results in termination of `Do_Step` without further steps being executed.*

# B

---

## INSTALL\_X\_INTERFACE

---

The step Install\_X\_Interface runs the following steps in this order.

- ❑ 1. RESTORE\_NOTES *[1 minute]*  
This step restores the X\_Interface Release Notes into `!Machine.Release.Release_Notes`.  
  
*Note: Use the "RAW" option when printing the \_Ps copy of the release note.*
- ❑ 2. RESTORE\_KEYMAP\_OVERLAYS *[1 minute]*  
This step restores the X\_Interface Keymap Overlays (Inserts) into `!Machine.Editor_Data.Keymap_Overlays`.  
  
*Note: Use the "RAW" option when printing the \_Ps copy of the overlays.*
- ❑ 3. RELEASE\_RESTORE *[30 minutes]*  
This step restores the archives. When completed, a filtered error log is displayed. Examine this log for errors; ignore errors about switches and links.
- ❑ 4. RECORD\_INSTALLATION *[1 minute]*  
This step records release information in `!Machine.Release.Current.Products` and sets the login limit back to unlimited logins. This is a required step and establishes that the release has successfully been installed.



# C

---

## Supported Configurations

---

The customer site must have all of these before an RXI installation can proceed.

- A workstation/fileserver (or an X-based terminal with a separate host workstation), with:
  - one of the supported machine architectures,
  - running one of the supported operating system versions,
  - and, TCP/IP network connections allowing it to connect to the R1000(s).
- The workstation, operating system, and TCP/IP must be installed and operational before RXI can be installed.
- A workstation or an X-based terminal with:
  - A supported keyboard,
  - and, a supported version of the X Window System.
- The workstation/terminal and the X Window System must be installed and operational before RXI can be installed.
- R1000 with Environment release D\_12\_5\_0 or later.

RXI has been developed and tested on the following workstation configurations. This release of RXI may work with other versions of operating or window systems. However, as only these configurations have been tested, Rational can not support, or guarantee that other configurations work.

Note: These configurations can be used in any combination. That is, the RXI executable can be running on any supported binary host machine, while, the RXI window appears on any supported X server, allowing, the user to use any supported keyboard attached to that server, connected with, any R1000 with an installed keymap for that keyboard/X-server combination.

| <b>Workstation</b> | <b>Operating System</b> | <b>Window/ Network System</b>     | <b>Keyboard Type</b> | <b>RXI Terminal Type</b>          | <b>Workstation Install Section</b> |
|--------------------|-------------------------|-----------------------------------|----------------------|-----------------------------------|------------------------------------|
| DEC DECstation     | Ultrix V4.1             | DECwindows                        | LK201                | Xultus_Ds_Ultrix4_1               | page 9                             |
| DEC VAXstation     | Ultrix V4.1             | DECwindows                        | LK201                | Xultus_Vs_Ultrix4_1               | page 9                             |
| DEC VAXstation     | VMS 5.4                 | DECwindows, Wollongong 5.0.2      | LK201                | Xdecus_Vms5_4_Wollongong502       | page 6                             |
| DEC VAXstation     | VMS 5.4                 | DECwindows, UltrixConnection 1.3A | LK201                | Xdecus_Vms5_4_Ultrixconnection13a | page 6                             |
| HP                 | HP-UX 8.0               | X Windows                         | 46021A               | Xhp46021a_Hpux8_0                 | page 12                            |
| IBM RS6000         | AIX 3.1                 | AIX/R2 X-Windows 3.1              | IBM U.S.             | Xr6us_Aix3_1                      | page 15                            |
| Macintosh          | AUX 4.0                 | AUX/R, X-Windows                  | Apple U.S.           | Xapus_Aux4_0                      | page 19                            |
| Sun Sparc          | SunOS 4.1.2             | Sun X11/NeWS V3.0 (Openwindows)   | Type 4               | Xnews4_V3_0_Sparc                 | page 22                            |
| Sun Sparc          | SunOS 4.1.2             | Sun X11/NeWS V3.0 (Openwindows)   | Type 101             | Xnews101_V3_0_Sparc               | page 22                            |
| Sun Sparc          | SunOS 4.1.2             | MIT X11.R5                        | Type 4               | Xsun4_X11r5_Sparc                 | page 27                            |



# D

---

## Feedback

---

In order to help us improve our instructions and make installations easier to do, the following form is provided to allow you to give us feedback. Please complete and send along with a printout of the file `Do_Step_Execution_Time` located in `!Machine.Release.Archive.X_Interface.Release10_10_1.Logs` to:

Rational  
Atten: SMSE  
3320 Scott Blvd.  
Santa Clara, CA 95054-3197  
Re: X\_Interface

- ☐ 1. How long did the installation take you?
- ☐ 2. Did the installation proceed as described in the instructions?
- ☐ 3. What could be done to improve these (or other) instructions/installations?
- ☐ 4. What did you like about this set of instructions/installation?

# Rational X Interface Release Information

Release10\_10\_1 Release

Copyright © 1989-1992 by Rational

Part Number: 508-003219-006

09/30/92 (Software Rev. Release10\_10\_1)

Rational and R1000 are registered trademarks and Rational Environment and Rational Subsystems are trademarks of Rational.

UNIX is a registered trademark of AT&T in the USA and other countries.

Apple, A/UX, and Macintosh are registered trademarks of Apple Computer, Inc.

DEC, DECnet, DECstation, DECwindows, ULTRIX, ULTRIXConnection, VAX, VAXstation, and VMS are trademarks of Digital Equipment Corporation.

Excelan is a registered trademark of Excelan, Inc.

AIX, AIXwindows, IBM, and RISC System/6000 are trademarks of International Business Machines.

The X Window System is a registered trademark of the Massachusetts Institute of Technology.

NCD16 and NCD19 are trademarks of Network Computing Devices, Inc.

NeWS and Sun are registered trademarks, and, Sun Workstation, SunOS, Sun-3, and SPARC are trademarks of Sun Microsystems, Inc.

MultiNet is a registered trademark of TGV, Inc.

WIN/TCP is a trademark of The Wollongong Group, Inc.

Ethernet is a trademark of the Xerox Corporation.

Rational  
3320 Scott Boulevard  
Santa Clara, California 95054-3197

## Contents

|                                              |   |
|----------------------------------------------|---|
| 1. Overview . . . . .                        | 1 |
| 2. Configurations . . . . .                  | 1 |
| 2.1. Apple . . . . .                         | 1 |
| 2.2. Digital Equipment Corporation . . . . . | 2 |
| 2.2.1. DEC VMS . . . . .                     | 2 |
| 2.2.2. DEC ULTRIX . . . . .                  | 2 |
| 2.2.3. Installation Options . . . . .        | 3 |
| 2.3. Hewlett-Packard . . . . .               | 3 |
| 2.4. IBM . . . . .                           | 3 |
| 2.5. NCD . . . . .                           | 4 |
| 2.6. Sun Microsystems . . . . .              | 4 |
| 3. Compatibility . . . . .                   | 5 |
| 4. Upgrade_Impact . . . . .                  | 5 |
| 5. Known Problems . . . . .                  | 6 |
| 6. New Features . . . . .                    | 6 |
| 7. Changes . . . . .                         | 6 |
| 8. Documentation . . . . .                   | 6 |
| 9. Copyrights . . . . .                      | 7 |



## 1. Overview

The Rational X Interface (RXI) allows users to access the Rational Environment via TCP/IP on their Ethernet from any of the supported workstation or X terminal platforms.

The Release10\_10\_1 release of the Rational X Interface supports the following platforms:

- Apple - MacII running A/UX; extended keyboard.
- DEC - all VMS 5.4 platforms using ULTRIXConnection or Wollongong TCP/IP software.
- DEC - ULTRIX 4.1 for the DECstation or the VAXstation architectures.
- Hewlett-Packard - 9000 series 300/800.
- IBM - RISC System/6000.
- NCD - NCD16 and NCD19 hosted by any other supported platform.
- Sun - workstation models: 4 (SPARC).

See the Configurations section below for a complete listing of supported workstation/X-terminal hardware configurations.

This is the first production release of the DEC ULTRIXConnection support.

Installation instructions for RXI may be found in the Rational X Interface Release10\_10\_1 Installation Procedures document(s).

## 2. Configurations

Release10\_10\_1 is compatible with Rational Environment versions D\_12\_5\_0 or later.

### 2.1. Apple

RXI is built and tested on a MacII running A/UX with the extended keyboard. It should operate correctly on any Apple 68k platform running A/UX. It requires:

- A/UX 4.0.
- A/UX X Windows version 4.0.
- Apple extended keyboard; other keyboards are not presently supported.

A/UX must be installed and running before RXI can be successfully installed.

A/UX X Windows must be installed but need not be actually running before RXI can be successfully installed.

RXI can be installed in two ways on A/UX.

A "binary" installation simply installs the Rational-supplied RXI executables on the MacII.

A "full source" installation recompiles RXI from the C sources.

## **2.2. Digital Equipment Corporation**

### **2.2.1. DEC VMS**

RXI for VMS is built and tested on a VAXstation 3100. It should operate correctly on any VMS platform. It requires:

- VMS 5.4
- DECwindows 5.4
- DEC standard LK201 keyboard; other keyboards are not presently supported.
- Either of:
  - Digital ULTRIXConnection version 1.3A
  - Wollongong WIN/TCP version 5.0.2.

VMS must be installed and running before RXI can be successfully installed.

ULTRIXConnection or WIN/TCP must be installed and running before RXI can be successfully installed.

DECwindows must be installed but need not be actually running before RXI can be successfully installed.

### **2.2.2. DEC ULTRIX**

RXI for ULTRIX is built and tested on both a DECstation 3100 and a VAXstation 3100. It should operate correctly on any DECstation or Vax platform. It requires:

- ULTRIX 4.1.
- DECwindows for ULTRIX 4.1.
- DEC standard LK201 keyboard; other keyboards are not presently supported.

ULTRIX and TCP/IP must be installed and running before RXI can be successfully installed.

### 2.2.3. Installation Options

RXI can be installed in three ways on VMS or ULTRIX:

A "binary" installation simply installs the Rational-supplied RXI executables on the VMS platform.

An "object module" installation relinks RXI with local shared libraries for increased runtime efficiency.

A "full source" installation recompiles RXI from the C sources.

Both the "object module" and the "full source" installation require a C license from DEC. The "binary" installation may be performed without a C license. RXI can be installed as a "sharable image" using DEC's Install facility; this decreases the overall memory requirements when running multiple copies of RXI on one platform.

### 2.3. Hewlett-Packard

RXI is built and tested on an HP 9000 series 300. It should operate correctly on any HP 9000 platform. It requires:

- HP-UX 8.0.
- HP's X Window System for HP-UX 8.0.
- Model 46021A keyboard; other keyboards are not presently supported.

HP-UX must be installed and running before RXI can be successfully installed.

The X Window System must be installed but need not be actually running before RXI can be successfully installed.

RXI can be installed in three ways on HP-UX.

A "binary" installation simply installs the Rational-supplied RXI executables on the 9000 platform.

An "object module" installation relinks RXI with local shared libraries for increased runtime efficiency.

A "full source" installation recompiles RXI from the C sources.

### 2.4. IBM

RXI is built and tested on a RISC System/6000 model 530. It should operate correctly on any RISC System/6000 platform. It requires:



- AIX 3.1.
- AIXwindows version 3.1.
- IBM standard PC keyboard; other keyboards are not presently supported.

AIX must be installed and running before RXI can be successfully installed.

AIXwindows must be installed but need not be actually running before RXI can be successfully installed.

RXI can be installed in three ways on AIX.

A "binary" installation simply installs the Rational-supplied RXI executables on the RISC platform.

An "object module" installation relinks RXI with local shared libraries for increased runtime efficiency.

A "full source" installation recompiles RXI from the C sources.

The IBM X-120 X terminal is also a supported platform. The X-120 keyboard is physically identical to the RS/6000 keyboard. In addition, the software keyboard translation tables within the X server for the RS/6000 and the X server for the X-120 are sufficiently close that the XR6US terminal type on the R1000 is used for both keyboards.

## 2.5. NCD

RXI supports the NCD as yet-another-keyboard attached to any supported workstation platform. RXI is installed for an NCD by installing the NCD-specific files on an R1000 and then by installing the appropriate workstation files on some host workstation. RXI actually runs on the host workstation and the RXI window and menus appear on the NCD screen.

The only difference between RXI on a platform and RXI on an NCD hosted by that same platform is the type and layout of the keyboard.

RXI supports the NCD16 and NCD19 terminals using the NCD101 keyboard.

The NCD terminal must be installed and fully operational before RXI can be successfully installed.

## 2.6. Sun Microsystems

RXI is built and tested on a Sun SPARCstation 2. It should operate correctly on any model SPARC platform. It requires:

- SunOS 4.1.2. (4.1.1 will not work.)

- Either of:
  - X Window System from MIT, version 11, release 5.
  - X11/NeWS version 3.0.
- Model 4 or model 101A keyboard; other keyboards are not presently supported.

SunOS must be installed and running before RXI can be successfully installed.

The windowing system must be installed but need not be actually running before RXI can be successfully installed.

RXI can be installed in three ways on SunOS

A "binary" installation simply installs the Rational-supplied RXI executables on the Sun platform.

An "object module" installation relinks RXI with local shared libraries for increased runtime efficiency.

A "full source" installation recompiles RXI from the C sources.

### 3. Compatibility

Release10\_10\_1 is fully compatible with Rational Environment version D\_12\_7\_2. Release10\_10\_1 is **not** compatible with Rational Environment versions prior to D\_12\_5\_0.

Keybindings for optional products are only available if the optional products are available for the version of the Environment being used.

Apollo workstations are no longer supported.

MultiNet TCP/IP is no longer supported on VMS.

Sun-3 workstations and the Sun Type 3 keyboard are no longer supported.

### 4. Upgrade\_Impact

Release10\_10\_1 of RXI completely replaces all previous versions for all platforms. Installation will replace all existing files and programs. Any local customizations that may have been performed will be overwritten by the new files and programs.

## 5. Known Problems

The Motif window manager requires most "meta" function keys for its own use. (A meta function key is any function key that is pressed while the "meta" key is also pressed.)

This means that user applications (RXI included) cannot use M\_F1, M\_F2, etc. The HP 9000 and the IBM RISC System/6000 versions of RXI are configured to be usable under Motif. This means that the R1000 keymaps for these configurations do not have any Rational Environment functions bound to the meta function keys.

## 6. New Features

ULTRIXConnection 1.3A is officially supported.

The new Sun Type 101A keyboard is supported.

## 7. Changes

There are no changes in this release. See New Features.

## 8. Documentation

A version of the RXI User's Guide is available for these platforms:

- DEC (LK201 keyboard) (with an appendix for the NCD (NCD101 keyboard)) - Product Number 4000-00425

Keyboard overlays are available fore these keyboards:

- Apple (extended keyboard) - Product Number ????-???? (A/UX)
- DEC (LK201 keyboard) - Product Number 4000-00427 (both VMS and ULTRIX)
- IBM (RS/6000 or X-120 keyboard) - Product Number 4000-00479
- HP (46021A keyboard) - Product Number 4000-00484
- NCD (NCD101 keyboard) - Product Number 4000-00428
- Sun (model 101A keyboard, IBM keyboard) - Product Number 4000-00479
- Sun (model 4 keyboard, MIT X11) - Product Number 4000-00317
- Sun (model 4 keyboard, X11/NeWS) - Product Number 4000-00533

## 9. Copyrights

In addition to objects written by Rational, the RXI distribution includes some objects originally delivered with the X Window System from MIT. All objects which originated in the X Window System distribution contain the original copyright notice in their source file. This includes objects delivered with the X Window System and modified by Rational.

The following copyright notices pertain to those objects that have not been written wholly by Rational.

Copyright 1989 - 1990 by Rational, Santa Clara, California. Copyright 1988 by Wyse Technology, Inc., San Jose, California. Copyright 1987 - 1989 by Digital Equipment Corporation, Maynard, Massachusetts. Copyright 1987 - 1989 by Massachusetts Institute of Technology, Cambridge, Massachusetts.

All Rights Reserved.

Permission to use, copy, modify, and distribute this software and its documentation for any purpose and without fee is hereby granted, provided that the above copyright notice(s) appear in all copies and that both that copyright notice(s) and this permission notice appear in supporting documentation, and that the names of Digital, MIT, Wyse, or Rational not be used in advertising or publicity pertaining to distribution of the software without specific, written prior permission.

Digital, MIT, Wyse, and Rational disclaim all warranties with regard to this software, including all implied warranties of merchantability and fitness, in no event shall Digital, MIT, Wyse, or Rational be liable for any special, indirect or consequential damages or any damages whatsoever resulting from loss of use, data or profits, whether in an action of contract, negligence or other tortious action, arising out of or in connection with the use or performance of this software.

---

Installation Procedure

---

---

Release1\_0\_1

---

---

**Copyright © 1992 by Rational**

---

**PN 507-003288-003**

**This document is subject to change without notice.**

**Rational and R1000 are registered trademarks and Rational Environment is a trademark of Rational.**

**Rational, 3320 Scott Boulevard, Santa Clara, California 95054-3197**

## Table Of Contents

|   |                                                          |    |
|---|----------------------------------------------------------|----|
| 1 | Installation                                             | 1  |
|   | Overview                                                 | 1  |
|   | Assessing the Impact of the Release                      | 1  |
|   | Verifying Prerequisites to the Installation              | 2  |
|   | Supported Configurations                                 | 2  |
|   | Installing the release on the R1000                      | 2  |
|   | Installing the workstation software from the R1000       | 3  |
|   | Installing the workstation software from the workstation | 4  |
|   | Rational Access fonts                                    | 6  |
|   | Gaining visibility to Rational_Access                    | 6  |
|   | Testing the New Release                                  | 8  |
|   | Cleaning up                                              | 8  |
|   | Appendix A Procedure Do_Step                             | 11 |
|   | Appendix B Troubleshooting                               | 13 |
|   | Appendix C Install_Rational_Access                       | 15 |
|   | Appendix D Release Contents                              | 17 |
|   | Appendix E System Manager Checklist                      | 19 |
|   | Appendix F Feedback                                      | 21 |





# 1

---

## Installation

---

---

### Overview

---

Estimated time for installation: 2 hours.

This package is composed of:

- Rational\_Access Release1\_0\_1 tape
- These installation instructions
- Rational\_Access Release1\_0\_1 Release Information.

When installing the Rational\_Access release, you can view the upgrade process as a series of phases. These phases must be performed in a serial manner in the order specified.

- Assessing the impact of the release on the user community
- Verifying prerequisites to the installation
- Installing the release on the R1000
- Installing the release on the Workstation
- Cleaning up

---

### Assessing the Impact of the Release

---

In this phase of the installation, you will assess the impact of the release on the user community and the machine.

- ☐ 1. Users will not be able to use Rational\_Access during installation. During installation the Rational\_Access server will be killed and the new Rational\_Access server will be started.

- ☐ 2. Rational\_Access\_Commands in !Machine.Editor\_Data will be over written.

---

## Verifying Prerequisites to the Installation

---

In this phase of the installation, you will verify that the prerequisites for installing this upgrade have been met. Do not proceed with the upgrade until all prerequisites have been fulfilled.

- ☐ 1. You have read this entire install note.
- ☐ 2. You have read the entire Release Information.
- ☐ 3. You have at least one of the configurations listed in the supported configurations section below.

---

## Supported Configurations

---

Rational\_Access Release1\_0\_1 supports the following configurations:

- ☒ 1. A R1000 running Environment D\_12\_7\_3 (or later) and
- ☐ 2. A IBM RS/6000 running AIX 3.2. or
- ☐ 3. A Sun Sparc workstation running SunOS 4.1.2 with one of the following X Window Systems:
  - MIT X11R4 with MWM
  - MIT X11R5 with MWM
  - Sun OpenWindows 2.0 (not recommended)
  - Sun OpenWindows 3.0 (not recommended)

Rational recommends MIT X11R4 or X11R5 with the Motif Window Manager.

---

## Installing the release on the R1000

---

In this phase of the install, you will restore the tape which contains a release world located in !Machine.Release.Archive.Rational\_Access.Release1\_0\_1. This release world contains nested archives which will then be restored to complete the installation.

*The following steps utilize the procedure Do\_Step as described in Appendix Do\_Step.*

- ✓ 1. Load the Rational\_Access Release1\_0\_1 tape into the R1000's tape drive.
- ✓ 2. Log in to the Environment as user `Rational`, which must be a member of group *Privileged*.
- ✓ 3. **LOAD\_TAPE** [10 Minutes]  
This step restores the Rational\_Access Release1\_0\_1 tape. This step can be executed from any command window. Answer the mount request at the Operators Console. There should be no errors.
- ✓ 4. Traverse to `!Machine.Release.Archive.Rational_Access.Release1_0_1`.
- ✓ 5. **INSTALL\_RA** [30 minutes]  
This step will execute the steps necessary to do most of the installation on the R1000. After each step you will be asked if you want to continue the installation (an answer of "QUIT" will abort the install). Review any logs displayed by the step before continuing to the next step (an entry of "..." in the log indicates no errors). The section `INSTALL_RA` contains more information about the steps executed by this step and possible errors. The steps may be run individually if desired.

---

## Installing the workstation software from the R1000

---

Use these instructions to install Rational\_Access on workstations and file servers from the R1000. If you prefer to install the workstation software from the workstation, see the section *Installing the workstation software from the workstation*.

### Overview

In `!Machine.Release.Rational_Access.Release1_0_1` there is a procedure called `Workstation_Install` which allows you to install the Rational\_Access software on the workstation entirely from the R1000, that is, you do not have to log on to the workstation to do the install.

`Workstation_Install` requires `rsh` to be available on the workstation, if `rsh` is not available, use the instructions in the *Installing the workstation software from the workstation* section of this document.

Your session on the R1000 must be considered a trusted host on the workstation when installing the workstation portion of Rational\_Access. The R1000 is considered a trusted host when there is an entry for the R1000 and the user installing the software on the R1000 in the `.rhosts` file of the user listed in the `Remote_Username` parameter of `Workstation_Install`.

For example, if you are installing Rational\_Access on a R1000 named `my_r1000`], as the user `RATIONAL`, and you are using the username `TECHREP` for the `Remote_Username` parameter of `Workstation_Install`. The entry in the `.rhosts` file in the user `TECHREP`'s home library on the workstation would be:

```
my_r1000 rational
```

The procedure `Workstation_Install` will add the entry to the `.rhosts` file if necessary. The `.rhosts` file will be restored to its original condition when the install has completed.

*Note: If `Workstation_Install` is aborted, the `.rhosts` file may not be restored to the original version.*

`Workstation_Install` will not modify any files or create any directories on the workstation without approval from the installer.

*Note: Remember that UNIX is case-sensitive so be sure you are using the right case for usernames, etc. when installing/using Rational\_Access.*

- ☐ 1. On the R1000, traverse to `!Machine.Release.Rational_Access.Release1_0_1`
- ☐ 2. Create a command window and execute the procedure `Workstation_Install`.

Be sure that the user you supply for the `Remote_Username` parameter has write access to the directory you supply in the `Remote_Directory` parameter. The location `/vendor/rational/Rational_Access/Rev1_0_1` is used in the examples below, however, `Rational_Access` may be installed in any location on the workstation.

For example, to transfer the files to a workstation called "my\_sun", with a username "root" and a password "root\_passwd", use the following command:

*Note: When installing Rational\_Access on a server, use a directory name which is common on all machines, and make sure the appropriate filesystems are mounted read-write. i.e. `/vendor/rational`" instead of `/tmp_mnt/vendor/rational`."*

```
Workstation_Install (Remote_Machine => "my_sun",
 Remote_Username => "root",
 Remote_Password => "root_passwd",
 Remote_Directory =>
 "/vendor/rational/Rational_Access/rev1_0_1",
 Response => "<PROFILE>");
```

The procedure takes about 5 minutes. If errors are displayed, correct the problem and run `Workstation_Install` again.

---

## Installing the workstation software from the workstation

---

The Rational\_Access workstation software can also be installed by executing commands from the workstation. Use these directions if `rsh` is not available on the workstation, or if the workstation's system administrator prefers installing the Rational\_Access software from the workstation.

The lack of detail in this section is intentional. If the installer does not know enough UNIX to use these directions, use the directions in the **Installing the workstation software from the R1000** section of this document.

- ☐ 1. Create a directory for the Rational\_Access software on the workstation (/vendor/rational/Rational\_Access/rev1\_0\_1 in these examples).
- ☐ 2. Traverse to the directory created in the above step.
- ☐ 3. Use `ftp` (in binary mode) to copy
  - `!Machine.Release.Rational_Access.Release1_0_1.Ws_Files` from the R1000 to a file named `rational.z` on the workstation and
  - `!Machine.Release.Rational_Access.Release1_0_1.Install_Release` from the R1000 to a file named `install_release` on the workstation.

The procedure

`!Machine.Release.Rational_Access.Release1_0_1.Workstation_Install` can be used to create the directory and transfer the files to the workstation. If you want to complete the install manually, enter `NO` to the prompt "Perform installation of Rational Access on workstation?" from `Workstation_Install`.

The following dialog is an example of using FTP to transfer the file `Ws_Files` from the R1000 to a file named `rational.z` on the workstation. The file `Install_Release` must also be transferred to the workstation in a similar manner. This example assumes a R1000 named `my_r1000`.

```
ftp my_r1000
Connected to my_r1000.
220 Rational FTP Server version Delta
Name (my_r1000:root): operator
331 Username OK. Please enter password:
Password: -- enter password
230 User logged in, current context !USERS.OPERATOR
ftp> binary
ftp> get
(remote-file)
!Machine.Release.Rational_Access.Release1_0_1.Ws_Files
(local-file) rational.Z
200 PORT info updated
150 File status ok; connecting to 89.64.2.105, 5.35
226 Transfer is complete; Closing data connection.
local: rational.Z remote:
!Machine.Release.Rational_Access.Release1_0_1.Ws_Files
2564096 bytes received in 51 seconds (49 Kbytes/s)
ftp> quit
221 Session complete - closing connection
```

- ❑ 4. Install the workstation software using `install_release`. `install_release` takes one argument which is the location that Rational\_Access will be installed on the workstation.

```
csh install_release /vendor/rational/Rational_Access/rev1_0_1
```

Takes about 5 minutes to complete the execution of `install_release`.

---

## Rational Access fonts

---

Rational\_Access is delivered with it's own sets of fonts. These fonts come compiled for Sun workstations running MIT X11 or OpenWindows, and IBM RS6000 workstations. Font sources are included for other X Servers.

It is possible to use other fonts with Rational\_Access. Use the `-fn` and `-fb` options when starting Rational\_Access to change the font used. For more information about specifying fonts used, please refer to the users guide. Using non-Rational supplied fonts may cause things like frame borders to not be displayed correctly.

There is a program called `checkfonts` in the `bin` directory of Rational\_Access. This utility can be used to check to see if the Rational\_Access fonts are installed on the X Server. By default, `checkfonts` checks fonts for the display specified in the environment variable `DISPLAY`. The `-display` option is used to specify a different X display. For example, "`checkfonts -display rational:0`" checks for Rational\_Access font visibility on the `rational:0` display.

There is another program in the `bin` directory called `install_fonts`. Use this program to install the Rational\_Access fonts on Sun and IBM RS6000 workstations, if necessary. To install the fonts:

- Traverse to the `bin` directory of Rational\_Access.
- Type `install_fonts` and press [Return].
- Answer the prompts displayed by `install_fonts`.

If you are using a X Server other than Sun or IBM, you will have to install the fonts on that server manually. In the `fonts` directory, there is a directory called `source` that contains the source files for the Rational\_Access fonts. These source files must be copied to the X Server and be compiled and installed for that X Server. The commands used to compile and install the Rational\_Access fonts are different on different X Servers, see the documentation for your X Server for information about installing fonts on that X Server.

---

## Gaining visibility to Rational\_Access

---

Once Rational\_Access has been installed on the workstation (by `Workstation_Install` or `install_release`) steps must be taken to give users visibility to Rational\_Access.

In the Rational\_Access directory on the workstation (`/vendor/rational/Rational_Access/rev1_0_1` in the above examples) the following items are of interest:

- `bin`: This directory contains the Rational\_Access executables, the script to start Rational\_Access (named `rational`) and the font installation procedures.
- `fonts`: This directory structure contains fonts in both source and compiled forms.
- `man`: This directory contains man page files for Rational\_Access.
- `app_defaults`: This directory contains the application defaults for Rational\_Access.
- `config.csh`: This script can be used to gain visibility to Rational\_Access for C shell users or C shell equivalent users.
- `config.sh`: This script can be used to gain visibility to Rational\_Access for Bourne shell users or Bourne shell equivalent users.

At a minimum, users need visibility to the script `rational` in the `bin` directory.

Depending on how the Rational\_Access fonts were installed, users may have to take steps to gain visibility to the fonts. If the Rational\_Access fonts were installed in the default font path, users would not have to take additional action to gain visibility to the Rational\_Access fonts.

If the Rational\_Access fonts were installed in some other location, users would have to take additional steps to gain visibility to the Rational\_Access fonts.

- Use the `xset` program to add the Rational\_Access fonts to the X server's font search path. For example, if the fonts were installed in `/fonts/ra`, then try the command `xset +fp /fonts/ra`. The X server must be running for this to work. This will have to be used each time the X server is restarted.
- Put an `xset` command entry for the font directory in the user's `.xinitrc` file. (The name of this file also varies from vendor to vendor.) This file contains shell commands that are executed when the X server is started by this user.
- There may be other ways to add the Rational\_Access fonts to the X server. See the vendor's windowing system users guide for more information.

If messages in the form of "X Toolkit Warning: can't convert..." are displayed when starting Rational\_Access, the X Server can't find the Rational\_Access fonts.

The `rational` script will attempt to gain visibility to the application defaults.

Users may also want to modify their `MANPATH` environment variable to gain visibility to the Rational\_Access man pages.

A shell script called `config.csh` or `config.sh` is built by the Rational\_Access install procedure. Users can gain visibility to Rational\_Access by executing

```
source config.csh
```

or

```
. config.sh
```

from the Rational\_Access directory. It is suggested that users add the entries of the corresponding config file to their initialization procedure.

---

## Testing the New Release

---

This phase provides a minimal amount of testing. If any errors are encountered during testing, please refer to the Troubleshooting appendix for corrective actions.

- ☐ 1. Go to the workstation and execute `rational` in a shell prompt. Make sure that there are no error or warning messages.
- ☐ 2. Click on the entry **On Key Bindings** in the Help menu. A help dialog box will be brought out. Make sure that the text file does contain help in it.
- ☐ 3. Click on the entry **On Environment** in the Help menu. A dialog box will be brought out. Type `cmvc` in the **Pattern:** text field, and click on **Filter**. Make sure that there are entries shown under **Filtered Topics:**.
- ☐ 4. Connect to the R1000. And try promoting a command with the **F8** key.

---

## Cleaning up

---

In this phase, you will perform some cleanup activities.

- ☐ 1. Go to `!Machine.Release.Archive.Rational_Access.Release1_0_1.Logs` and print out the file `Do_Step_Execution_Time`. Return this printout to Rational along with other feedback from the **Feedback** appendix.
- ☐ 2. **CLEANUP** *[3 minutes]*  
After you have confirmed that the release has been successfully installed, this step will destroy



the release and archive libraries for this release of Rational\_Access. If the install must be repeated (on the R1000 or workstation(s)) after this step is run, the Rational\_Access product will have to be restored from tape. Watch the output from this step for objects that could not be destroyed, you may want to take additional steps to clean up these objects. Objects to be destroyed by this step are:

```
!Machine.Release.Archive.Rational_Access
!Machine.Release.Rational_Access
```

- 3. On the workstation, the file `rational.z` can be deleted. The install procedure installs both the executable code for Sun SPARC and IBM RS6000. Therefore, to save disk space, delete unwanted executable, namely `bin/access-sparc` or `bin/access-rs6k`. If the installation is intended for both platforms, the previous two files should not be deleted. Please note that the above files are mentioned with respect to the install directory.



# A

---

## Procedure Do\_Step

---

The majority of the steps which must be performed for this installation utilize the procedure `Do_Step` (included as part of the release tape and located in `!Commands`) to execute the required sequence of commands which implement the step. This procedure **must always** be executed from a command window in the release library

```
!Machine.Release.Archive.Rational_Access.Release1_0_1
```

Each step which utilizes procedure `Do_Step` will be of the form

```
<STEP_NAME> [<TIME TO EXECUTE>]
 <DESCRIPTION>
```

where `<STEP_NAME>` is passed as the parameter to procedure `Do_Step` which performs the necessary actions to complete the step, `<TIME TO EXECUTE>` is the amount of time the step takes to execute (this may be expressed as a range, in which case the lower values is typically the minimum, and the upper value is an estimate of the maximum), and `<DESCRIPTION>` is a description of what the step does, including any action which you will need to manually perform. For example:

**FOO**

*[5 Minutes]*

This step is an example of the format used for steps which are executed using procedure `Do_Step`. To execute step **FOO**, you would go to `!Machine.Release.Archive.Rational_Access.Release1_0_1` and in a command window, execute:

```
Do_Step ("FOO");
```

which would result in automatic execution of all commands required to implement this step. If you needed to take any manual action in addition to executing `Do_Step`, it would be noted as :

☐ Perform some manual check

If any errors occur for this step, you should always fix the problem before proceeding on to the next step. Each step is defined by a fragment of Ada code which is executed by `Program.Run`. These fragments are stored in the `Steps` file

located in the `Command_Data` library of the release. In the event you want to modify a step, you can use the "PROMPT => <STEP>" form when invoking `Do_Step`. See the spec of procedure `Do_Step` (located in `!Commands`) for more detailed information.

*Warning: If you interrupt the execution of `Do_Step` (by using `Job.Interrupt` such as `CONTROL-G`) it is possible that certain interactive commands executed by some steps, such as `Common.Definition`, may fail with an exception and display a message such as*

*Unable to read file due to `Constraint_Error (Null Access)`*

*In general, this message has no negative impact on the execution or completion of the step, and can be ignored.*

*Note: When multiple steps are executed, you will be prompted between steps with the following:*

*Continue with <Step Name> step (Continue | Skip | Quit)? [Continue] : {input}*

*Select the action to be taken, one of `Continue`, `Skip`, or `Quit`. `Continue` (the default if `PROMOTE` is entered without typing anything) results in the step <Step Name> being executed. `Skip` will skip <Step Name> and prompt with the step following <Step Name>. `Quit` results in termination of `Do_Step` without further steps being executed.*

# B

---

## Troubleshooting

---

**Error:** "Warning: translation table syntax error: Unknown keysym name: osf..." **Fix:** Append the file `$ACCESHOME/Rational_Access/XKeysymDB` to the file `/usr/lib/XKeysymDB` on the workstation. `$ACCESHOME` is the directory where Rational Access was installed.

**Error:** "X Toolkit Warning: can't convert..." when running `rational`  
The error means the X Server can not find the Rational\_Access fonts. **Fix:** If displaying on a machine other than a Sun or IBM RS6000, most probably fonts aren't installed on that machine or the font path has not been set correctly.

**Error:** "Unable to find .../access..." when running `rational`  
**Fix:** Most likely the `rational` script has been moved to a different place from where it was originally installed, so it can not find the Rational\_Access binaries. Or links have been created to point to the `rational` script to gain visibility, in which case the `rational` script gets confused as to where Rational\_Access binaries live. To solve both problems, do not create a link or move the `rational` script. Instead, create a script that calls the `rational` script with the whole pathname, or change users search path to look in the Rational\_Access bin directory.

**Error:** "No manual entry for rational" when doing `man rational`  
**Fix:** The Rational\_Access man pages directory has not been added to the users environment variable `MANPATH`. To fix this problem, add the Rational\_Access man pages directory to `MANPATH`. For example on how to set the environment variable `MANPATH`, please look at one of the config files in the Rational\_Access directory.

**Error:** "`rational: Command not found.`" when running `rational`  
**Fix:** Rational\_Access is not visible to the user's search path. To fix this problem, add the Rational\_Access bin directory to the user's search path. For example on how to change the user's search path, please look at one of the config files in the Rational\_Access directory.

**Error:** "... terminated due to unhandled exception  
!Lrm.System.Nonexistent\_Page\_Error" when executing a key binding or an entry from the menu.  
**Fix:** This unhandled exception is most probably caused by not having the Rational Access server running. To fix this problem, promote  
!Machine.Initialization.Rational.Rational\_Access\_User\_Interface. A workaround for not being able to use the key `F8`. Promote using `Ctrl-Enter`.



# C

---

## Install\_Rational\_Access

---

The step `Install_Rational_Access` runs the following steps in this order.

- ❑ 1. `RELEASE_RESTORE` *[5 minutes]*  
This step restores the archives. When completed, a filtered error log is displayed. Examine this log for errors; ignore errors about switches and links.
- ❑ 2. `RESTORE_NOTES` *[1 minute]*  
This step restores the Rational\_Access Release Notes into `!Machine.Release.Release_Notes`.  
  
*Note: Use the "RAW" option when printing the \_Ps copy of the release note.*
- ❑ 3. `INSTALL_PRODUCT` *[5 minutes + Refresh\_Terminal\_Information]*  
This step installs the subsystems that make up this release. This step will also run `Refresh_Terminal_Information` if requested to do so. Ignore the error "cannot run - needs a selection" when running `Refresh_Terminal_Information`. When completed, a filtered error log will be displayed. Examine this log for errors, there should be no errors.
- ❑ 4. `RECORD_INSTALLATION` *[1 minute]*  
This step records release information in `!Machine.Release.Current.Products` and sets the login limit back to unlimited logins. This is a required step and establishes that the release has successfully been installed.





# D

---

## Release Contents

---

This release contains the following components on the R1000:

`!Commands.Menu_Operations`

`!Machine.Editor_Data.Rational_Access_Commands`

`!Machine.Editor_Data.Rational_Access_Keys`

`!Machine.Editor_Data.Rational_Access_Key_Names`

`!Machine.Initialization.Rational.Rational_Access_User_Interface`

This release contains the following files on the workstation. The installation of Rational\_Access requires about 20 MB of disk space. Once Rational\_Access is installed, the `rational.z` file can be deleted and only 15 MB of disk space will be used.

`Rational_Access/Env_Help_Topics`

`Rational_Access/Help_Data`

`Rational_Access/XSun3-xmodmap`

`Rational_Access/Xsun4-xmodmap`

`app-defaults/RATIONAL`

`bin/access-rs6k`

`bin/access-sparc`

`bin/checkfonts`

`bin/install_fonts`

bin/rational

fonts/source

fonts/SPARC\_MIT

fonts/SPARC\_NeWS

fonts/RS6000

man/man/rational.n

Note that the actual location of the workstation components is with respect to the location at which they are installed.

# E

---

## System Manager Checklist

---

- ☐ 1. Make a full Environment backup.
- ☐ 2. Update Daily Message indicating that Rational\_Access Release1\_0\_1 has been installed.



# F

---

## Feedback

---

In order to help us improve our instructions and make installations easier to do, the following form is provided to allow you to give us feedback. Please complete and send along with a printout of the file `Do_Step_Execution_Time` located in `!Machine.Release.Archive.Rational_Access.Release1_0_1.Logs` to:

Rational  
Atten: SMSE  
3320 Scott Blvd.  
Santa Clara, CA 95054-3197  
Re: Rational\_Access

- ☐ 1. How long did the installation take you?  
On the R1000?  
  
On the Workstation?
- ☐ 2. Did the installation proceed as described in the instructions?
- ☐ 3. What could be done to improve these (or other) instructions/installations?
- ☐ 4. What did you like about this set of instructions/installation?
- ☐ 5. For the Workstation part of the installation:
  - Did the Rational Tech Rep or customer System Administrator do the install?
  - Which installation method was used? (R1000 or workstation based)
  - Which method was used to gain visibility to the product?
  - What did you like/dislike about the workstation installation?

# Rational Access Release Information

Release 1\_0\_1

Copyright © 1992 by Rational

Part Number: 508-003288-001

November 1992 (Software Release 1\_0\_1)

AIX and RISC System/6000 are trademarks and IBM is a registered trademark of International Business Machines Corporation.

OpenWindows is a trademark of Sun Microsystems, Inc.

OSF/Motif is a trademark of Open Software Foundation, Inc.

PostScript is a registered trademark of Adobe Systems Incorporated.

Rational and R1000 are registered trademarks and Rational Environment is a trademark of Rational.

Sun and SunOS are trademarks of Sun Microsystems, Inc.

SPARCstation is a trademark of SPARC International, Inc., licensed exclusively to Sun Microsystems, Inc.

UNIX is a registered trademark of UNIX System Laboratories, Inc.

VAXstation is a trademark of Digital Equipment Corporation.

X Window System is a trademark of MIT.

Rational  
3320 Scott Boulevard  
Santa Clara, California 95054-3197

# Contents

|                                                                |   |
|----------------------------------------------------------------|---|
| 1. Overview . . . . .                                          | 1 |
| 2. Components Included in this Release . . . . .               | 1 |
| 2.1. R1000 Components . . . . .                                | 1 |
| 2.2. Workstation Components . . . . .                          | 1 |
| 3. Supported Configurations . . . . .                          | 2 |
| 4. Compatibility with Other Rational Products . . . . .        | 2 |
| 5. Known Problems . . . . .                                    | 3 |
| 5.1. Creating and Activating Buttons . . . . .                 | 3 |
| 5.2. Resizing the Environment Area . . . . .                   | 3 |
| 5.3. Promoting Withdrawn Units . . . . .                       | 4 |
| 5.4. Incremental Operations . . . . .                          | 4 |
| 5.5. Using Telnet Within the Access Window . . . . .           | 4 |
| 6. Limitations . . . . .                                       | 4 |
| 6.1. Key Combinations Reserved by Other Applications . . . . . | 4 |
| 6.2. Key Combinations Reserved by Access . . . . .             | 4 |
| 6.3. Limitation on User-Defined Buttons . . . . .              | 5 |
| 6.4. Customization . . . . .                                   | 5 |
| 7. Documentation . . . . .                                     | 5 |
| 7.1. Printed Documentation . . . . .                           | 5 |
| 7.1.1. Corrections . . . . .                                   | 6 |
| 7.1.2. Clarifications and Additions . . . . .                  | 6 |
| 7.2. Online Documentation . . . . .                            | 6 |
| 8. Training . . . . .                                          | 6 |
| A Compatibility with OpenWindows . . . . .                     | 7 |





# 1. Overview

This release information describes Release 1\_0\_1 of Rational Access, an OSF/Motif™-based graphical user interface for the Rational Environment.™ Access runs as an X Window System™ (X) application on a UNIX® workstation. It features:

- The OSF/Motif mouse, menu, and dialog box paradigm.
- Many new “full service” commands, which combine several lower-functionality Environment operations.
- A comprehensive, menu-based online help facility.
- Platform-independent key and mouse bindings.
- Full support for existing Environment features such as “item-operation” combinations and command windows.

For more information about Access, refer to accompanying *Rational Access User's Guide*.

*Note: This release information can be found online in the !Machine.Release.Release\_Notes world in line-printer (Access\_Release1\_0\_1\_Lpt) and PostScript® (Access\_Release1\_0\_1\_Ps) formats.*

## 2. Components Included in this Release

### 2.1. R1000 Components

The following Access components are installed on the R1000® as part of this release:

- !Commands.Mail package
- !Commands.Menu\_Operations package
- !Machine.Editor\_Data.Rational\_Access\_Commands procedure
- !Machine.Editor\_Data.Rational\_Access\_Key\_Names package
- !Machine.Editor\_Data.Rational\_Access\_Keys file
- !Machine.Initialization.Rational.Rational\_Access\_User\_Interface loaded main program

### 2.2. Workstation Components

The following Access components are installed on the workstation as part of this release:

- Rational\_Access/Env\_Help\_Topics file
- Rational\_Access/Help\_Data file
- Rational\_Access/XSun3-xmodmap file
- Rational\_Access/Xsun4-xmodmap file
- Rational\_Access/XKeysymDB file
- app-defaults/RATIONAL file
- bin/access-rs6k file

- bin/access-sparc file
- bin/checkfonts file
- bin/install\_fonts file
- bin/rational file
- fonts/source directory
- fonts/SPARC\_MIT directory
- fonts/SPARC\_NeWS directory
- fonts/RS6000 directory
- man/man/rational.n file

Note that the actual location of the workstation components is with respect to the location at which they are installed. For more information about installation, refer to the installation instructions delivered with this release.

### 3. Supported Configurations

Access has been developed and tested on the following workstation configurations. This release of Access may work with other configurations; however, Rational cannot support or guarantee other configurations.

*Table 1 Supported Configurations*

| Rational Environment | Workstation                                        | Operating System          | Window System                                | Keyboard                  |
|----------------------|----------------------------------------------------|---------------------------|----------------------------------------------|---------------------------|
| D_12_7_3             | Sun <sup>TM</sup><br>SPARCStation <sup>TM</sup>    | SunOS <sup>TM</sup> 4.1.2 | MIT X11R4 and<br>the Motif Window<br>Manager | Sun Type 4 or<br>U.S. 101 |
| D_12_7_3             | IBM <sup>®</sup> RISC<br>System/6000 <sup>TM</sup> | AIX <sup>TM</sup> 3.2     | MIT X11R4 and<br>the Motif Window<br>Manager | IBM U.S.                  |

Although not recommended or supported, Access will run under OpenWindows<sup>TM</sup>. For information, see Appendix A, "Compatibility with OpenWindows."

### 4. Compatibility with Other Rational Products

This 1\_0\_1 release of Access supports the Rational layered-software products listed in Table 2.

**Table 2 Compatibility with Layered Products**

| <b>Rational Product</b>       | <b>Compatible Release</b> |
|-------------------------------|---------------------------|
| Insight                       | 1_3_0                     |
| Design Facility: 2167A        | 6_2_5 or later            |
| Rational Publishing Interface | 1_0_2 or later            |
| Rational Teamwork Interface   | 2_1_2 or later            |

Supported layered products can be started, used, and exited using Access menu commands. Commands for supported layered products appear in the Tools menu. If you try to use a layered product that is not installed or authorized, a message appears in the Environment message window. For more information about using layered products with Access, see the online help for specific menu commands.

Layered products that are not supported by Access can still be used from command windows within an Access window.

## 5. Known Problems

The following problems are specific to this 1\_0\_1 release of Access and were known at the time of release.

### 5.1. Creating and Activating Buttons

When the Access window is the maximum height allowed on the display, adding new user-defined buttons does not resize the button panel. Thus, new buttons may be hidden until you pull down the sash.

Sometimes a button remains selected (colored in) after it has been activated. This behavior is most likely to occur when you activate window-control or user-defined buttons from the keyboard. It has also been noticed in persistent dialog boxes such as the Image Palette, Function Key Palette, Debugger Palette, and Help:On Environment box. The button is displayed correctly the next time it is drawn. To force Access to redraw the button, cover and uncover the window containing the button.

### 5.2. Resizing the Environment Area

You cannot use the keyboard to move the sash in the main Access window.

Resizing the Environment area, either by moving the sash or resizing the entire Access window, sometimes results in a misplaced Environment cursor.

### 5.3. Promoting Withdrawn Units

Program:Promote to Source attempts to promote withdrawn units to the installed state, without bringing up a dialog box or prompting for confirmation. This problem is caused by an underlying Environment procedure and cannot be fixed in Access.

### 5.4. Incremental Operations

If the cursor is in an Ada unit and nothing is selected, the Program:Incremental:Incremental Edit command demotes the entire unit to the source state. This behavior occurs because Incremental Edit directly calls the Environment command !Commands.Common.Edit, which is intended for context-sensitive use, based on the presence (or lack of) a selection.

### 5.5. Using Telnet Within the Access Window

A single Access window should be used for each connection to the R1000. Attempting to make several telnet connections between a single Access window and multiple Environment sessions may not work correctly. If you encounter problems, remember that:

- Session:Screen:Rational Mode should be set for Environment operations. It should not be set for UNIX or telnet operations.
- Environment key bindings are session-specific, not user-specific. Thus, if you are connected to two Environment sessions through a single Access window, your key bindings will change when you switch sessions.
- [Meta][L] clears the Environment area of the Access window.
- [Control][L] repaints the Environment area of the Access window.

## 6. Limitations

The following limitations apply to this 1\_0\_1 release of Rational Access.

### 6.1. Key Combinations Reserved by Other Applications

When there is a conflict between key combinations that are reserved by your window manager or display and Access key bindings, the Access bindings are ineffective. For example, regardless of any Access bindings:

- From a Sun SPARCStation running the Motif Window Manager, [Control][Meta][1] toggles the screen.
- From an IBM RS/6000, [Control][Meta][Delete] and [Control][Meta][Back Space] kill the window manager.
- From a VAXstation,<sup>TM</sup> [Control][F2] reboots the system.

### 6.2. Key Combinations Reserved by Access

You should not attempt to rebind the following key combinations:

[Meta][B]  
[Meta][H]  
[Meta][O]

[Meta][E]  
[Meta][I]  
[Meta][P]

[Meta][F]  
[Meta][M]  
[Meta][S]

These key combinations are reserved for displaying Access menus. Attempts to rebind these keys may result in unpredictable behavior.

### 6.3. Limitation on User-Defined Buttons

Some Access menu items are executed on the workstation without interacting with the server on the R1000. You cannot create buttons for them. These menu items are:

|                                  |                                 |
|----------------------------------|---------------------------------|
| File:Exit                        | Debug:Debugger Commands Palette |
| Session:Screen:Save Button Panel | Session:Screen:Full Reset       |
| Session:Screen:Rational Mode     | Session:Screen:Inverse Video    |
| Session:Screen:Visual Bell       | Help:On Help                    |
| Help:On Getting Started          | Help:On Key Bindings            |
| Help:On Function Keys            | Help:On Mouse                   |
| Help:On Window Panel             | Help:On Menu:File Menu Help     |
| Help:On Menu>Edit Menu Help      | Help:On Menu:Navigate Menu Help |
| Help:On Menu:Program Menu Help   | Help:On Menu:CMVC Menu Help     |
| Help:On Menu:Debug Menu Help     | Help:On Menu:Session Menu Help  |
| Help:On Menu:Tools Menu Help     | Help:On Menu:Help Menu Help     |
| Help:On Environment              | Help:On Version                 |

### 6.4. Customization

In this 1\_0\_1 release of Access, it is not possible to change the items listed on the Access menus. It is possible, however, to change the commands that are executed by the items (through customized key binding procedures). Note, however, that such customization of Access menus has not been thoroughly tested and, therefore, is not documented or supported by Rational.

In this release of Access, it is also possible to change user-defined buttons by editing your .Rational-Access-buttons file directly. Note, again, that such customization has not been thoroughly tested and, therefore, is not documented or supported by Rational.

## 7. Documentation

### 7.1. Printed Documentation

Rational Access has been documented in the *Rational Access User's Guide*, product number 4000-00722, dated November 1992.

The following subsections contain corrections, clarifications, and additions to the information presented in the *Rational Access User's Guide*.

### 7.1.1. Corrections

Some of the screens shown in the *Rational Access User's Guide* differ slightly from those on your display. In particular, you may notice differences in fonts, the sizes of buttons, and the capitalization of labels. Functionality, however, has not been changed.

On page 166 of the *Rational Access User's Guide*, the table entry for [Line]-[Delete] states that [Control][F1]-[Delete] deletes to the end of the line. This is incorrect. [Control][F1]-[Delete] deletes to the beginning of the line.

### 7.1.2. Clarifications and Additions

If you using an IBM RS/6000 with an IBM U.S. keyboard, note that key combinations that include arrow keys refer to the arrow keys between the alphanumeric keys and the numeric keypad. The arrow keys on the numeric keypad do not perform the same operations. Specifically, pressing an arrow key on the numeric keypad executes the Access operation bound to [Shift] plus the arrow key.

Some text entry boxes, such as the Comments entry box in the CMVC dialog boxes, accept the [Tab] and [Return] characters as part of the entry. To use the keyboard to move the focus from such an entry box, press [Control][Tab] or [Shift][Tab]. To activate the default command button (usually OK) from the keyboard, first move the focus to another field in the dialog box; then the [Return] key will work as expected.

The Image Palette cannot be used to redisplay objects or images that have been abandoned unless the Image Palette lists the full pathname of those objects. In particular, the Image Palette cannot find mailboxes, mail messages, the window directory, or I/O windows that have been abandoned. (Images are abandoned using File:Close, [Control][F1]-[G], or [Control][F1]-[X]. The Remove Window button and [Control][F1]-[D] do not abandon the object. Objects that have been removed using either of those methods can be retrieved from the Image Palette without problems.)

## 7.2. Online Documentation

Access provides several kinds of online documentation:

- A complete **man** page for the **rational** command explains the options and X defaults that can be used to tailor the Access window.
- A comprehensive, menu-based online help system provides information about Access menus, key and mouse bindings, buttons, and special features.

The Access online help is integrated with the Environment help through the Help:On Environment facility.

For information about the online help facility, choose Help:On Help from the main Access window.

## 8. Training

A new version of the *Rational Environment Training: Fundamentals* course is being developed. The new version will be tailored to suit the needs to Rational Access users. It will be available in the near future.

## Appendix A

### Compatibility with OpenWindows

The recommended configuration for Access is the MIT X11 X server and the Motif Window Manager. Access will run, however, under OpenWindows. Functionality is complete, but robustness is not guaranteed and the product look is much different.

When attempting to use Access with OpenWindows, make sure that the /usr/lib/X11/XKeysymDB file on the workstation contains the following entries:

```

!

! OSF Keysyms

!

osfBackSpace :1004FF08

osfInsert :1004FF63

osfDelete :1004FFFF

osfCopy :1004FF02

osfCut :1004FF03

osfPaste :1004FF04

osfAddMode :1004FF31

osfPrimaryPaste :1004FF32

osfQuickPaste :1004FF33

osfPageUp :1004FF41

osfPageDown :1004FF42

osfEndLine :1004FF57

osfBeginLine :1004FF58

osfActivate :1004FF44

osfMenuBar :1004FF45

osfClear :1004FF0B

osfCancel :1004FF69

osfHelp :1004FF6A

osfMenu :1004FF67

osfSelect :1004FF60

osfUndo :1004FF65

osfLeft :1004FF51

osfUp :1004FF52

osfRight :1004FF53

osfDown :1004FF54

```

An XKeysymDB file with these entries is distributed with Access.



---

Installation Procedure

---

---

Release10\_1\_1

---

---

**Copyright © 1991 by Rational**

---

**PN 507-003275-002**

**This document is subject to change without notice.**

**Rational and R1000 are registered trademarks and Rational Environment is a trademark of Rational.**

**Rational, 3320 Scott Boulevard, Santa Clara, California 95054-3197**

## Table Of Contents

|                                     |    |
|-------------------------------------|----|
| 1 Installation                      | 1  |
| Overview                            | 1  |
| Assessing Impact                    | 2  |
| Verifying Prerequisites             | 2  |
| Terminal Types                      | 3  |
| Installing The Release On The R1000 | 3  |
| Installing The Release On The PC    | 3  |
| Testing RWI                         | 5  |
| Cleaning Up                         | 5  |
| Appendix A Procedure Do_Step        | 7  |
| Appendix B Install_RWI              | 9  |
| Appendix C Customer Checklist       | 11 |
| Appendix D Feedback                 | 13 |



# 1

---

## Installation

---

---

### Overview

---

Estimated time for installation: 1 hour.

This package is composed of:

- ☐ 1. RWI Release10\_1\_1 tape for R1000
- ☐ 2. RWI Release10\_1\_1 diskette for the PC with the following files:
  - a. InstRwi.exe
  - b. RwiSer.exe
  - c. RwiNet.exe
  - d. Rwi.mnu
  - e. PcBasic.mnu
  - f. Fontres.fon
- ☐ 3. These installation instructions
- ☐ 4. RWI release note

When installing the RWI release, you can view the upgrade process as a series of phases. These phases must be performed in a serial manner.

- ☐ 1. Assessing the impact of the release on the user community
- ☐ 2. Verifying prerequisites to the installation
- ☐ 3. Verifying the terminal types supported

- ☐ 4. Installing the release on the R1000
- ☐ 5. Installing the release on the PC
- ☐ 6. Testing the new release
- ☐ 7. Cleaning up

The following sections take you through the installation process a phase at a time. Be sure to complete ALL of the phases of installation and to perform these phases in the order in which they appear here.

**Warning:** *Follow these installation steps in a serial manner. Failure to adhere to these instructions may result in the failure of portions of the installation.*

---

## Assessing Impact

---

In this phase of the install, you will assess the impact of the release on the user community and the machine.

Keymaps are updated; therefore, previous keymaps are unrecoverable unless backups of the keymaps have been done before the installation.

---

## Verifying Prerequisites

---

In this phase of the installation, you will verify that the prerequisites for installing this upgrade have been met. Do not proceed with the upgrade until all prerequisites have been fulfilled.

- ☐ 1. You have read the release note.
- ☐ 2. You have read this entire install note.
- ☒ 3. The R1000 is executing Environment release D\_12\_1\_1 or later.
- ☐ 4. The PC is running MS-DOS or PC-DOS version 3.1 or later.
- ☐ 5. On PC's that will not use serial communication, networking must be installed and running on the PC. This release supports Novell Lan Workplace Rev4.0 with Microsoft Windows 3.0.
- ☐ 6. For network users the PC must be running Microsoft Windows 3.0, and Windows must be set up to use the network.

- ☐ 7. For serial users the PC must be running Microsoft Windows 3.0, must have a serial port or a modem depending on the remoteness of the machines.

---

## Terminal Types

---

The following terminal type configurations are supported in this release.

- `Pc101_Ms_Windows_3_0` - 101/102 key keyboard. Numeric keypad on right, arrow and other keys between alpha keys and numeric keypad.
- `Pc91_Ms_Windows_3_0` - 91 key keyboard. Numeric keypad on right, no keys between alpha keys and numeric keypad.
- `Pc86_Ms_Windows_3_0` - 86 key keyboard. No numeric keypad on right.
- `PcBasic_Ms_Windows_3_0` - Compatible with the above three keyboards. Designed for training purposes.

---

## Installing The Release On The R1000

---

In this phase of the release, you will restore the tape which contains a release world located in `!Machine.Release.Archive.RWI.Release10_1_1`. This release world contains nested archives which will then be restored to complete the installation.

- ✓ 1. Load the RWI Release10\_1\_1 tape onto the tape drive.
- ✓ 2. Log in as user `Rational`, which must be a member of group *Privileged*. Do NOT use another account due to problems which might arise with product token usage.
- ✓ 3. `LOAD_TAPE` *[3 minutes]*  
 This step restores RWI Release10\_1\_1 tape. This step can be executed from any command window. There should be no errors. If `Do_Step` is not available, use `Archive.Restore (Options => "Promote, Replace");`
- ✓ 4. Go to `!Machine.Release.Archive.RWI.Release10_1_1`.
- ✓ 5. `INSTALL_RWI` *[45 Minutes]*  
 This step will execute the steps necessary to do most of the installation on the R1000. The appendix `Install_RWI` contains more detailed information about the steps executed by this step and possible errors. This step may be run individually if desired.

---

## Installing The Release On The PC

---

The software for the PC is delivered on a diskette. The following examples assume the diskette will be read using drive A.

Windows must be running in order to install RWI.

- ☐ 1. Insert the RWI diskette in drive A.
- ☐ 2. Execute "a:instrwi.exe" from Windows.
  - Pull down the **F**iles menu and select the **R**un command. The Run dialog box opens.
  - Enter **a:instrwi** in the **C**ommand **L**ine edit box.
  - Press [Enter].

Follow the instructions, and if you are not sure on any of the choices, use the default value provided. The default values will install RWI in the following configuration. These settings may not be correct for your PC/Modem. Verify they are correct for your application and change them as necessary.

Data bits: 8  
Stop bits: 1  
Parity bit: None  
Flow Control: None

- ☐ 3. (optional) Add the RWI icon to a Program Manager group.
  - Select the Program Manager group in which you want to place the RWI icon by moving the mouse pointer over the group and double-clicking the left mouse button.
  - Pull down the **F**iles menu and select the **N**ew command.
  - Single-click on the **P**rogram **I**tem radio button if it is not already selected.
  - Press [Enter]. The **P**rogram **I**tem **P**roperties dialog box opens.
  - Enter **RWI** in the **D**escription edit box.
  - Enter **RWI** in the **C**ommand **L**ine edit box.



- Press [Enter]. The Program Manager adds RWI to the group.

You can now start RWI one of two ways; by using the Run command or Application-Group Icon from the Program Manager.

- Run command of Program Manager:

- Pull down the **Files** menu and select the **Run** command. The **Run** dialog box opens.
- Enter **RWI** in the **Command Line** edit box.
- Press [Enter].

- Application-Group Icon in a Program Manager group:

- Place the mouse pointer on the RWI icon.
- Double-click the left mouse button.

See the Getting Started section in the RWI User's Guide for more information on setting-up and running RWI.

---

## Testing RWI

---

After the installation is complete, log-on to the R1000 from the PC and try some function keys and menus to verify RWI is working correctly.

---

## Cleaning Up

---

In this phase, you will perform some cleanup activities. The time required for this phase will vary widely, depending on how much cleanup is required to be done.

1. Go to **!Machine.Release.Archive.RWI.Release10\_1\_1.Logs** and print out the file **Do\_Step\_Execution\_Time**. Return this printout to SMSE along with other feedback supplied on the last page of this note. Note that this step must be executed prior to performing the following step which will destroy the **Do\_Step\_Execution\_Time** file.

2. **CLEANUP**

This step destroys the release and archive libraries. Only execute this step when you are confident that the release has been properly installed.

3. Have customer take a full Environment backup.



# A

---

## Procedure Do\_Step

---

These phases must be performed in a serial manner, as must the steps in each of the phases. This installation note is organized by phase, with each phase comprising a section of this document. Once you have performed this installation, please fill out and return the feedback form that is the last page of these instructions along with a printout of the `Do_Step_Execution_Time` found in the `Logs` library of the release archive. Your feedback is very important in helping us to improve our installation instructions.

The majority of the steps which must be performed for this installation utilize the procedure `Do_Step` (included as part of the release tape and located in `!Commands`) to execute the required sequence of commands which implement the step. This procedure **must always** be executed from a command window in the release library

```
!Machine.Release.Archive.RWI.Release10_1_1
```

Each step which utilizes procedure `Do_Step` will be of the form

```
<STEP_NAME> [<TIME TO EXECUTE>]
 <DESCRIPTION>
```

where `<STEP_NAME>` is passed as the parameter to procedure `Do_Step` which performs the necessary actions to complete the step, `<TIME TO EXECUTE>` is the amount of time the step takes to execute (this may be expressed as a range, in which case the lower values is typically the minimum, and the upper value is an estimate of the maximum), and `<DESCRIPTION>` is a description of what the step does, including any action which you will need to manually perform. For example:

**FOO**

*[5 Minutes]*

This step is an example of the format used for steps which are executed using procedure `Do_Step`. To execute step `FOO`, you would go to `!Machine.Release.Archive.RWI.Release10_1_1` and in a command window, execute:

```
Do_Step ("FOO");
```

which would result in automatic execution of all commands required to implement this step. If you needed to take any manual action in addition to executing `Do_Step`, it would be noted as :

□ Perform some manual check

If any errors occur for this step, you should always fix the problem before proceeding on to the next step. Each step is defined by a fragment of Ada code which is executed by `Program.Run`. These fragments are stored in the `Steps` file located in the `Command_Data` library of the release. In the event you want to modify a step, you can use the "PROMPT => <STEP>" form when invoking `Do_Step`. See the spec of procedure `Do_Step` (located in `!Commands`) for more detailed information.

*Warning: If you interrupt the execution of `Do_Step` (by using `Job.Interrupt` such as `CONTROL-G`) it is possible that certain interactive commands executed by some steps, such as `Common.Definition`, may fail with an exception and display a message such as*

*Unable to read file due to Constraint\_Error (Null Access)*

*In general, this message has no negative impact on the execution or completion of the step, and can be ignored.*

# B

---

## Install\_RWI

---

The step `INSTALL_RWI` runs the following steps in this order.

□ 1. `RELEASE_RESTORE`

*[10 Minutes]*

This step restores the `Release` archive. When completed, a filtered error log is displayed (`Restore_Release_Log_Summary` located in `!Machine.Release.Archive.RWI.Release10_1_1.Logs`) Examine this log for errors. Ignore the following types of errors (if present):

```
!!! can't restore link...

+++ Can't resolve default
 switch file ...
```

□ 2. `INSTALL_KEYMAPS`

*[15 minutes]*

This step installs all the keymaps for PcBasic, 101/102, 91, and 86 key keyboards, and restores their respective keymap overlays. When completed, a filter error log will be displayed. Check this log for errors, there should be no errors.

□ 3. `RESTORE_NOTES`

*[1 minute]*

This step restores RWI Release Notes into `!Machine.Release.Release_Notes`.

□ 4. `RECORD_INSTALLATION`

This step records release information in `!Machine.Release.Current.Products` and sets the login limit back to unlimited logins. This is a required step and establishes that the release has successfully been installed.

□ 5. The RWI Keymaps must be enabled before they can be used.

■ Create a Command window and type:

```
Refresh_Terminal_Information
```

and press the `[PROMOTE]` key to enable the keymaps.



# C

---

## Customer Checklist

---

- ☐ 1. Each user should merge the contents of the default activity into his or her own private activity file with the command **Activity.Merge**.

```
Activity.Merge (Source => "!Machine.Release.Current.Activity",
 Mode => Activity.Differential,
 Target => ">>User Activity<<");
```

Note that this command may have the undesirable effect of modifying spec/load view settings which were explicitly set in the user activity.

- ☐ 2. Distribute the release note to user community. A postscript and lineprinter copy of the release note can be found in **!Machine.Release.Release\_Notes**.
- ☐ 3. Update the Daily Message indicating that RWI Release10\_1\_1 has been installed.
- ☐ 4. Set the ACL's on new specs/files restored as part of this release, if desired.
- ☐ 5. Make a Full backup of the Environment.





# D

---

## Feedback

---

---

In order to help us improve our instructions and make installations easier to do, the following form is provided to allow you to give us feedback. Please complete and send along with a printout of the file `Do_Step_Execution_Time` located in `!Machine.Release.Archive.RWI.Release10_1_1.Logs` to:

Rational  
Atten: SMSE  
3320 Scott Blvd.  
Santa Clara, CA 95054-3197  
Re: RWI

- ☐ 1. How long did the installation take you?
- ☐ 2. Did the installation proceed as described in the instructions?
- ☐ 3. What could be done to improve these (or other) instructions/installations?
- ☐ 4. What did you like about this set of instructions/installation?

# **Rational Windows Interface Release Information**

**Release 10\_1\_1**

Copyright © 1991 by Rational

Part Number: 508-003275-003

December 1991 (Software Release 10\_1\_1)

Rational and R1000 are registered trademarks and Rational Environment is a trademark of Rational.

Microsoft and MS-DOS are registered trademarks and Windows is a trademark of Microsoft Corporation.

Novell is a registered trademark and LAN WorkPlace is a trademark of Novell, Inc.

IBM is a registered trademark of International Business Machines Corporation.

PostScript is a registered trademark of Adobe Systems Incorporated.

Rational  
3320 Scott Boulevard  
Santa Clara, California 95054-3197

## 1. Overview

The Rational Windows Interface (RWI) provides access to the Rational Environment™ from IBM®-compatible personal computers running Microsoft® Windows™ 3.0 and connected to R1000® systems through TCP/IP network or RS232 serial connections, including modems. The RWI application is a terminal emulator.

This is the first production release.

See the “Configurations” section, below, for a complete listing of supported hardware configurations.

Installation instructions for RWI are found in the “Installation Procedure for RWI, Release 10\_1\_1.”

## 2. Configurations

Release 10\_1\_1 is compatible with Rational Environment versions D\_12\_1\_1 or later.

### 2.1. PC Hardware Requirements

- PC/AT-compatible or better
- 640K of conventional memory, minimum for Microsoft Windows Real mode; additional memory required for Standard and Enhanced Windows modes
- A monitor that is compatible with Microsoft Windows 3.0
- A keyboard with 12 function keys (note that custom changes can be made by the user that may allow other keyboards)
- Approximately 256K disk space available, local or server
- A mouse that is compatible with Microsoft Windows is recommended
- One of the following communications hardware configurations:
  - RS232 serial port or modem
  - Network adaptor that is supported by an ODI (ODLI) driver, such as those from Novell, 3Com, or IBM token ring support

### 2.2. PC Software Requirements

- Microsoft Windows Graphical Environment, version 3.0, operating in 386 Enhanced, Standard, or Real mode
- MS-DOS® or PC-DOS, version 3.1 or later
- If using a network connection, Novell® LAN WorkPlace™ 4.0 for DOS
- If you want to run a network operating system, it must be compatible with the ODI support required by LWP4

### 2.3. Installation

RWI is installed according to the “Installation Procedure for RWI, Release 10\_1\_1.” The installation requires that Windows be installed and running on the PC. If a network connection

is used, the network software should already be correctly installed.

If you are using RS232 serial communications with either a direct serial connection or a modem, and no network, then select a serial installation during the installation procedure. The installation procedure will install an executable file named `RWISER.EXE`.

If you are using a network, or a network and serial ports, then select the network installation during the installation procedure. The installation procedure will install an executable file named `RWINET.EXE`, which includes code for serial ports so that `RWISER.EXE` is not needed.

## 2.4. Executables

There are two RWI executables: one for basic serial connections, and one that adds network capability. The executable for basic serial connections is named `RWISER.EXE`, and the executable that adds network capability is `RWINET.EXE`. Note both of the following:

- The network version, `RWINET.EXE`, requires that the network software be installed.
- The network version includes code for serial connections.

See the "Installation Procedure for RWI" for instructions on installing the executable that is required for your installation.

## 2.5. Terminal Types or Keyboard Variants

The different types of PC keyboards supported by RWI are identified by a terminal type, or keyboard variant. RWI currently supports four of these keyboard variants:

- The PC101 keyboard variant provides a standard set of Environment key bindings for the enhanced keyboard with 101 keys.
- The PC91 keyboard variant provides a standard set of Environment key bindings for laptops that have twelve function keys and a separate numeric keypad.
- The PC86 keyboard variant provides a standard set of Environment key bindings for laptops that have twelve function keys and no numeric keypad.
- The PCBASIC keyboard variant provides a basic set of Environment key bindings that is much smaller than the standard set. The PCBASIC keyboard variant is designed for training purposes, and it works with all three of the above keyboards.

These keyboard variants are selected during installation and may be changed later by the user. The R1000 needs to be informed of the chosen keyboard variant. This can be done automatically through a setting in the RWI Terminal Information dialog box or manually in response to the Environment's Enter Terminal Type prompt. For more information, see the "Installation Procedure for RWI" and the *RWI User's Guide*.

## 2.6. Pull-Down Menus

User-defined pull-down menus are available in RWI. These pull-down menus are defined in text files stored on the PC. The initial RWI profile is installed with one of two sample pull-down menus, depending on which keyboard variant is chosen:

- A menu that is compatible with the standard set of Environment key bindings (PC101, PC91, PC86)
- A menu that is compatible with the basic set of Environment key bindings (PCBASIC)

Note that both sample pull-down menus are installed on the PC hard disk, so that the user can incorporate either menu in subsequent user-created custom profiles.

### 3. Compatibility

First production release of RWI.

Compatible with Environment release D\_12\_1\_1 or later.

### 4. Upgrade Impact

None.

### 5. Known Problems

#### 5.1. Flow Control and Pasting from the ClipBoard

When a serial connection is used, the flow control should be set properly to avoid problems when pasting large buffers of data from the Windows Clipboard into RWI.

Pasting large data buffers is slow, and a file-transfer utility may be more appropriate. One such utility is LAN WorkPlace 4.0 File Express, available to network users.

#### 5.2. Environment Window Style and Copy to Clipboard

If the Window\_Have\_Sides session switch is set to True, then Copy to Clipboard includes border characters with the text sent to the Windows Clipboard. Set the Window\_Have\_Sides switch to False to correct this problem.

#### 5.3. Incorrect RWI Window Size

If the RWI window itself is not contained within the boundaries of the display monitor screen, reduce the lines or columns or both through the Terminal Information dialog box:

- Pull down the RWI Control menu and select the Terminal command. The Terminal Information dialog box opens.
- Enter a smaller value in the Lines or Columns edit box.
- Single-click on the Save button with the left mouse button. The RWI window size reduces. Repeat this procedure until you are satisfied with the window dimensions.

If, after logging into the Rational Environment, you find that the Environment windows are not properly contained in the RWI window, try resizing the RWI window slightly with the mouse.

If the Resize Host Window check box in the Terminal Information dialog box is selected, then resizing the RWI window with the mouse will inform the Environment of the actual RWI window size, and the Environment will adjust its windows to fit the RWI window. This problem usually results from logging in with a nonstandard window size (the standard window size is 66 lines by 80 columns) without specifying the nonstandard window dimensions.

To specify nonstandard window dimensions when logging in, do one of the following:

- Select the RWI profile's Send Terminal Type check box (Telnet connections only)
- Edit the Initial Connection String to include the desired window dimensions as part of the Terminal Type entry
- Enter the window dimensions manually along with the terminal type (see the *RWI User's Guide*)

## 5.4. Cursor is not in a Window

If the cursor is not within an Environment frame when the RWI window is resized, the Environment window will be cleared and the message "cursor is not in a window" will be displayed. To fix this, reposition the cursor in an Environment frame, and resize the window.

## 6. New Features

First release.

## 7. Changes

First release.

## 8. Documentation

### 8.1. User's Guide

A user's guide is available for RWI:

- *Rational Windows Interface (RWI) User's Guide*, product number 3000-00613

Note the following changes to the *RWI User's Guide*:

- It is now possible to select a 19.2K baud rate from the Communications Setup dialog box (see Chapter 4).
- Two default menu files are supplied with RWI: PCBASIC.MNU and RWI.MNU.

### 8.2. Keyboard Overlays

A plastic keyboard overlay with replaceable paper inserts is available for:

- The 101-key Enhanced keyboard using the PC101 key bindings, product number 3000-00614

Paper keyboard overlays are provided with this release information for:

- The 101-key Enhanced keyboard
- The 91-key laptop keyboard
- The 86-key laptop keyboard

The paper overlays are provided online in !Machine.Editor\_Data.Keyboard\_Overlays as PostScript® files that can be printed on PostScript printers:

- Keymap\_Pc101\_User\_Defs\_Ps
- Keymap\_Pc86\_User\_Defs\_Ps
- Keymap\_Pc91\_User\_Defs\_Ps
- Keymap\_Pcbasic\_User\_Defs\_Ps

The source for these files, written for the Rational Document Formatter (Compose), is also provided online in the following files:

- Keymap\_Pc101\_User\_Defs
- Keymap\_Pc86\_User\_Defs
- Keymap\_Pc91\_User\_Defs
- Keymap\_Pcbasic\_User\_Defs
- Keymap\_Template\_User\_Defs

The source includes user-customizable templates with instructions. Note that the PCBASIC key bindings and overlay are applicable to all three keyboard types.



# Training

---

Installation Procedure

---

---

Release1\_1\_1

---

RATIONAL

---

**Copyright © 1992 by Rational**

---

**PN 507-003265-003**

**This document is subject to change without notice.**

**Rational and R1000 are registered trademarks and Rational Environment is a trademark of Rational.**

**Rational, 3320 Scott Boulevard, Santa Clara, California 95054-3197**

## Table Of Contents

|                              |   |
|------------------------------|---|
| 1 Installation               | 1 |
| Overview                     | 1 |
| VERIFYING PREREQUISITES      | 1 |
| INSTALLING THE RELEASE       | 2 |
| Appendix A Procedure Do_Step | 5 |
| Appendix B Feedback          | 7 |



# 1

---

## Installation

---

---

### Overview

---

When installing the Training release, you can view the process as a series of phases. These phases must be performed in a serial manner.

- Verifying prerequisites to the installation
- Installing the release

These phases must be performed in a serial manner, as must the steps in each of the phases. This installation note is organized by phase, with each phase comprising a section of this document. Once you have performed this installation, please fill out and return the feedback form that is the last page of these instructions along with a printout of the **Do\_Step\_Execution\_Time** found in the **Logs** library of the release archive. Your feedback is very important in helping us to improve our installation instructions.

---

### VERIFYING PREREQUISITES

---

In this phase of the installation, you will verify that the prerequisites for installing this upgrade have been met. Do not proceed with the upgrade until all prerequisites have been fulfilled.

- ☐ 1. You have read the release note.
- ☐ 2. You have read this entire install note.
- ☐ 3. The machine is executing Environment release D\_12\_1\_1 or later.
- ☐ 4. The machine is running the necessary product(s) to support the functioning of the desired training course.
- ☐ 5. If desired, request the **Training\_Authorization** code for use in inhibiting product token checking, allowing the maximum number of students to be trained without restriction to the number of tokens purchased by the customer. This must be done in advance of the training date.

Note that this installation allows selective restoration of Rational training courses.

---

## INSTALLING THE RELEASE

---

In this phase of the release, you will restore the tools tape which contains a release world located in `!Machine.Release.Archive.Training.Release1_1_1`. (*Note: Most machines will have this release archived located in `!Machine.Release.Archive.Training.Release1_1_1`.*) This release world contains nested archives which will then be restored to complete the installation.

- ☐ 1. Load the Training Release1\_1\_1 tools tape into the tape drive.
- ☐ 2. Log in as user `Rational`, which must be a member of group *Privileged*.
- ☐ 3. **LOAD\_TAPE** *[5 minutes]*  
This step restores the Training Release1\_1\_1 tape. This step can be executed from any command window. There should be no errors.
- ☐ 4. Go to `!Machine.Release.Archive.Training.Release1_1_1`.

The following steps utilize the procedure `Do_Step` as described previously.

- ☐ 5. **RESTORE\_NOTES** *[1 Minute]*  
This step restores the release note `Training_Release1_1_1_LPT` to `!Machine.Release.Release_Notes`.
- ☐ 6. **RECORD\_INSTALLATION** This step records release information in `!Machine.Release.Current.Products` This is a required step and establishes that the release has successfully been installed.
- ☐ 7. Only execute the step name(s) corresponding to the course(s) you wish to install on the machine (see release note). It is assumed that the product(s) required to support installation of a course are resident and operational on the machine.  
Valid names are:  
  
`ENV_FUNDAMENTALS`  
`ENV_ADVANCED_TOPICS`  
`ENV_TOOLSMTIHG`  
`ENVIRONMENT`  
`NETWORKING`  
`PROJECT_DEVELOPMENT_METHODS`  
`CMVC`  
`SYSTEM_MANAGEMENT`  
`MC68020_BARE`  
`MC68020_OS2000`  
`RCF_FUNDAMENTALS`  
`RCI_FUNDAMENTALS`

RDF\_2167A\_FUNDAMENTALS  
RDF\_2167A\_TOOLSMITHING  
RDF\_2167\_FUNDAMENTALS  
RDF\_2167\_TOOLSMITHING  
TESTMATE

Once a course has been restored, follow the instructions in the release note for creating the appropriate number of training user accounts.





# A

---

## Procedure Do\_Step

---

The majority of the steps which must be performed for this installation utilize the procedure `Do_Step` (included as part of the release tape and located in `!Commands`) to execute the required sequence of commands which implement the step. This procedure **must always** be executed from a command window in the release library

`!Machine.Release.Archive.Training.Release1_1_1`

Each step which utilizes procedure `Do_Step` will be of the form

|                                |                                        |
|--------------------------------|----------------------------------------|
| <code>&lt;STEP_NAME&gt;</code> | <code>[&lt;TIME TO EXECUTE&gt;]</code> |
|                                | <code>&lt;DESCRIPTION&gt;</code>       |

where `<STEP_NAME>` is passed as the parameter to procedure `Do_Step` which performs the necessary actions to complete the step, `<TIME TO EXECUTE>` is the amount of time the step takes to execute (this may be expressed as a range, in which case the lower values is typically the minimum, and the upper value is an estimate of the maximum), and `<DESCRIPTION>` is a description of what the step does, including any action which you will need to manually perform. For example:

`FOO`

*[5 Minutes]*

This step is an example of the format used for steps which are executed using procedure `Do_Step`. To execute step `FOO`, you would go to `!Machine.Release.Archive.Training.Release1_1_1` and in a command window, execute:

```
Do_Step ("FOO");
```

which would result in automatic execution of all commands required to implement this step. If you needed to take any manual action in addition to executing `Do_Step`, it would be noted as :

☐ Perform some manual check

If any errors occur for this step, you should always fix the problem before proceeding on to the next step. Each step is defined by a fragment of Ada code which is executed by `Program.Run`. These fragments are stored in the `Steps` file

located in the `Command_Data` library of the release. In the event you want to modify a step, you can use the "PROMPT => <STEP>" form when invoking `Do_Step`. See the spec of procedure `Do_Step` (located in `!Commands`) for more detailed information.

**Warning:** *If you interrupt the execution of `Do_Step` (by using `Job.Interrupt` such as `CONTROL-G`) it is possible that certain interactive commands executed by some steps, such as `Common.Definition`, may fail with an exception and display a message such as*

*Unable to read file due to Constraint\_Error (Null Access)*

*In general, this message has no negative impact on the execution or completion of the step, and can be ignored.*

**Note:** *When multiple steps are executed, you will be prompted between steps with the following:*

*Continue with <Step Name> step (Continue | Skip | Quit)? [Continue] : [input]*

*Select the action to be taken, one of `Continue`, `Skip`, or `Quit`. `Continue` (the default if `PROMOTE` is entered without typing anything) results in the step <Step Name> being executed. `Skip` will skip <Step Name> and prompt with the step following <Step Name>. `Quit` results in termination of `Do_Step` without further steps being executed.*

# B

---

## Feedback

---

---

In order to help us improve our instructions and make installations easier to do, the following form is provided to allow you to give us feedback. Please complete and send along with a printout of the file `Do_Step_Execution_Time` located in `!Machine.Release.Archive.Training.Release1_1_1.Logs` to:

Rational  
Atten: SMSE  
3320 Scott Blvd.  
Santa Clara, CA 95054-3197  
Re: Training

- ☐ 1. How long did the installation take you?
- ☐ 2. Did the installation proceed as described in the instructions?
- ☐ 3. What could be done to improve these (or other) instructions/installations?
- ☐ 4. What did you like about this set of instructions/installation?

# SMSE Checklist

## MC68020\_OS2000

### Release7\_2\_2

☒ **PRODUCT QA STAMP (On Packing List)**

☒ **SESSION AUTHORIZATION**

| <input checked="" type="checkbox"/> <u>PART NUMBER</u> | <u>DESCRIPTION</u>                                            |
|--------------------------------------------------------|---------------------------------------------------------------|
| <input checked="" type="checkbox"/> 509-003218-002     | MC68020_OS2000 Release7_2_2 Authorization Codes               |
| <input checked="" type="checkbox"/> 505-003249-001     | _____ Session Authorizations<br>Support Activity Report (SAR) |

☐ **PRODUCT REQUIREMENTS**

| <input checked="" type="checkbox"/> <u>PART NUMBER</u> | <u>DESCRIPTION</u>                                                                                                                                                                        |
|--------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <input checked="" type="checkbox"/> 507-003218-007     | Installation Procedure for MC68020_OS2000 Release7_2_2                                                                                                                                    |
| <input checked="" type="checkbox"/> 508-003218-007     | MC68020_OS2000 Release7_2_2 Release Note                                                                                                                                                  |
| <input checked="" type="checkbox"/>                    | <u>INSTALLATION ACTIVITY</u>                                                                                                                                                              |
| <input type="checkbox"/>                               | Full Installation in Field Required.                                                                                                                                                      |
| <input checked="" type="checkbox"/>                    | Partial installation at Factory performed, Field installation of unchecked steps required, starting with step <u>1</u> of <i>Installation Procedure for MC68020_OS2000 Release7_2_2</i> . |
| <input type="checkbox"/>                               | Complete installation performed at Factory.                                                                                                                                               |

- OVER -

☐ FIELD INSTALLATION REQUIREMENTS

|                          | <u>PART NUMBER</u>                      | <u>DESCRIPTION</u>               |
|--------------------------|-----------------------------------------|----------------------------------|
| <input type="checkbox"/> |                                         | <u>TAPE MEDIA</u>                |
|                          | <input type="checkbox"/> 541-002872-001 | 9 Track                          |
|                          | <input type="checkbox"/> 541-002873-001 | 8mm                              |
| <input type="checkbox"/> | 690-003218-013                          | MC68020_OS2000 Release7_2_2 Tape |

---

Installation Procedure

---

---

Release7\_2\_2

---

---

**Copyright © 1992 by Rational**

---

**PN 507-003218-007**

**This document is subject to change without notice.**

**Rational and R1000 are registered trademarks and Rational Environment is a trademark of Rational.**

**Rational, 3320 Scott Boulevard, Santa Clara, California 95054-3197**

## Table Of Contents

|                                     |    |
|-------------------------------------|----|
| 1 Installation                      | 1  |
| Overview                            | 1  |
| Assessing Impact                    | 1  |
| Verifying Prerequisites             | 2  |
| Preparation                         | 3  |
| Installing the Release              | 3  |
| Testing                             | 7  |
| Restoring User State                | 7  |
| Cleaning up                         | 7  |
| Appendix A Procedure Do_Step        | 9  |
| Appendix B System Manager Checklist | 11 |
| Appendix C Feedback                 | 13 |





---

# 1

---

## Installation

---

---

### Overview

---

Estimated time for installation: 3 hours.

When installing the MC68020\_OS2000 release, you can view the upgrade process as a series of phases. These phases must be performed in a serial manner.

- ☐ 1. Assessing the impact of the release on the user community
- ☐ 2. Verifying prerequisites to the installation
- ☐ 3. Preparing the user community and the machine
- ☐ 4. Loading the release
- ☐ 5. Testing the new release
- ☐ 6. Restoring state
- ☐ 7. Cleaning up

These phases must be performed in a serial manner, as must the steps in each of the phases. This installation note is organized by phase, with each phase comprising a section of this document. Once you have performed this installation, please fill out and return the feedback form that is the last page of these instructions along with a printout of the `Do_Step_Execution_Time` found in the `Logs` library of the release archive. Your feedback is very important in helping us to improve our installation instructions.

The following sections take you through the installation process a phase at a time. Be sure to complete ALL of the phases of installation and to perform these phases in the order in which they appear here.

**Warning:** Users may be logged in during this installation, but must NOT be viewing or working with CDF related objects.

---

## Assessing Impact

---

In this phase of the install, you will assess the impact of the release on the user community and the machine.

- ☐ 1. This installation will take a minimum of 4 hours on a system which has never had the MC68020\_OS2000 product installed. For systems which have a previous version of the MC68020\_OS2000 product installed along with compiled units, this time will increase, dependent upon the amount of time required to demote and recompile all units.
- ☐ 2. All user defined models will need to be modified to link to the new predefined location. In addition, all worlds/views which used these models will need to be refreshed with the new links.
- ☐ 3. When installing a new Environment release or a new layered product release, private activity files may become obsolesced (i.e., the subsystems they reference may not be compatible with the software being upgraded to). When installing several products at once, the Environment release should be installed first. Subsequently, each layered product can be installed. Following this process will result in the system default activity (`!Machine.Release.Current.Activity`) referencing a compatible set of subsystems.

After the upgrades have been completed, each user should merge the contents of the default activity into his or her own private activity file with the command `Activity.Merge`.

- ☐ 4. During this installation no users should be logged onto the system as there is potential they may cause portions of the installation to fail.
- ☐ 5. All *working*<sup>1</sup> MC68020\_os2000 target libraries will be demoted, requiring recompilation.

---

## Verifying Prerequisites

---

In this phase of the installation, you will verify that the prerequisites for installing this upgrade have been met.

- ☒ 1. You have read the release note.
- ☒ 2. You have read this entire install note.
- ☒ 3. The system Environment must be D\_12\_5\_0 (or later)

<sup>1</sup>Frozen worlds will not be demoted; i.e. released CMVC views.

- ☐ 4. If the MC68020\_OS2000 product has previously been installed on the machine, then the compiler must be running on the machine prior to installing this release.

---

## Preparation

---

In this phase of the installation you will prepare both the user community and the machine for the installation.

- ☐ 1. All MC68020\_OS2000 libraries will be demoted. Request that the users demote to the Archived state and/or delete all unnecessary MC68020\_OS2000 Ada units.

---

## Installing the Release

---

In this phase, you will load the tools required to install the user visible part of the release onto the machine. The release world located in

`!Machine.Release.Archive.MC68020_OS2000.Release7_2_2`

contains nested archives. This phase of the install will take a minimum of 3 hours on a machine which has not had the MC68020\_OS2000 installed before. For machines which have MC68020\_OS2000 code compiled, this time will increase dependent upon the amount of recompilation required.

*The following steps utilize the procedure `Do_Step` as described in the appendix on page 9.*

- ☒ 1. Load the MC68020\_OS2000 Release7\_2\_2 tools tape onto the tape drive.

- ☒ 2. Log in as a user which is a member of group *Privileged*.

- ☒ 3. **LOAD\_TAPE**

*[10 Minutes]*

This step restores the MC68020\_OS2000 Release7\_2\_2 tape. This step can be executed from any command window. You will need to answer the mount request on the operators console. There should be no errors.

- ☒ 4. Go to `!Machine.Release.Archive.MC68020_OS2000.Release7_2_2`.

- ☒ 5. **AUTHORIZATION\_CHECK***[1 Minute]*

This step will verify that the appropriate products are authorized for this machine. These products are :

CMVC  
CMVC.Source\_Control  
Motorola\_68k  
MC68020\_OS2000

If there are products which need to be authorized, this will be indicated and a command window created with an appropriate call. Modify the parameters and authorize the products required using the codes from the Authorization Codes sheet.

## ☒ 6. USERS\_CHECK

*[Optional - 1 Minute]*

This is an optional step which sets the login limit to 1 and verifies that there are no other users logged into the system. Note that the login limit will be restored to an unlimited value when the machine is rebooted later during the install.

## ☒ 7. RELEASE\_RESTORE

*[35 Minutes]*

This step restores the Release and Release\_Notes archive. In addition, the new predefined objects are restored to their new location. When completed, a filtered error log is displayed (Release\_Restore\_Log\_Summary located in

!Machine.Release.Archive.MC68020\_OS2000.Release7\_2\_2.Logs). Examine this log for errors. Ignore the following errors if present:

```
*** Exception (AMBIGUOUS_SWITCH_NAME) setting switch
... Elab_Order_Listing to FALSE.
```

## ☐ 8. Go to

!Targets.Implementation.Release\_7\_2\_2.Motorola\_68k.Mc68020\_Os2000.

- ☐ 9. In order to allow code views to be transferred between machines, the predefined units are now in a subsystem view. For this reason the links in all pre-existing worlds which are to be recompiled with Rev7\_2 must point to the new predefined units.

The file Model\_Map is a text file containing pairs of model names. The first is an old model name which is to be replaced by the model name following it. The names of old user-defined models and their corresponding new models can be added to this list so that they will be updated automatically.

The World\_List and Model\_World\_List files can be altered to avoid updating certain worlds.

The file World\_List is used to determine which worlds are demoted when the INSTALL\_PRODUCT step is run. It contains a naming expression which resolves to the worlds to be demoted, and should not name the Rev6 predefineds or the Rev6 runtimes. The Model\_World\_List file is used by the UPDATE\_MODELS step to determine which worlds to examine and potentially update their reference. Not all worlds named in this file will be updated. In particular, model worlds,

predefineds, and released views will never be updated.

☐ Examine (and possibly alter) the files `Model_Map`, `World_List`, and `Model_World_List`.

✓ 10. Go to `!Machine.Release.Archive.MC68020_OS2000.Release7_2_2`.

✓ 11. **INSTALL\_PRODUCT**

*[1.5 - 8 Hours]*

This step executes the `INSTALL_PRODUCT` procedure located in

`!Targets.Implementation.Release_7_2_2`.

When completed, a filtered error log is displayed (`Install_Log_Summary` located in

`!Targets.Implementation.Release_7_2_2`).

Examine this log for errors. Ignore the following errors if present:

```
*** [_Cdf_Worlds_List]?'c(ada).

*** Unable to iterate over child objects while setting
... ACLs because of some unknown failure in the directory
... system while operating on <1>
*** Cmvc_Access_Control.Initialize is quitting after
... errors.

*** Can't resolve
 "_archive_freeze_file" ...

*** Had trouble finding units to
 demote...

*** Had trouble finding objects to
 delete...
```

✓ 12. **RESTART\_COMPILER**

*[2 Minutes]*

This step restarts the compiler (1 stream). Note that you can have from 1 to 3 streams, allowing from 1 to 3 CDF compilations to occur simultaneously.

✓ 13. **UPDATE\_MODELS**

*[1 Minute - ?]*

This step calls the `Update_Models` procedure located in

`!Targets.Implementation.Release_7_2_2`.

`Motorola_68k.MC68020_OS2000`, which will

change the model in subsystem views according to the `Model_Map`, and update the links in other worlds to point to the new predefined packages.

The source of this procedure is provided and can be altered to fit local requirements. This procedure uses the `World_List` and `Model_Map` files. Ignore any warning (!!!) messages.

#### ☒ 14. BUILD\_PROGRAM\_LIBRARIES [2 minutes - ?]

This step builds program libraries for all of the worlds demoted by execution of procedure `Update_to_7_2` using `Program_Library_Maintenance.Build` (located in `!Tools.Program_Library`). This ensures that program libraries exist for all MC68020\_OS2000 objects to be compiled using the new compiler.

#### ☒ 15. LOAD\_EEDB

[3 Minutes]

This step loads (and elaborates) the EEDB subsystem for `CROSS_DEVELOPMENT`, updating the currently running configuration. This step will only change the subsystem if the one currently executing is of a previous version than the one included in this release. A summary error file (`Load_EEDB_Log_Summary` located in `!Machine.Release.Archive.MC68020_OS2000.Release7_2_2.Logs`) is displayed. There should be no errors.

#### ☐ 16. [2 Minutes]

All D85 target boxes which will use this release must have the new target components installed. Go to `Motorola_68k.MC68020_OS2000` library located in `!Targets.Implementation.Release_7_2_2` and execute the `Install_Product_On_Target` procedure:

```
Install_Product_On_Target
(Target_Machine => ">>Target Machine Name<<",
 Runtime_Module_Directory => "/h0/cmds/bootobjs",
 Runtime_Errors_Directory => "/h0/sys",
 Debugger_Module_Directory => "/h0/cmds");
```

This will download to the following files on the D85 target:

```
/H0/CMD5/BOOTOBJS/ADA_RUNTIME.
/H0/CMD5/BOOTOBJS/ADA_TASK.
/H0/SYS/ART_ERRMSG.
/H0/CMD5/RATCOM.
/H0/CMD5/RATDBG.
/H0/CMD5/RATSEND.
/H0/CMD5/RATSHUT.
/H0/CMD5/DBGCOM.
```

☐ Note that the startup command file on the D85 system may need to be modified to get the runtime modules loaded and initialized upon boot. An entry of the following form should be in the startup file of the D85 system:

```
load /h0/cmds/bootobjs/ada_runtime; ada_runtime
load /h0/cmds/bootobjs/ada_task
```

☐ Reboot the target box to effect loading of the new runtime modules.

## ✓ 17. RECORD\_INSTALLATION [1 Minute]

This step records release information for the MC68020\_OS2000 product. This is a required step.

---

### Testing

---

None.

---

### Restoring User State

---

In this phase of the install you will restore to the original state user units which were demoted by the installation. The time required for this phase will vary widely, depending on how much state is required to be restored.

#### ☐ 1. RESTORE\_STATE

This step repromotes user units which were demoted by this installation. The files `Install_File`, `Code_File`, and `Freeze_File`, located in `!Targets.Implementation.Release_7_2_2` are used as indirect files to repromote and refreeze units affected by the upgrade. The units listed in these files are in the correct compilation order. The `Install_file` and `Code_File` are split up into multiple files, each containing 300 lines or less, allowing more control over the recompilation and it's affect on disk garbage generation/collection. Compilation on each file is run as a separate job. When done, a summary output log is displayed (`!MACHINE.RELEASE.ARCHIVE.MC68020_OS2000.Release7_2_2.LOGS.Restore_User_State_Log_Summary`). Examine this for errors.

---

### Cleaning up

---

In this phase, you will perform some cleanup activities. The time required for this phase will vary widely, depending on how much cleanup is required to be done.

- ☐ 1. Go to `!Machine.Release.Archive.MC68020_OS2000.Release7_2_2.Logs` and print out the file `Do_Step_Execution_Time`. Return this printout to Rational with other feedback supplied on the last page of this note. Note that this step must be executed prior to performing the following step which will destroy the `Do_Step_Execution_Time` file.



☐ 2. DESTROY\_ARCHIVE

*[1 Minute]*

This step destroys the release archive library in `!Machine.Release.Archive`. Only execute this step when you are confident that the release has been properly restored.

- ☐ 3. Destroy older release libraries located in `!Targets.Implementation`. It is recommended that the latest release library be maintained for possible future use, especially if there were any errors during the installation, but this is not a requirement and this library may also be deleted.

- ☐ 4. Take a full Environment backup.

# A

---

## Procedure Do\_Step

---

The majority of the steps which must be performed for this installation utilize the procedure `Do_Step` (included as part of the release tape and located in `!Commands`) to execute the required sequence of commands which implement the step. This procedure **must** always be executed from a command window in the release library

`!Machine.Release.Archive.MC68020_OS2000.Release7_2_2`

Each step which utilizes procedure `Do_Step` will be of the form

|                                |                                        |
|--------------------------------|----------------------------------------|
| <code>&lt;STEP_NAME&gt;</code> | <code>[&lt;TIME TO EXECUTE&gt;]</code> |
|                                | <code>&lt;DESCRIPTION&gt;</code>       |

where `<STEP_NAME>` is passed as the parameter to procedure `Do_Step` which performs the necessary actions to complete the step, `<TIME TO EXECUTE>` is the amount of time the step takes to execute (this may be expressed as a range, in which case the lower values is typically the minimum, and the upper value is an estimate of the maximum), and `<DESCRIPTION>` is a description of what the step does, including any action which you will need to manually perform. For example:

**FOO**

*[5 Minutes]*

This step is an example of the format used for steps which are executed using procedure `Do_Step`. To execute step **FOO**, you would go to `!Machine.Release.Archive.MC68020_OS2000.Release7_2_2` and in a command window, execute:

```
Do_Step ("FOO");
```

which would result in automatic execution of all commands required to implement this step. If you needed to take any manual action in addition to executing `Do_Step`, it would be noted as :

☐ Perform some manual check

If any errors occur for this step, you should always fix the problem before proceeding on to the next step. Each step is defined by a fragment of Ada code which is executed by `Program.Run`. These fragments are stored in the `Steps` file

located in the `Command_Data` library of the release. In the event you want to modify a step, you can use the "PROMPT => <STEP>" form when invoking `Do_Step`. See the spec of procedure `Do_Step` (located in `!Commands`) for more detailed information.

**Warning:** *If you interrupt the execution of `Do_Step` (by using `Job.Interrupt` such as `CONTROL-G`) it is possible that certain interactive commands executed by some steps, such as `Common.Definition`, may fail with an exception and display a message such as*

*Unable to read file due to Constraint\_Error (Null Access)*

*In general, this message has no negative impact on the execution or completion of the step, and can be ignored.*

**Note:** *When multiple steps are executed, you will be prompted between steps with the following:*

*Continue with <Step Name> step (Continue | Skip | Quit)? [Continue] : [input]*

*Select the action to be taken, one of `Continue`, `Skip`, or `Quit`. `Continue` (the default if `PROMOTE` is entered without typing anything) results in the step <Step Name> being executed. `skip` will skip <Step Name> and prompt with the step following <Step Name>. `Quit` results in termination of `Do_Step` without further steps being executed.*

# B

---

## System Manager Checklist

---

- ☐ 1. Each user should merge the contents of the default activity into his or her own private activity file with the command `Activity.Merge`.

```
Activity.Merge (Source => "!Machine.Release.Current.Activity",
 Mode => Activity.Differential,
 Target => ">>User Activity<<");
```

Note that this command may have the undesirable effect of modifying spec/load view settings which were explicitly set in the user activity.

- ☐ 2. Update all user models to reference the new location of the predefines. Refresh all worlds/views which used these models to reference the new locations.
- ☐ 3. Update all D85 target boxes with new target components using procedure `Install_Product_On_Target` located in `!Targets.Implementation.Release_7_2_2`.
- ☐ 4. Distribute the release note to user community. A postscript and lineprinter copy of the release note can be found in `!Machine.Release.Release_Notes`.
- ☐ 5. Update the Daily Message indicating that MC68020\_OS2000 Release7\_2\_2 has been installed.
- ☐ 6. Set the ACL's on new specs/files restored as part of this release, if desired.
- ☐ 7. Make a Full backup of the Environment.



# C

---

## Feedback

---

---

In order to help us improve our instructions and make installations easier to do, the following form is provided to allow you to give us feedback. Please complete and send along with a printout of the file `Do_Step_Execution_Time` located in `!Machine.Release.Archive.MC68020_OS2000.Release7_2_2.Logs` to:

Rational  
Atten: SMSE  
3320 Scott Blvd.  
Santa Clara, CA 95054-3197  
Re: MC68020\_OS2000

- ☐ 1. How long did the installation take you?
- ☐ 2. Did the installation proceed as described in the instructions?
- ☐ 3. What could be done to improve these (or other) instructions/installations?
- ☐ 4. What did you like about this set of instructions/installation?

Rational Cross-Development Facility:  
M68020/OS-2000  
Release Information

Rev7\_2\_2 Release

Copyright © 1992 by Rational

Part Number: 508-003218-007

October 1992 (Software Release Rev7\_2\_2)

Motorola is a registered trademark of Motorola, Inc.

Rational is a registered trademark and Rational Environment is a trademark of Rational.

Rational  
3320 Scott Boulevard  
Santa Clara, California 95054-3197



# 1. Overview

Rev7\_2\_2 is a release of the Rational M68020/OS-2000 Cross-Development Facility (CDF). This release note lists new features added, problems fixed, and problems still outstanding since the previous release of the CDF.

The main features of this release are:

- Support for shared-code generics
- Support for spec/load views
- Support for code views
- Integration with "Program\_Library" facilities, providing faster prelinking
- Reduced compilation times
- Object files are now generated directly by the compiler

This release **REQUIRES** environment release D\_12\_4\_7 or later.

This release generates code that is incompatible with code generated by previous versions of the CDF.

## 1.1. Views in This Release

This release includes the following views:

- !TARGETS.IMPLEMENTATION.MC68020\_OS2000\_DEBUGGERS.REV11\_SPEC
- !TARGETS.IMPLEMENTATION.MC68020\_OS2000\_DEBUGGERS.REV11\_0\_3
- !TARGETS.IMPLEMENTATION.MC68020\_OS2000\_TARGET.REV11\_SPEC
- !TARGETS.IMPLEMENTATION.MC68020\_OS2000\_TARGET.CODE11\_0\_3
- !TARGETS.IMPLEMENTATION.MOTOROLA\_68K.REV7\_2\_SPEC
- !TARGETS.IMPLEMENTATION.MOTOROLA\_68K.CODE7\_2\_20
- !TARGETS.IMPLEMENTATION.MOTOROLA\_68K\_HOST.REV11\_5\_SPEC
- !TARGETS.IMPLEMENTATION.MOTOROLA\_68K\_HOST.CODE\_REV7\_13
- !TARGETS.IMPLEMENTATION.MOTOROLA\_68K\_TRANSFER.REV3\_SPEC
- !TARGETS.IMPLEMENTATION.MOTOROLA\_68K\_TRANSFER.REV3\_1\_1
- !TARGETS.IMPLEMENTATION.OBJECT\_CONVERSION.REV7\_2\_SPEC
- !TARGETS.IMPLEMENTATION.OBJECT\_CONVERSION.CODE7\_2\_0
- !TARGETS.IMPLEMENTATION.OBJECT\_MODULE.REV7\_2\_SPEC
- !TARGETS.IMPLEMENTATION.OBJECT\_MODULE.CODE7\_2\_0
- !TARGETS.IMPLEMENTATION.MC68020\_OS2000\_RUNTIMES.REV7\_2\_3
- !TARGETS.IMPLEMENTATION.MOTOROLA\_68K\_MACHINE.OS2000\_11\_0\_6

## 1.2. Changed Predefined Units

The predefined units themselves have not changed, but their location has. See Section 13, "Code Views and the Predefined Subsystem," for more information.

## 2. Status of Software Problems in This Release

### 2.1. Reported Problems Fixed

- 0-0251-6  
The compiler now deals more efficiently with constant strings.
- 1181016-Shei-Lsj  
Internal\_error - float constant out of range. This has been fixed.
- 4814499-Etoi-Kjm  
Pragma Pack on unconstrained arrays of booleans caused code generation error. This has been fixed.
- 5307212-Stan-Mv  
There were problems when declaring subtypes of short integer. This has been fixed.
- 721413-Etoi-Mv  
Internal error during phase: LAB. This has been fixed.
- 9883117-Etoi-Mv  
The code generator caused an error when it encountered certain combinations of multiply nested loops and case statements. This has been fixed.
- 9723630-0104-6  
Support for pragma Inline in compilation tools. Comp.Make now codes bodies as quickly as possible after the spec. This is not a full solution, but makes the problems much less likely to occur.
- 0-0248-9  
Combined views required because of inlining and generics. Since shared-code generics are no longer proscribed from spec views, it is now feasible to use spec-load subsystems with the Mc68020\_Os2000 target. Inlining subprogram calls through spec views is not supported, but this does not prevent the use of spec-load subsystems.
- 9723630-0022-1  
Instantiation in limited private task type. Fixed in this release.
- 9723630-0120-2  
Slow compilation and linking. The Cross\_Cg.Linker\_Generate\_Map switch can now be used to control whether a link map file is generated when a main program is coded.
- 2887093-Shei-Jst  
Middle pass problem with spliced contexts. Fixed in this release.
- 5003830-Euse-Lsj  
Constraint Error is no longer raised when compiling Nosc.Speller.
- 1680472-Shei-Lsj  
Constraint Error in stmt\_gen.gen\_stmt. This has been fixed.

- 4031776-Shei-Jst  
Array inequality on packed arrays of discretely failed. This has been fixed.
- 8518071-Etoi-Mv  
Evaluation of 3D boolean array failed. This has been fixed.
- 3244099-Rati-Pbk  
A request to zero only the portions of records which are unused because of alignment issues; the loops that are currently generated are expensive. This has been fixed.
- 7582583-Etoi-Mv  
A representation clause was accepted after the default determination of an entity. Since this situation is illegal (according to the LRM), this should have been flagged as an error. Fixed in Delta 3.0.
- SPR#900515-3  
CDF Code generation error (internal error in WRI)
- SPR#900822-16, 900822-24  
Deadlock or crash of operating system/ada runtime
- SPR#900928-3  
Constraint Error when assigning integer
- SPR#911003-10  
Unexpected Traps
- SPR#911230-1  
Package Command\_Line cannot handle long lines
- SPR#920130-5  
Compiler error in coding stage
- SPR#920203-24  
Priorities of main programs and tasks
- SPR#920205-2  
Rational allows pragma shared on records
- SPR#920305-1  
CDF Linker aborts with errors (Undefined symbol)
- SPR#920323-13  
Error in dead code elimination
- SPR#920505-17, 920522-13  
Collection sizes too large - waste of heap memory
- SPR#920505-23  
Elaboration fails (Array incorrectly initialized)
- MFP 74  
[ITX] Illegal operand combination for Am.Move
- MFP 73  
Program\_Error in Piwg
- MFP 72  
runtime problem
- MFP 71  
[ITX] E-Stack underflow

- MFP 69  
[WTR] Bad pc relative address
- MFP 68  
[WTR] Compiler capacity exceeded
- MFP 67  
[MID] in Decl\_Gen.Gen\_Decl - Constraint\_Error (Null Access)
- MFP 66  
[MID] Access\_Gen.Operations.Unchecked\_Deallocation: Formal\_Private\_Class.
- MFP 64  
[MID] Attempt to reconstruct object in current unit
- MFP 63  
[MID] in Stmt\_Gen.Gen\_Stmt - Constraint\_Error (Null Access)
- MFP 62  
Sedt runtime problem
- MFP 60  
[MID] in Stmt\_Gen.Gen\_Stmt - !Lrm.System.Assertion\_Error
- MFP 59  
[CSE] Internal error (Unhandled Exception => Constraint\_Error (Array Index))
- MFP 58  
CDF accepts fixed point in out object, which leads to infinite loop and storage leak
- MFP 57  
[MID] in Decl\_Gen.Gen\_Decl - !Lrm.System.Assertion\_Error
- MFP 55  
[MID] in Code\_Generation.Generate - Constraint\_Error (Null Access)
- MFP 54  
Incorrect raise of Numeric\_Error
- MFP 53  
[MID] in Decl\_Gen.Gen\_Decl - Program\_Error (prompt executed)
- MFP 53a  
ERROR [LNK] Can't process library ...
- MFP 51  
Debugger puts wrong indicies for arrays in record
- MFP 50  
[MID] in Stmt\_Gen.Gen\_Stmt - !Lrm.System.Assertion\_Error
- MFP 49  
[ITX] Illegal operand combination for Am.Move.
- MFP 48  
[MID] Context.Enclosing\_Generic\_Package\_Id failed.
- MFP 46  
[CHK] Not a valid exp\_tree node
- MFP 45  
[ITX] Illegal operand combination for Am.Move.

- MFP 44  
[CHK] Null pointer.
- MFP 43  
[MID] Shared instance subprogram lacks Original\_Node attribute: 0.DN\_VOID.
- MFP 42  
[CHK] Object is down-level addressed.
- MFP 41  
[MID] Unexpected Diana: Types.Arrays. Reconstruct\_Array\_Info DN\_IN\_ID.
- MFP 40  
[MID] Unexpected Diana: Types.Arrays. Reconstruct\_Array\_Info DN\_IN\_ID.
- MFP 39  
[CHK] dot does not have a basetype
- MFP 38  
[MID] in Stmt\_Gen.Gen\_Stmt - !Lrm.System.Assertion\_Error [This is really the same as Mfp\_35]
- MFP 37  
[MID] in Code\_Generation.Generate - Constraint\_Error (Null Access)
- MFP 36  
[MID] Shared instance subprogram lacks Original\_Node attribute: 0.DN\_VOID.
- MFP 35  
Unimplemented feature: Task entry as shared generic actual subprogram.
- MFP 34  
Unimplemented feature: Shared generic subprogram skins over Unchecked Conversion
- MFP 33  
[ITX] Illegal address mode for Am.Add.
- MFP 32  
[MID] Shared instance subprogram lacks Original\_Node attribute: 0.DN\_VOID.
- MFP 31  
[MID] Float\_Util.Comparison\_Operation: bad class.
- MFP 30  
WARNING [MID] Reconstruction of instantiation base not implemented
- MFP 29  
[MID] Unimplemented Feature: Tasks in outermost level of shared generic
- MFP 28  
[MID] Unimplemented Feature: In-out array objects
- MFP 27  
[MID] Attempt to reconstruct object in current unit
- MFP 26  
Debugger Problem
- MFP 25  
[Cse] Constraint\_Error (Null Access) At #41b503, #4ff
- MFP 23  
Bad code for variant record

- MFP 22  
[ITX] Exception in CODER while coding routine \_RECORD ...
- MFP 21  
[CSE] Constraint\_Error (Null Access) at #52A902, #5C6
- MFP 20  
Execution-time misuse of base register
- MFP 18  
[CSE] Constraint\_Error (Type Range) at #3A4104, #4BC
- MFP 16  
Ada runtime traps not handled as exceptions
- MFP 15  
[ASY] Value for constant block does not fit element size
- MFP 14  
[ITX] Exception in CODER while coding routine \_RECORD ...
- MFP 13  
[ITX] E-Stack underflow.
- MFP 11  
[Cse] Constraint\_Error (Type Range) At #320902, #C60
- MFP 10  
[Itx] Attempt To Restore Spilled Register ...
- MFP 9  
[Lab] Constraint\_Error (Null Access) At #383102, #Be6
- MFP 8  
[Evo] Internal Error In Phase Evo At ...
- MFP 7  
[Mid] Unexpected Diana: Instantiation\_Checking.Get\_Indices
- MFP 6  
[Itx] Internal Coder Stack Inconsistency
- MFP 4  
[Mid] In Stmt\_Gen.Gen\_Stmt - !Lrm.System.Assertion\_Error
- MFP 1  
[Mid] In Dec\_Gen.Gen\_Decl Assertion\_Error At #3aa902, #B09

## 2.2. Reported Problems Not Fixed in This Release

The following problems reported against previous revisions are *not* fixed in the current release:

- SPR#910614-1  
Memory allocated in a declare block is not reused in subsequent non-overlapping declare blocks.
- SPR#910731-1  
Text\_Io.Get\_Line skips a line of input.

- MFP 65  
[MID] Unimplemented Feature: Unchecked\_Conversion of a formal private type used as a formal subprogram.
- MFP 54a  
The debugger can be invoked correctly, but the directory of the program must be given as part of the name of the program.
- MFP 47 Debugger unable to put complex packed object
- MFP 24 Unimplemented Feature: Formal fixed point types
- MFP 19 Size representation clause failed to compile
- MFP 17 [WTR] Lost symbol. when coding task body
- MFP 12 [Chk] Object Is Down-Level Addressed.
- MFP 5 Representation Clause Permitted After Forcing Occurrence
- MFP 3 [Mid] In Decl\_Gen.Gen\_Decl Assertion\_Error At #3abd02, #C9
- MFP 2 Undefined Symbol Referenced During Link Time

### 2.3. Other Issues Not Dealt With in This Release

This section contains suggestions and errors that need further clarification or evaluation.

- 9723630-0020-9, 9723630-0067-2, SPR#890627-3, SPR#980731-3  
Odd length of modules. Will be considered for a future release.
- 4815206-Etoi-Kjm  
Stack usage for main programs have changed radically. The scheme for determining the size of discriminated records was improved, causing an object which used to be allocated on the heap to be allocated on the stack. It appears from the PRS that the customer has a tool which thinks that this storage is never assigned to, although the generated code does seem to be doing the assignment. This may no longer be a problem.
- 0-0215-1  
Warning messages "Tree\_Check\_Error" still remain. Waiting for test case from customer.
- 9723630-0105-1  
An assignment statement that doesn't work. This is not reproduceable. Waiting for test case from customer.
- SPR#881116-1  
Minimize run-time library checks of linker command file
- SPR#881212-2  
Error path starting at /h0 costs memory
- SPR#881222-3  
Add option "CPU" to Debug.Invoke
- SPR#890509-1  
Debugger support for tuning heap and stack sizes
- SPR#890518-2  
Incremental change of pragma main

### 3. Changes and New Features

The following features have been changed or added. Consult your Rational technical representative for more details.

#### 3.1. Improvements in Compiler Operation

##### 3.1.1. Incremental Changes

Support for incremental operations has been improved; in particular, the user may now make incremental changes in coded package specifications. As a result, the OS-2000 compiler now supports all incremental operations provided by the R1000's native compiler.

##### 3.1.2. Compile-Time Improvements

Compile-time when handling floating point literals has been greatly reduced. This is especially visible with large array aggregates consisting of float literals.

The Build\_Conflict\_Graph algorithm now runs over fewer nodes, which improves performance because it is an  $O(N)^2$  algorithm.

Several expensive optimizer algorithms were improved to reduce compile times.

The assembly-code writer has been replaced with an object-code writer. This eliminates the step of running the assembler, which was previously a bottleneck in the code generation process, and results in a substantial reduction in compile time. If the Cross\_Cg.Asm\_Listing switch is set to True, a listing file is generated by running the assembly-code writer in addition to the object-code writer.

##### 3.1.3. More Efficient Use of Compiler Streams

Invocations of the CDF back ends have always been serialized; that is, each compiler stream can only handle a single compilation at a time. This is still true, but it is now the case for main programs that the serialization lock is relinquished as soon as the linker is invoked. This means that the compilation stream on which a main program is being coded is freed for another compilation as soon as the linker is invoked, rather than after it completes.

##### 3.1.4. Address Clause Support

Support for address clauses on objects has been improved:

- The address need not be determinable at compile time if the object is an array or record.
- Addresses clauses may now be used on arrays and records whose size is not determinable at compile time.

The address given in the address clause must be determinable at compile time unless the specified object is an array or record, in which case the address may be dynamic. An address is considered to be determinable at compile time only if it takes the form of a call to System.To\_Address with a compile-time determinable argument.



An object may not have both pragma Import\_Object and an address clause.

An address clause with a static address will cause the object to be addressed using absolute addressing. Pragma Import\_Object will cause the object to be addressed A5-relative, with the linker resolving the external symbolic name to get the offset.

An object having an address clause is NOT subject to default initialization unless this is explicitly requested using pragma Initialize. Pragma Initialize has the following format:

```
pragma INITIALIZE(simple_name);
```

This pragma causes the imported or address-claused object denoted by simple\_name to become subject to default initialization.

The parameter to pragma Initialize, simple\_name, must denote a variable declared earlier in the same declarative part. The variable must be an array or record and must have an address clause or pragma Import\_Object applied to it before the occurrence of pragma Initialize. The object must not have an explicit initial value.

### 3.1.5. <Elab\_Order\_Listing> file

If the Cross\_Cg.Elabor\_Order\_Listing switch is set to True, the Os2000 CDF now generates an <Elab\_Order\_Listing> file when a main program body is coded. This file consists of a list of the fully qualified names of the units in the closure of the main program, listed in the order of their elaboration. The file also includes comments indicating how a unit came to be included in the closure. The format of this file is suitable for use as an indirect file in a naming expression, although names of any units out of code views will not be resolvable.

When the main program is demoted to installed, the <Elab\_Order\_Listing> file is destroyed.

### 3.1.6. Object-Module Converter

The object-module converter is now run automatically as part of coding a main program, generating an <Os2000> file. When a main program is demoted from coded to installed, the <Os2000> file is automatically deleted.

Since the <Os2000> file is now automatically generated, the compiler no longer creates a <Exe> file by default. An <Exe> file can be created by setting the Cross\_Cg.Produce\_Full\_Exe switch to true.

## 3.2. Improvements in Generated Code

Compiler code generation has been improved in the following areas (among others):

- The propagation of constraint information has been improved. This improves the elimination of unnecessary overflow and constraint checks.
- To reduce overall register costs, the allocation of common subexpressions has been integrated with that of other objects.
- The allocation of volatile registers has been improved. This increases the number of opportunities for stack frame elimination when the debugging level is PARTIAL or NONE.
- Object preferencing has been improved, reducing the number of move instructions.

- Processing of case statements by the optimizer has been much improved. The optimizer chooses from a number of different techniques to implement each case statement, depending on factors such as its size and density. Different techniques are combined when appropriate, for example to handle a large, generally sparse case statement having small regions of high density.
- Constant folding has been expanded. In particular, constant folding of floating point expressions has been improved.

### 3.3. Shared-Code Generics

Users may request that all instantiations of a given generic share the same general code, that all instantiations of a given generic have their own custom code, or that all instantiations by default share the same code but that specified instantiations have their own code. Two new pragmas, `Generic_Policy` and `Instance_Policy`, are used to specify the manner in which code is to be generated for generics.

#### 3.3.1. Pragma `Generic_Policy`

```
pragma GENERIC_POLICY ([GENERIC_UNIT =>] simple_name,
 [CODE =>] REPLICATED | SHARED)
```

The simple name must denote a generic package or subprogram.

The pragma and the declaration of the named generic must occur immediately within the same declarative part or package specification; the generic declaration must occur before the pragma. Only one `Generic_Policy` pragma may be applied to a generic.

The pragma specifies how code should be generated for the generic. If the `Shared` option is selected, code will be generated for the generic itself; this code will be used for all instantiations of the generic (except as noted below). If the `Replicated` option is selected, code will not be generated for the generic, but rather for each instantiation of the generic. The replicated code for an instantiation will be smaller and faster than the shared code for the corresponding generic.

Demoting the body of a replicated generic to installed causes all instantiations of that generic to also be demoted to installed, whereas demoting the body of a shared generic to installed does not cause demotion of any instantiations (except those that were explicitly made replicated using pragma `Instance_Policy`).

If pragma `Generic_Policy` is not specified, the compiler decides between `Shared` and `Replicated`. At present, the default is always `Replicated`.

Replicated generics are not permitted in spec views (therefore, a generic in a spec view must also have a generic policy pragma specifying that the code is to be shared). The generic policy must be the same for corresponding generics in a load view and spec view. This is currently not enforced, so care should be taken to ensure that all generics which appear in a spec view are implemented as shared-code generics.

To simplify the rules for spec/load view compatibility, and because complete control is afforded by the pragmas defined here, pragma `Optimize` does *not* play a role in selecting between shared and replicated code.

### 3.3.2. Pragma Instance\_Policy

When a policy of shared has been selected for a generic, the user may still wish for certain instantiations to have replicated code for performance reasons. Pragma Instance\_Policy allows this to be specified for a particular instantiation.

```
pragma INSTANCE_POLICY ([INSTANTIATION =>] simple_name,
 [CODE =>] REPLICATED)
```

The simple name must denote an instantiation of a generic (which may be either shared or replicated).

The pragma and the declaration of the named instantiation must occur immediately within the same declarative part or package specification; the instantiation must occur before the pragma. If the simple\_name names several overloaded subprogram instantiations in the current declarative part or package specification, the pragma applies to all of them. Only one Instance\_Policy pragma may be applied to a single instantiation.

The pragma is ignored if the instantiation refers to a generic in a spec view.

### 3.3.3. Shared-Code Generic Features

The following subset of shared-code generic features is supported in this release:

- Formal floating point types
- Formal private and limited private types
- Formal access types
- Formal subprograms
- Formal objects of mode IN
- Formal arrays
- Formal discrete and integer types
- Formal scalar objects of mode IN OUT
- Formal nondiscriminated record objects of mode IN OUT
- Formal array objects of mode IN OUT
- Generics (shared or replicated) nested immediately within shared generic packages
- Subunits, and subunits of subunits

A Generic\_Policy pragma with a policy of Shared will be rejected if the generic has any of the following formal parameters:

- Formal fixed point types
- Formal discriminated private and limited private types
- IN OUT discriminated objects
- IN OUT formal private or limited private objects

Exceptions declared in shared generics are not handled strictly correctly. The Ada language requires that such exceptions be replicated for each instantiation (as happens naturally when a macro-expansion technique is used); at the present time, this does not occur for shared generics.

There is a temporary restriction on the use of bodiless library unit generic packages. Use of this construct will result in an error at code generation time.

The temporary restriction concerning the declaration of task types and objects in shared generic package specs and bodies has been removed; such declarations will now be correctly processed.

### 3.3.4. Guidelines for Using Shared Code Generics

This section discusses the performance impact of shared code generics and offers some suggestions on when to use them and how to optimize performance when they are used. In general, shared code generics will be less time efficient than replicated generics, but will often be more space efficient as well as permitting the spec of the generic to be placed in a spec view. The code is less efficient because code is generated to handle all the parameters with which a generic can be instantiated, rather than only the code necessary for the specific parameters which are available at the instantiation site (where code for replicated generics is generated). Shared generics must also manage the state of a generic package (i.e., objects declared in the package). This is always less efficient, since it requires an extra level of indirection, and sometimes more than one. In general, shared code generics will be more space efficient because code will only be generated once rather than at each of the instantiation sites, but only if there are several instantiations of the generic (it is always more efficient to use replicated generics if there is only one instantiation).

It is also possible that a shared code generic will allocate more heap or stack than the same generic replicated. This occurs when the shared generic needs to allocate space but has less information about how much space to allocate than it would if it were replicated. For instance, suppose a generic has a formal in object which is used as a bound in an array. If the generic is replicated, the compiler may be aware of the value that has been given to the object, in which case it will allocate the array statically. If the generic is shared, however, the array must be allocated dynamically, since the compiler must generate code without knowing the bounds of the array. Whether more space will be used because of a given generic depends heavily on what kinds of generic parameters it takes, so here is a type by type list of some of the effects.

- **Formal Floating Point Types**  
All simple objects of a formal floating point type will cause more stack space to be allocated, since the compiler allocates enough space to store a double-word value. Arrays of formal floats will be only as long as is needed, but they will be dynamic in all cases, since the size of the element cannot be known at compile time.
- **Formal Subprograms**  
All formal subprograms are implemented by passing a compiler generated subprogram which does any needed type conversions and then calls the actual subprogram. They have little effect on the stack or heap used, although they do cause an extra stack frame to be allocated.
- **Formal Access Types**  
No extra space is allocated for an object of a formal access type, but extra master layers may be created in some cases where we are unable to tell whether the access type is an access to a task type.
- **Formal Objects of Mode IN**  
Formal IN objects have no significant effect on the amount of space allocated.

- **Formal Arrays**  
Formal arrays cause little difference in the amount of space allocated on either heap or stack.
- **Formal Private Types**  
Formal private types can incur much more overhead with shared code generics than with replicated generics. This is because the compiler must now allow for the possibility of any type. In general, objects of a formal private type will be allocated a single word on the stack. If the actual type is a simple type (i.e. a scalar), the word will contain the value of object. If the actual type is more complex, the word will point to another object allocated dynamically which will contain the actual data, or possibly more pointers to other dynamic objects. With this scheme, little more space is allocated that is necessary, but space which would have been on the stack in a replicated generic will now be allocated dynamically, sometimes on the heap. Also, since pointers must be dereferenced (in some cases a chain of pointers must be followed), operations on formal private types will be slower than in replicated generics. In many cases, runtime calls are made to perform operations (e.g., equality tests, assignment, function return of formal privates, etc).
- **Formal Discrete and Integer Types**  
The only effect of formal discrete and integer types is that some arrays which would be allocated statically will become dynamic.
- **Formal Scalars of Mode IN OUT**  
In out objects cause little change in the amount of space allocated.

With all types an effort has been made to ensure that shared code generics behave in the same way as the same generic does when replicated, including raising the same errors at the same locations. The only exception to this is that in some cases with formal floating point types Constraint\_Error will be raised where Numeric\_Error was previously raised.

### 3.4. Code Views and the Predefined Subsystem

Prior to this release, code views could not be transferred between machines because the link names of the predefined units (e.g. system) were not the same on different machines. This has been corrected by placing the predefined units in a subsystem view called !Targets.Predefined. As a result, *new models are required which point to the new predefined units.*

This new organization permits worlds which were compiled with previous compilers to coexist on the same machine with worlds compiled with the new compiler. Note, however, that modules produced by one version of the compiler cannot be linked with modules produced by another version.

*Existing subsystem views which are to be compiled with the new compiler must have their models replaced.* The links in other worlds must be changed to point to the new predefined units. A tool is provided with this release which will demote units in existing worlds and update these worlds for the Rev7\_2 compiler - see the installation notes for more details.

### 3.5. Other New Features

#### 3.5.1. Pragma Assert

```
pragma ASSERT ([Predicate =>] boolean_expression
 [, [Class =>] integer_expression]);
```

When the pragma is encountered, `boolean_expression` is evaluated. If the resulting value is FALSE, the exception `System.Assertion_Error` is raised; otherwise, no further action is taken. If `System.Assertion_Error` is not defined, `Program_Error` is raised instead.

The `integer_expression` is ignored; for the moment, it is allowed only for compatibility with the R100C Ada code generator.

This pragma may appear wherever a statement or declaration would be legal.

#### 3.5.2. Pragma Collection\_Policy

The pragma `Collection_Policy` can be used to customize storage management. The complete syntax for this pragma is:

```
pragma COLLECTION_POLICY (Access_Type => access_type,
 Initial_Size => integer_expression
 [, Extensible => boolean_expression]
 [, Extension_Size => integer_expression]);
```

This pragma must appear in the same declarative region as the access type that it applies to, and the access type's full type declaration must occur before the pragma. This implies that, in order to use a private type for the `Access_Type` argument, the pragma must be given in the private part after the complete access-type declaration. At most one such pragma is allowed for a given access type.

The `Collection_Policy` pragma is ignored when:

- a `Storage_Size` clause is given for the access type; in this case, the `Storage_Size` takes precedence. This is because a statement of the form:

```
for X' Storage_Size use size;
```

is equivalent to:

```
pragma Collection_Policy (Access_Type => X,
 Initial_Size => size,
 Extensible => False);
```

- the pragma follows a forcing occurrence of the access type (LRM 15.1.6).

`Access_Type` and `Initial_Size` are required. `Extensible` and `Extension_Size` are optional. All arguments must be specified using named association, as shown in all examples in this pragma's discussion.

If Extensible is False, the collection is nonextensible. If Extensible is True, which is the default if not specified, then the collection will be extended by a certain number of storage units when it fills up, if it is possible to do so. The number of storage units in the extension size is chosen by the runtime system unless the Extension\_Size is specified.

Extension\_Size allows the user to specify the number of storage units to be allocated whenever the collection needs to be extended. This parameter is ignored if Extensible is False.

Initial\_Size gives the size in storage units of the collection created at the point of the type declaration. If no pragma Collection\_Policy is used, the initial size of the collection is determined by the runtime system. A nonpositive Initial\_Value on a nonextensible collection raises the Storage\_Error exception in the situations described below. Currently,

```
for X'Storage_Size use 0;
```

means that no collection should be created for X. The same effect is achieved with pragma Collection\_Policy by giving:

```
pragma Collection_Policy (Access_Type => X,
 Initial_Size => 0,
 Extensible => False);
```

In both of the above cases, an attempt to use an allocator will result in a Storage\_Error exception. However, you could also say:

```
pragma Collection_Policy (Access_Type => x,
 Initial_Size => 0,
 Extensible => True,
 Extension_Size => 100);
```

which means "do not create a collection until an allocation is performed." When the first allocator is executed, the collection is created with the extension size.

### 3.5.3. New Attribute: 'Homogeneous

Applicable to an access type; yields a Boolean value. The value returned is False if the designated type given in the access type declaration is an unconstrained array type or an unconstrained discriminated record type, and True otherwise.

If the attribute is True, then all objects in the collection must have the same size. The converse, however, is not true.

Note that the attribute is a property of the type, not of the subtype. Thus, for any access type T, T'Homogeneous = T'Base'Homogeneous.

For example:

```
type T1 is access String;
type T2 is new T1 (1 .. 10); -- T2'Homogeneous = False
```

At the implementation level, the attribute indicates whether or not constraint information needs to be stored with allocated objects.

### 3.5.4. Object Writer

The compiler now generates an object file directly. Previously, the compiler generated an assembly file and invoked the assembler to obtain an object file. This change results in code generation times that are approximately 25% faster. As a consequence, *the Cross\_Cg.Asm\_Source switch is now obsolete*. To obtain an assembly listing, use the Cross\_Cg.Listing switch.

### 3.5.5. Elaboration Checks

The CDF now supports the Suppress\_Elaboration\_Objects switch, which eliminates all elaboration objects and the code to test and set them. When this new switch is set to true, the compiler behaves as if the user had inserted:

```
pragma Suppress (Elaboration_Check, On => Snaggle);
```

for each subprogram 'snaggle' in each compilation unit in the library. This significantly reduces the amount of initialization code in Ada units.

Because the prelinker guarantees that spec/load subsystems are elaborated in layers, the compiler never generates elaboration checks on cross-subsystem calls to subprograms that are imported via a spec view. Due to this, it is not necessary that the Suppress\_Elaboration\_Objects switch have the same value in corresponding spec and load views.

Elaboration objects for generics are NOT affected by the Suppress\_Elaboration\_Objects switch.



## Software Installation Checklist

All products listed below have been installed at Rational prior to shipment (unless otherwise noted). Some products require steps that are not done at Rational (for example, downloading software to a workstation, optional steps such as installing an example project). Please take the time to review all of the documents included in this package. Steps done at Rational will be checked off in the installation note. Complete each product installation by executing the steps that have not been checked off in the product installation note.

Date : 23-Jun-93

Sales Order : SO 1009410

Machine Id : 462611

Site ID : 462611

Customer : TERMA

| Product #  | Product Name | Sessions | Release # | Notes     |
|------------|--------------|----------|-----------|-----------|
| 3000-00609 | ENV-400EXP   | 7        | D.12.7.3  | INSTALLED |
| 3000-00371 | M68020-BARE  | 4        | 6.3.3     | "         |
|            |              |          |           |           |
|            |              |          |           |           |
|            |              |          |           |           |
|            |              |          |           |           |
|            |              |          |           |           |
|            |              |          |           |           |

### Environment Disk Usage

| Volume | Used (kb) | DFS (kb) | Total (kb) |
|--------|-----------|----------|------------|
| 1      | 168,224   | 30,000   | 198,224    |
| 2      | 100,166   | 0        | 100,166    |
| 3      | 32,928    | 0        | 32,928     |
| 4      | 88,020    | 0        | 88,020     |

# SMSE Checklist SERIES\_400

✓ PRODUCT QA STAMP (On Packing List)

✓ PRODUCT REQUIREMENTS

| <input checked="" type="checkbox"/> <u>PART NUMBER</u> | <u>DESCRIPTION</u>                        |
|--------------------------------------------------------|-------------------------------------------|
| <input checked="" type="checkbox"/> 508-003234-005     | Exabyte Operations & Ordering Information |
| <input checked="" type="checkbox"/> 506-003274-005     | SMSE Checklist Disk_400                   |
| <input checked="" type="checkbox"/>                    | <u>TAPE MEDIA</u>                         |
| <input checked="" type="checkbox"/> 541-002873-001     | 8mm                                       |
| <input checked="" type="checkbox"/> 670-003271-001     | Environment Bundled SW Backup Tape        |
| <input checked="" type="checkbox"/> 680-003209-018     | Environment D_12_7_3 DFS Backup Tape      |
| <input checked="" type="checkbox"/> 505-003249-001     | Support Activity Report (SAR)             |

- OVER -

# Product Authorization Codes

## Machine ID : 462611

### PRODUCT SESSION AUTHORIZATION

SITE ID : 462611

Execute

```
Accept_Tokens (Product => [STRING-expression],
 Donation => [POSITIVE-expression],
 Resulting_Count => [POSITIVE-expression],
 Code => "",
 Authorization => "");
```

using the following values (NOTE: Values are case sensitive) :

| Product      | Donation | Resulting_Count | Code            | Authorization |
|--------------|----------|-----------------|-----------------|---------------|
| X Interface  | 7        | 7               | 780CB0551DDF37B | 1C6F16CE77    |
| Full Session | 7        | 7               | 6D1B74E2EAA496A | 1FCDE53B77    |

### PRODUCT AUTHORIZATION

Execute

```
"!Implementation".Product_Authorization.Register
(Product_Name => ">>Product Name<<",
 Authorization_Code => "",
 Expiration_Date => "");
```

using the following values (NOTE: Values are case sensitive) :

| Product_Name        | Authorization_Code | Expiration_Date |
|---------------------|--------------------|-----------------|
| Cmvc                | 11409CBFDAAF3B44   |                 |
| Cmvc.Source_Control | 100377E287906375   |                 |
| Ftp                 | 1602F441F807F865   |                 |
| Rpc                 | 3C21F796D27E177    |                 |
| Tcp/Ip              | 6846455ABFBD658    |                 |
| Telnet              | BCA525F4098AD1     |                 |
| Work_Orders         | 16144C9CB5785974   |                 |
| Lrm_Interface       | C51577822239E1     |                 |

SO => N/A

ORIGINATOR => KAM

COMMENT => TERMA

# Product Authorization Codes

## Machine ID : 462611

### PRODUCT SESSION AUTHORIZATION

SITE ID : 462611

Execute

```
Accept_Tokens (Product => [STRING-expression],
 Donation => [POSITIVE-expression],
 Resulting_Count => [POSITIVE-expression],
 Code => "",
 Authorization => "");
```

using the following values (NOTE: Values are case sensitive) :

| Product        | Donation | Resulting_Count | Code             | Authorization |
|----------------|----------|-----------------|------------------|---------------|
| Mc68020_Os2000 | 4        | 4               | 13FCA3FD2B13BEDE | 59CFB85144    |

### PRODUCT AUTHORIZATION

Execute

```
"!Implementation".Product_Authorization.Register
(Product_Name => ">>Product Name<<",
 Authorization_Code => "",
 Expiration_Date => "");
```

using the following values (NOTE: Values are case sensitive) :

| Product_Name   | Authorization_Code | Expiration_Date |
|----------------|--------------------|-----------------|
| Motorola_68k   | 67D99D18B2669B5    |                 |
| Mc68020_Os2000 | 15A7431C6D3ED840   |                 |

SO => N/A

ORIGINATOR => KAM

COMMENT => TERMA

# Exabyte Operations and Ordering Information

## 1. User/Operator Maintenance

In new system shipments or upgrades containing an Exabyte tape drive, Rational includes a blank 8mm tape cartridge and a 3-pass, approved cleaning kit.

The Exabyte unit requires low maintenance, however, it is dependent on the proper care and handling of the transport and magnetic tape. The cleaning procedures outlined in the Exabyte Cleaning Kit are brief and require only minutes of the operator's time, but cleaning must be done as explained in order to achieve continued reliability and low maintenance.

The User/Operator of the Exabyte tape drive is responsible for performing a periodic tape head/path cleaning. The cleaning of the tape head/path is accomplished with the Exabyte Cleaning Cartridge Kit. The cleaning kit contains the instructions and all items required to perform the tape head/path cleaning operation. The kit which Rational has included in your shipment is good for 3 passes. After the 3rd cleaning, the tape should be discarded. The recommended frequency for the cleaning is after 30 Gigabytes of data transfer or monthly, whichever occurs first.

### Reliability

To maximize reliability, please observe the points outlined in this section. Please note that failure to follow these recommendations or requirements, may result in voiding manufacturing and/or support warranties.

- Use only Exabyte's brand 8mm tape cartridge or the Sony, video 8, metal particle, data grade tape cartridge.
- Maintain proper environmental conditions for the tape drive and the media. The drive or media should not be operated or stored in areas where there is excessive heat or cold or in areas where temperatures may change at a rate of more than 10% (Fahrenheit) per hour. Also, the relative humidity should be between 20% and 80% non-condensing. (recommendation)
- Use only approved cleaning tapes. (Exabyte part no. 727113 or 727114) Unapproved video store cleaning tapes may damage the heads or transport. (requirement)
- Avoid using faulty or damaged cartridges/media. (requirement)

### Ordering Information

Orders for Exabyte-brand tapes and approved cleaning kits may be placed by calling Exabyte's sales outlet at 1-800-767-8273. (For outside U.S., 1-303-442-4333 or 1-303-447-7613)

| Part Number | Description                    |
|-------------|--------------------------------|
| -----       | -----                          |
| 180091      | Exatape, 15 Meter              |
| 180092      | Exatape, 54 Meter              |
| 180093      | Exatape, 112 Meter             |
| 727113      | Exabyte Cleaning Tape, 3-pass  |
| 727386      | Exabyte Cleaning Tape, 12-pass |

# SMSE Checklist

## Environment\_400EXP

☒ PRODUCT QA STAMP (On Packing List)

☒ SESSION AUTHORIZATION

| <input checked="" type="checkbox"/> <u>PART NUMBER</u> | <u>DESCRIPTION</u>                                                     |
|--------------------------------------------------------|------------------------------------------------------------------------|
| <input checked="" type="checkbox"/> 509-003267-001     | Environment_400EXP Authorization Codes<br>_____ Session Authorizations |
| <input checked="" type="checkbox"/> 505-003249-001     | Support Activity Report (SAR)                                          |

- OVER -

## PRODUCT REQUIREMENTS

| <u>PART NUMBER</u>       | <u>DESCRIPTION</u>                                                                                                                                                            |
|--------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ✓ 508-003207-007         | Rational Environment Release Information, Release D_12_7_3                                                                                                                    |
| ✓ 507-003219-012         | X_Interface Installation Procedure, Release10_10_1                                                                                                                            |
| ✓ 508-003219-006         | Rational X Interface Release Information, Release10_10_1                                                                                                                      |
| ✓ 507-003288-003         | Rational_Access Installation Procedure, Release1_0_1                                                                                                                          |
| ✓ 508-003288-001         | Rational Access Release Information, Release1_0_1                                                                                                                             |
| ✓ 507-003275-002         | RWI Installation Procedure, Release10_1_1                                                                                                                                     |
| ✓ 508-003275-003         | Rational Windows Interface Release Information, Release10_1_1                                                                                                                 |
| ✓ 690-003275-005         | RWI Release10_1_1D Diskette<br><input checked="" type="checkbox"/> 541-003258-001 3.5 Diskette<br><input checked="" type="checkbox"/> 541-003282-001 5.25 Diskette            |
| ✓ 507-003265-003         | Training Installation Procedure, Release1_1_1                                                                                                                                 |
|                          | <u>INSTALLATION ACTIVITY</u>                                                                                                                                                  |
| <input type="checkbox"/> | Full Installation in Field Required.                                                                                                                                          |
| <input type="checkbox"/> | Partial installation at Factory performed, Field installation of unchecked steps required, starting with step _____ of <i>Installation Procedure for Environment_400EXP</i> . |
| ✓                        | Complete installation performed at Factory.                                                                                                                                   |

# Product Authorization Codes

## Machine ID : 462611

### PRODUCT SESSION AUTHORIZATION

SITE ID : 462611

Execute

```
Accept_Tokens (Product => [STRING-expression],
 Donation => [POSITIVE-expression],
 Resulting_Count => [POSITIVE-expression],
 Code => "",
 Authorization => "");
```

using the following values (NOTE: Values are case sensitive) :

| Product      | Donation | Resulting_Count | Code            | Authorization |
|--------------|----------|-----------------|-----------------|---------------|
| X Interface  | 7        | 7               | 780CB0551DDF37B | 1C6F16CE77    |
| Full Session | 7        | 7               | 6D1B74E2EAA496A | 1FCDE53B77    |

### PRODUCT AUTHORIZATION

Execute

```
"!Implementation".Product_Authorization.Register
(Product_Name => ">>Product Name<<",
 Authorization_Code => "",
 Expiration_Date => "");
```

using the following values (NOTE: Values are case sensitive) :

| Product_Name        | Authorization_Code | Expiration_Date |
|---------------------|--------------------|-----------------|
| Cmvc                | 11409CBFDAAF3B44   |                 |
| Cmvc.Source_Control | 100377E287906375   |                 |
| Ftp                 | 1602F441F807F865   |                 |
| Rpc                 | 3C21F796D27E177    |                 |
| Tcp/Ip              | 6846455ABFBD658    |                 |
| Telnet              | BCA525F4098AD1     |                 |
| Work_Orders         | 16144C9CB5785974   |                 |
| Lrm_Interface       | C51577822239E1     |                 |

SO => N/A

ORIGINATOR => KAM

COMMENT => TERMA