IBM® Rational® DOORS

Using Rational DOORS for Rational Rose Interface

IBM Rational DOORS for Rational Rose Interface

Using Rational DOORS for Rational Rose Interface

Release 2.10.1

Before using this information, be sure to read the general information under the "Notices" chapter on page 27.

This edition applies to **IBM Rational DOORS for Rational Rose Interface**, **VERSION 2.10.1**, and to all subsequent releases and modifications until otherwise indicated in new editions.

Table of contents

1 About this manual

Welcome to IBM® Rational® DOORS® for Rational Rose® Interface.

Rational DOORS for Rational Rose Interface transfers data between Rational DOORS and Rational Rose. It lets you add traceability to your software design process.

This manual describes how to use version 2.10.1 of Rational DOORS for Rational Rose Interface. It assumes that you know how to use both Rational DOORS and Rational Rose.

Typographical conventions

The following typographical conventions are used in this manual:

Typeface or Symbol	Meaning
Bold	Important items, and items that you can select, including buttons and menus. For example: Click **Yes** to continue.
Italics	Book titles
`Courier`	Commands, files, and directories; computer output. For example: Edit your `.properties` file.
>	A menu choice. For example: Select **File > Open**. This means select the File menu, then select the Open command from it.

Related documentation

The following table describes where to find information in the documentation set:

For information on	See
How to use Rational DOORS	The Rational DOORS Information Center
How to use Rational Rose	The Rational Rose documentation set

For information on	See
How to install Rational DOORS for Rational Rose Interface	The Rational DOORS Information Center

2 Concepts

This chapter introduces and explains the following background concepts:

- About Rational DOORS for Rational Rose Interface
- About surrogate modules
- Creating traceability
- Keeping surrogate modules updated
- Developing model elements in Rational DOORS
- Rearranging surrogate module data

About Rational DOORS for Rational Rose Interface

Rational DOORS is a powerful tool for creating, structuring and managing complex information, such as the requirements and use case scenarios associated with software development projects.

Rational Rose is an object-oriented modeling tool that supports the analysis and design of software. Rational DOORS for Rational Rose Interface lets you transfer data between Rational DOORS and Rational Rose, allowing you to add traceability to your software design process.

With Rational DOORS for Rational Rose Interface, you can:

- Manage your software development projects by establishing traceability between requirements and software design.
- Assess how requirements impact your software design.
- Identify requirements that have not been addressed by the software design.
- Find design elements that are not justified by requirements.
- Find out which requirements are associated with a specific element of the software design, and vice versa.
- Verify and demonstrate that your design meets your customer requirements.

About surrogate modules

Rational DOORS for Rational Rose Interface lets you send elements in a Rational Rose model to a Rational DOORS project.

Rational DOORS for Rational Rose Interface creates a Rational DOORS formal **surrogate module** to store the Rational Rose data. The surrogate module is an anchor point in Rational DOORS that lets you link the requirements in Rational DOORS to the software elements in Rational Rose.

The surrogate module has the same name as the Rational Rose model.

For each element you send, Rational DOORS for Rational Rose Interface automatically creates an object in the surrogate module. The new object has the following Rational DOORS attributes:

Rational DOORS attribute	Value
Deleted in Rational Rose?	This is used to keep track of whether the element has been deleted from the Rational Rose model. When you first send the element to Rational DOORS, it is set to **False**. If you subsequently delete the element from the Rational Rose model, it is set to **True** the next time you update the data in Rational DOORS, so that you can see what it was linked to (see "Keeping surrogate modules updated," on page 5).
Documentation from Rational Rose	If you send the documentation associated with the element in the Rational Rose model, it is stored in this attribute.
External Documents from Rational Rose	The location of any attachments to the Rational Rose Model File.
Object Short Text	The name of the element in the Rational Rose model.
Object Text	The section and name of the element in the Rational Rose model.
Read only in Rational Rose	This attribute is **True** for read-only Rational Rose elements, and **False** for elements that can be modified.
Rational Rose Element Identifier	A unique identifier assigned by Rational Rose to the element in the Rational Rose model.
Rational Rose Model File	The location of the Rational Rose model.

Rational DOORS attribute	Value
Rational Rose Type	The type of the element in the Rational Rose model, for example **UseCase**, **Class** or **Category**.
Stereotype from Rational Rose	The stereotype of the element in the Rational Rose model.

The objects in the surrogate module are organized in a hierarchy based on the type of the model element.

All of the classes are grouped under the heading **Class**, all of the categories are grouped under the heading **Category**, and so forth. When you send class elements, you can also send the operations and attributes associated with the class. Each operation and attribute becomes a child of the class object.

Creating traceability

You use a surrogate module in the same way as all other Rational DOORS modules. You can link its objects to objects in other modules, you can create your own views and you can exploit the power of Rational DOORS traceability tools.

The surrogate module acts like a bridge between the Rational Rose model and the Rational DOORS project.

With Rational DOORS for Rational Rose Interface, you can apply standard Rational DOORS traceability functions, such as impact and trace analysis, to elements in Rational Rose models. For example, you can find out which requirements are impacted by a particular element or set of elements in your Rational Rose model, or you can find out if there are any requirements which are not linked to elements in your Rational Rose model.

Keeping surrogate modules updated

You use Rational DOORS to store your requirements, and Rational Rose to store your software model.

When you work on requirements, you use Rational DOORS. When you work on your software model, you use Rational Rose.

As you develop your software model in Rational Rose, you should periodically feed the changes you make in Rational Rose back to the surrogate module, to keep it up to date. To update, refer to "Updating the surrogate module," on page 17.

If you delete an element from the Rational Rose model, the next time you update the surrogate module, the **Deleted in Rose?** attribute of the corresponding Rational DOORS object is set to **True**. This lets you identify the requirements that are linked to deleted model elements so you can link them to valid model elements.

Note In earlier versions of Rational DOORS for Rational Rose Interface, we used the terms **synchronizing** and **performing consistency checks** to mean updating the surrogate module.

Developing model elements in Rational DOORS

You can use Rational DOORS to develop model elements and then send them to Rational Rose.

For example, you may decide to generate your use cases in Rational DOORS while you're developing the requirements for your project. You may find it helpful to create links between the requirements and use cases as you develop them.

Note Rational DOORS for Rational Rose Interface provides a template to help you develop use cases in Rational DOORS. To access the template, restore the module archive `use_case_template.dma` in your `IBM\Rational\DOORS\9.2\training` directory.

When you are ready to transfer the use cases to your Rational Rose model, use Rational DOORS for Rational Rose Interface to send the data from Rational DOORS to Rational Rose. Each object you send becomes an element in the target Rational Rose model.

The new element has the following properties:

Rational Rose property	Value
Name	The Rational DOORS object heading. If the object does not have a heading, the object identifier is used.
Documentation	The Rational DOORS object text.

Rational DOORS for Rational Rose Interface automatically sends the data straight back to Rational DOORS, creating a surrogate module in Rational DOORS which contains an object for each object you sent. It creates a link

between the object in the original Rational DOORS module and the corresponding object in the Rational DOORS surrogate module.

As you develop your software model in Rational Rose, you should periodically feed the changes you make in Rational Rose back to the surrogate module, to update it (see "Updating the surrogate module," on page 17).

Note Rational DOORS for Rational Rose Interface does not update the original Rational DOORS module; it only updates the surrogate module. If you want to keep the original module up to date, you must do so manually.

Rearranging surrogate module data

By default, the data in surrogate modules is arranged so that each class operation is located below the last sub-class that inherited the operation.

For example, there are two classes A and B, where:

- B is a sub-class of A
- A has two operations, OPR1 and OPR2
- B inherits both operations

In the Rational DOORS surrogate module, the operations are displayed below the B class heading.

When sending data from Rational DOORS to Rational Rose, or updating a surrogate module, you can rearrange the data so that each operation is located under its original base class.

3 Using Rational DOORS for Rational Rose Interface

This chapter describes how to use:

- Starting and stopping Rational DOORS for Rational Rose Interface sessions
- Sending data from Rational DOORS to Rational Rose
- Navigating from Rational DOORS to Rational Rose
- Linking directly to Rational DOORS Object IDs
- Sending data from Rational Rose to Rational DOORS
- Working with read-only Rational Rose elements
- Updating the surrogate module
- Navigating from Rational Rose to Rational DOORS
- Linking to items in a Rational Rose diagram
- Linking to items in a Rational Rose browser tree
- Enabling and disabling the rearrange data option

Starting and stopping Rational DOORS for Rational Rose Interface sessions

Before you can use Rational DOORS for Rational Rose Interface to transfer data or navigate between Rational DOORS and Rational Rose, you must start a Rational DOORS for Rational Rose Interface session. You can start a session on either Windows® or UNIX®.

Follow the steps below according you the platform you are using.

To start a Rational DOORS for Rational Rose Interface session on Windows:

1. Start Rational Rose.
2. Log in to Rational DOORS.

You are ready to start transferring data between Rational Rose and Rational DOORS.

To start a Rational DOORS for Rational Rose Interface session on UNIX:

1. Log in to Rational DOORS.

2. From Rational DOORS Explorer or from a module window, click **DOORSConnect > Rose > Start Session**.

   Rational DOORS for Rational Rose Interface starts Rational Rose, then opens up a communications channel between Rational DOORS and Rational Rose. This communications channel lets you transfer data between the currently selected folder or project in Rational DOORS and the model currently open in Rational Rose.

   Your Rational DOORS for Rational Rose Interface session is automatically ended when you exit from Rational DOORS. You can also end your session by clicking **DOORSConnect > Rose > Stop Session**, from Rational DOORS Explorer or a module window.

Note Ending a session shuts down the communications channel between Rational DOORS and Rational Rose. It does not shut down Rational DOORS or Rational Rose.

Sending data from Rational DOORS to Rational Rose

You can send data from one Rational DOORS module to more than one Rational Rose model. A single Rational DOORS project or folder can contain multiple surrogate modules. If the surrogate module does not already exist, you must have **create** access to its project or folder.

To send data from Rational DOORS to Rational Rose:

1. Using Rational Rose, open the target model.
2. Using Rational DOORS, open the module that contains the data you want to send to Rational Rose. If you have previously sent data from this Rational DOORS module to Rational Rose, or are using the `use_case_template.dma` file to create Rational Rose Items, go to Step 4.
3. Click **DOORSConnect > Rose > Setup Module**.

   In the source module, Rational DOORS for Rational Rose Interface:

   - Creates a new attribute called **Rose Type**.
   - Creates a new view, which is identical to the previous view except that it includes the new **Rose Type** column.

     The name of the new view is **DRL_viewname**, where **viewname** is the name of the previous view.

4. Using the new view of the module, set the **Rose Type** attribute of every object you want to send to Rational Rose. For example, for use cases, set the **Rose Type** attribute to **UseCase**.

The **Rose Type** value tells the application what type of element to create in the target Rational Rose model.

5. Using the new view of the module, click **DOORSConnect > Rose > Export Items to Rose**.

   The **Send to Rose** dialog box is displayed.

6. Click one of the **Send these objects** radio buttons to specify which objects you want to send to Rational Rose.

   These radio buttons are described in the following table:

Radio button	Description
Display set	Send all the objects in the current view.
Selection	Send only the objects that are currently selected.
Current object	Send only the current object.

   Note You can only send objects that have a value for their **Rose Type** attribute.

7. Use the drop-down list to select the link module you want to use for the links between the current (source) module and the surrogate module.

   Use the **link direction** radio button to specify the direction of the links. Select **in** if you want the links to go from the surrogate module to the current module, otherwise select **out**.

8. Click **OK**.

Rational DOORS for Rational Rose Interface:

- Sends the specified objects to the Rational Rose model you opened in Step 2. It displays a Rational DOORS message telling you how many objects were sent. Click **OK** to acknowledge the message.

  Each object becomes an element in the Rational Rose model. The object heading is used as the name of the element (if it doesn't have a heading, the Rational DOORS object identifier is used instead), and its object text is used as its documentation.

- Creates a surrogate Rational DOORS module for the Rational Rose model, if the surrogate module does not already exist.

  The surrogate module has the same name as the Rational Rose model. For example, if the Rational Rose model is called `test.mdl`, the surrogate

module is called `test.mdl`. If the name of the Rational Rose model includes any characters that are not allowed in Rational DOORS module names, these characters are omitted in the Rational DOORS module name. The surrogate module has the following views:

This view	Contains these columns
DRL Basic	ID (Object Identifier) Object Text (Name of Rational Rose element) Rose Type Stereotype
DRL Details	ID (Object Identifier) Object Text (Name of Rational Rose element) Rose Type Stereotype Documentation
DRL Deleted elements	ID (Object Identifier) Object Text (Name of Rational Rose element) Rose Type Stereotype Deleted in Rose?
No Parameters	Identifier Model (Name of Rational Rose element) Rose Type

- Sends the data back to the surrogate module (as if you had selected all the elements you just sent, in Rational Rose, and then sent them back to Rational DOORS). This creates a virtual link between the surrogate module and the model.
- Creates a link between each object in the source Rational DOORS module and the corresponding element in the surrogate module.
- Opens the **DRL Basic** view of the surrogate module, with a filter that shows only the data you just sent.

Note It is important that you do not change the name of the surrogate Rational DOORS module, the Rational Rose

model or any of the Rational DOORS attributes that Rational DOORS for Rational Rose Interface creates, such as the **Rose Type** attribute. If you change these names, subsequent updates will not work properly.

Navigating from Rational DOORS to Rational Rose

To navigate from an object in a Rational DOORS surrogate module to the corresponding model element in Rational Rose:

1. Using Rational DOORS, open the surrogate module.
2. Using Rational DOORS, select the object in the surrogate module that you want to navigate to in Rational Rose.
3. Using Rational DOORS, click **DOORSConnect > Rose > Find Item in Rose**.

   The **Class Specification for Cleaner** window is displayed in Rational Rose, for the corresponding element in Rational Rose.
4. Using Rational Rose, click **Browse > Select in Browser** to show the element highlighted in the Rational Rose Browser.

Linking directly to Rational DOORS Object IDs

When you export items that have a Rational Rose Type of either **Class** or **Actor** to Rational Rose, an **Object Identifier** attribute is created. This attribute stores the following information about the object in Rational DOORS with which the Rational Rose element is associated:

- The absolute path to the module containing the object
- The unique module identifier
- The Rational DOORS Object ID

An example **Object Identifier** is `/myproject/mymodule-00000010-SR2`, where `/myproject/mymodule` is the absolute path to the module, `00000010` is the unique identifier of the module and `SR2` is the Object ID of the object in the Rational DOORS module.

Note The unique identifier of a module is automatically generated by Rational DOORS when the module is created.

The Object Identifier is stored in the Rational DOORS surrogate modules and is displayed in all of the default views in a column titled Target Identifier. It is also stored as an attribute below the Rational Rose element.

You can associate the Rational Rose element with up to 50 Rational DOORS objects by adding the module path and the Rational DOORS Object ID to the **Documentation** field of the **Class Attribute Specification for Object Identifier** dialog. The next time you send data from Rational Rose to Rational DOORS the surrogate module is updated and links are created from the surrogate module to the object in the Rational DOORS module you specified in the Documentation field.

To associate a Rational Rose element with a Rational DOORS object:

1. In Rational Rose, double-click the Object Identifier.

   The **Class Attribute Specification for Object Identifier** dialog is displayed.

2. In the **Initial Value** field, type the path to the module, including the module name and the Rational DOORS Object ID of the object that you want to associate. Separate each entry with a comma.

   For example, add `,myproject/mymodule-SR4,myproject/mymodule-SR5` to the end of the text that is currently displayed in the Initial Value field.

3. Click OK.

4. Select **Tools > DOORS > Send to DOORS** or **Tools > DOORS > Update DOORS**.

   Links are created in Rational DOORS between the appropriate objects in Rational DOORS.

   In the above example, two new links are created between the appropriate object in the surrogate module and Object ID **SR4** in **mymodule**, and Object ID **SR5** in **mymodule**.

Sending data from Rational Rose to Rational DOORS

To send data from Rational Rose to Rational DOORS:

1. Using Rational Rose, open the model that contains the data you want to send.

2. Select **Tools > DOORS > Send to DOORS**.

3. Use the radio buttons to specify which elements you want to send to Rational DOORS.

The radio buttons are described in the following table:

Radio button	Description
Current diagram	Only send elements in the current Rational Rose Diagram.
Model	Send all elements in the current model.
Selected elements	Only send the selected elements in the current diagram or model.
Elements of type	Send all elements of the specified type in the current diagram or model. Use the drop-down list box to select the type.
All elements	Send all the elements in the current diagram or model.

Note If you have a read-only element that you want to send to Rational DOORS, make it writable when you send it, then reset its state to read-only. This ensures that it is automatically updated in the future. For more information, see "Working with read-only Rational Rose elements," on page 17.

In addition, select the following Send options:

- To send the documentation property with each element, check the **Include documentation** box.
- To send elements from the current diagram with associations, check the **Include associations** box.
- To send elements of type **Class**, use the **Include Rose class properties** check boxes to specify whether you want to send the operations and attributes associated with the classes.

4. Click **OK**.

   If this is the first time that the Rational Rose model has been sent to the Rational DOORS module, the **Locate Surrogate Module** dialog box is displayed.

5. Select the folder or project to which you want to send the Rational Rose model. If you are sending data for the first time, you need to have create access to this project or folder.

6. Click **OK**.

   Rational DOORS for Rational Rose Interface then:

   - Creates a surrogate Rational DOORS module for the Rational Rose model, if the module does not already exist.

     The surrogate module has the same name as the Rational Rose model. For example, if the Rational Rose model is called `test.mdl`, the surrogate module is called `test.mdl`. If the name of the Rational Rose model includes any characters that are not allowed in Rational DOORS module names, these characters are ignored and do not appear in the Rational DOORS module name. The surrogate module has the following views:

This view	Contains these columns
DRL Basic	ID (Object Identifier) Object Text (Name of Rational Rose element) Rose Type Stereotype
DRL Details	ID (Object Identifier) Object Text (Name of Rational Rose element) Rose Type Stereotype Documentation
DRL Deleted elements	ID (Object Identifier) Object Text (Name of Rational Rose element) Rose Type Stereotype Deleted in Rose?
No Parameters	Identifier Model (Name of Rational Rose element) Rose Type

   - Sends the data to the surrogate module, creating a new object for each item sent.

- Displays a Rational DOORS message listing how many surrogate objects were created. Click **OK** to acknowledge the message.
- Displays a message asking whether you want to rearrange the data so that each operation is located under its original base class in the Rational DOORS surrogate module. Click **Yes** or **No** as appropriate.
- Opens the **DRL Basic** view of the surrogate module, with a filter showing only the data you just sent.

Note Do not change the name of the surrogate Rational DOORS module, the Rational Rose model or any of the Rational DOORS attributes that Rational DOORS for Rational Rose Interface creates, such as the **Rose Type** attribute. If you change these names, subsequent updates will not work properly.

Working with read-only Rational Rose elements

If you have a read-only element that you want to send to Rational DOORS, and there is a possibility that the model element might change in the future, make it writable when you send it, then reset it to read only. This ensures that it is automatically updated in the future, even if it is read only at the time of the update.

If you send a read-only Rational Rose element to Rational DOORS, Rational DOORS for Rational Rose Interface cannot mark the element in Rational Rose as sent because the element is not writable. As a result, in subsequent updates, the corresponding surrogate object will not be automatically updated.

To update the surrogate object, manually select the object in Rational Rose and send it to Rational DOORS again. You can also send other elements to Rational DOORS at the same time.

If the model element is now writable in Rational Rose, when you send it to Rational DOORS, Rational DOORS for Rational Rose Interface automatically marks it in Rational Rose as sent. All future updates will now automatically update the corresponding surrogate object in Rational DOORS, together with any documentation and class properties that were sent with the writable element. Future updates now work even if the element is read-only in Rational Rose at the time of the update.

Updating the surrogate module

To update a surrogate module to feed changes made in Rational Rose back to Rational DOORS:

1. Using Rational Rose, open the appropriate model.
2. Using Rational Rose, click **Tools > DOORS > Update DOORS**.

   Rational DOORS for Rational Rose Interface:

   - Updates all the data previously sent to the surrogate module to reflect changes made in the Rational Rose model, provided that the data wasn't read-only in Rational Rose when it was sent to Rational DOORS (see "Working with read-only Rational Rose elements," on page 17).

     If an element has been deleted from the Rational Rose model since the surrogate module was last updated, the **Deleted in Rose?** attribute of the corresponding surrogate object is set to **True**.
   - Displays a Rational DOORS message telling you how many modifications were made to surrogate objects, and how many surrogate objects have been deleted from the Rational Rose model since the last update. For example, if you've changed the name and documentation of a Rational Rose element since the last update, the message says that two modifications were made. Click **OK** to acknowledge the message.
   - Displays a message asking whether you want to rearrange the data so that each operation is located under its original base class in the Rational DOORS surrogate module. Click **Yes** or **No** as appropriate.
   - Opens the **DRL Basic** view of the surrogate module, with a filter showing only the data that was just updated.

Navigating from Rational Rose to Rational DOORS

To navigate from elements in Rational Rose to the corresponding objects in the Rational DOORS surrogate module:

1. Using Rational Rose, open the appropriate Rational Rose model and select the elements you want to find in Rational DOORS.
2. Using Rational Rose, click **Tools > DOORS > Find in DOORS**.

   Rational DOORS for Rational Rose Interface opens the surrogate Rational DOORS module, with a filter showing the corresponding Rational DOORS objects.

Linking to items in a Rational Rose diagram

To link to one or more items in a Rational Rose diagram:

1. Select one or more items from your Rational Rose diagram.

2. Select a Rational DOORS requirement object in Rational DOORS.
3. Using Rational DOORS, click **DOORSConnect > Rose > Link to item(s) in Rose Diagram**.

   Your selected Rational Rose items are imported into the local Rational Rose surrogate module, and a link is created between the Rational DOORS object and the imported objects.

Linking to items in a Rational Rose browser tree

To link to one or more items in a Rational Rose browser tree:

1. Select one or more items from your Rational Rose component browser tree.
2. Select a Rational DOORS requirement object in Rational DOORS.
3. Using Rational DOORS, click **DOORSConnect > Rose > Link to item(s) in Rose browser tree**.

   Your selected Rational Rose items are imported into the local Rational Rose surrogate module, and a link is created between the Rational DOORS object and the imported objects.

Enabling and disabling the rearrange data option

Processing surrogate modules

By default, the data in surrogate modules is arranged so that each class operation is located below the last sub class that inherited the operation.

When you either send data from Rational Rose to Rational DOORS or update a surrogate module, you see a message that asks if you want to rearrange the data so that each operation is located under its original base class in the Rational DOORS surrogate module. This message is displayed after the data has been sent to Rational DOORS and you have closed the DRL metrics window.

For example, say you have two classes A and B, where:

- B is a sub class of A.
- A has two operations OPR1 and OPR2.
- B inherits OPR1.

In the Rational DOORS surrogate module, by default the operation OPR1 is displayed below the B class heading as A.OPR1, as follows:

1. Class

LogicalView::A	Class
LogicalView::A::OPR2	Operation
LogicalView::B	Class
LogicalView::A::OPR1	Operation

If you rearrange the data, LogicalView::A::OPR1 is moved to below the A class heading, as follows:

1. Class

LogicalView::A	Class
LogicalView::A::OPR2	Operation
LogicalView::A::OPR1	Operation
LogicalView::B	Class

Disabling the rearrange data option

By default, the data in surrogate modules is arranged so that each class operation is located below the last sub-class that inherited the operation.

When sending data from Rational DOORS to Rational Rose, or updating a surrogate module, you can choose to rearrange the data, so that each operation is located under its original base class.

If you want to disable the rearrange data option, edit the file `drlpostprocessing.inc` in the folder `lib\dxl\addins\doorsconnect\rose` in the Rational DOORS home directory.

Comment out the following lines by putting two forward slash (`//`) characters at the start of each line:

```
makeColumns (m)
//setup "no parameters" view which removes
//operation parameters
rearrangeOperationsAndAttributes (m)
```

4 Troubleshooting

This chapter describes how to troubleshoot:

- If your Rational DOORS menu options are missing in Rational Rose
- If you get a "suite objects.dll not found" error
- If associations are not updated
- If Rational Rose does not exit properly
- If you see two DOORSConnect menus in Rational DOORS

If your Rational DOORS menu options are missing in Rational Rose

If you re-install Rational Rose after you have installed Rational DOORS for Rational Rose Interface, the Rational DOORS menu options will disappear from your Rational Rose Tools menu.

To fix this, re-install Rational DOORS for Rational Rose Interface.

If you get a "suite objects.dll not found" error

When you are installing Rational Rose on Windows platforms, you may get a message saying that you must update the PATH variable. If you ignore that message, when you click **Start Session** to start a Rational DOORS for Rational Rose Interface session, you will get a "suite objects.dll not found" error.

To fix this problem, update the PATH variable. Edit your `autoexec.bat` file to add the `\rational\common` folder to your PATH. For example, add: `"c:\program files\rational\common"`.

If associations are not updated

When you update the surrogate module in Rational DOORS, associations are not automatically updated when you use the **Update Rational DOORS** menu option in Rational Rose.

Use the **Send to DOORS** menu option in Rational Rose instead. You can do one of the following:

- Explicitly select the associations in Rational Rose, and send **selected elements** to Rational DOORS.

- Send the **current diagram** to Rational DOORS; this updates everything, including the associations in the current diagram.

If Rational Rose does not exit properly

If you allow Rational DOORS to start the Rational Rose session then you must exit Rational DOORS before Rational Rose can be exited in the normal fashion.

If you see two DOORSConnect menus in Rational DOORS

When you upgrade Rational DOORS for Rational Rose Interface, a second **DOORSConnect** menu might sometimes appear in Rational DOORS.

Follow the steps below to fix this problem:

To remove one, do the following:

1. Navigate to the `$DOORSHOME\lib\dxl\addins` directory.
2. Open the `addins.idx` file in a text editor.

   An example of this file is shown below:

   ```
   userU _ User
   doorsconnectD _ DOORSConnect
   doorsconnectD _ DOORSConnect
   ```

3. Delete the third line and save the file.
4. Restart Rational DOORS.

5 Contacting support

This chapter contains the following topics:

- Contacting IBM Rational Software Support
- Prerequisites
- Submitting problems
- Other information

Contacting IBM Rational Software Support

If the self-help resources have not provided a resolution to your problem, you can contact IBM Rational Software Support for assistance in resolving product issues.

Note If you are a heritage Telelogic customer, you can go to http://support.telelogic.com/toolbar and download the IBM Rational Telelogic Software Support browser toolbar. This toolbar helps simplify the transition to the IBM Rational Telelogic product online resources. Also, a single reference site for all IBM Rational Telelogic support resources is located at http://www.ibm.com/software/rational/support/telelogic/

Prerequisites

To submit your problem to IBM Rational Software Support, you must have an active Passport Advantage® software maintenance agreement. Passport Advantage is the IBM comprehensive software licensing and software maintenance (product upgrades and technical support) offering. You can enroll online in Passport Advantage from http://www.ibm.com/software/lotus/passportadvantage/howtoenroll.html.

- To learn more about Passport Advantage, visit the Passport Advantage FAQs at http://www.ibm.com/software/lotus/passportadvantage/brochures_faqs_quickguides.html.
- For further assistance, contact your IBM representative.

To submit your problem online (from the IBM Web site) to IBM Rational Software Support, you must additionally:

- Be a registered user on the IBM Rational Software Support Web site. For details about registering, go to http://www-01.ibm.com/software/support/.
- Be listed as an authorized caller in the service request tool.

Submitting problems

To submit your problem to IBM Rational Software Support:

1. Determine the business impact of your problem. When you report a problem to IBM, you are asked to supply a severity level. Therefore, you need to understand and assess the business impact of the problem that you are reporting.

   Use the following table to determine the severity level.

Severity	Description
1	The problem has a *critical* business impact: You are unable to use the program, resulting in a critical impact on operations. This condition requires an immediate solution.
2	This problem has a *significant* business impact: The program is usable, but it is severely limited.
3	The problem has *some* business impact: The program is usable, but less significant features (not critical to operations) are unavailable.
4	The problem has *minimal* business impact: The problem causes little impact on operations or a reasonable circumvention to the problem was implemented.

2. Describe your problem and gather background information. When describing a problem to IBM, be as specific as possible. Include all relevant background information so that IBM Rational Software Support specialists can help you solve the problem efficiently. To save time, know the answers to these questions:

   - What software versions were you running when the problem occurred?

     To determine the exact product name and version, use the option applicable to you:

- Start the IBM Installation Manager and click **File** > **View Installed Packages**. Expand a package group and select a package to see the package name and version number.
- Start your product, and click **Help** > **About** to see the offering name and version number.

- What is your operating system and version number (including any service packs or patches)?
- Do you have logs, traces, and messages that are related to the problem symptoms?
- Can you recreate the problem? If so, what steps do you perform to recreate the problem?
- Did you make any changes to the system? For example, did you make changes to the hardware, operating system, networking software, or other system components?
- Are you currently using a workaround for the problem? If so, be prepared to describe the workaround when you report the problem.

3. Submit your problem to IBM Rational Software Support. You can submit your problem to IBM Rational Software Support in the following ways:

   - **Online**: Go to the IBM Rational Software Support Web site at https://www.ibm.com/software/rational/support/ and in the Rational support task navigator, click **Open Service Request.** Select the electronic problem reporting tool, and open a Problem Management Record (PMR), describing the problem accurately in your own words.

     For more information about opening a service request, go to http://www.ibm.com/software/support/help.html.

     You can also open an online service request using the IBM Support Assistant. For more information, go to http://www-01.ibm.com/software/support/isa/faq.html.

   - **By phone**: For the phone number to call in your country or region, go to the IBM directory of worldwide contacts at http://www.ibm.com/planetwide/ and click the name of your country or geographic region.
   - **Through your IBM Representative**: If you cannot access IBM Rational Software Support online or by phone, contact your IBM Representative. If necessary, your IBM Representative can open a service request for you. You can find complete contact information for each country at http://www.ibm.com/planetwide/.

If the problem you submit is for a software defect or for missing or inaccurate documentation, IBM Rational Software Support creates an Authorized Program Analysis Report (APAR). The APAR describes the problem in detail. Whenever possible, IBM Rational Software Support provides a workaround that you can implement until the APAR is resolved and a fix is delivered. IBM publishes resolved APARs on the IBM Rational Software Support Web site daily, so that other users who experience the same problem can benefit from the same resolution.

Other information

For Rational software product news, events, and other information, visit the IBM Rational Software Web site on http://www.ibm.com/software/rational/.

6 Notices

This information was developed for products and services offered in the U.S.A. IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send written license inquiries to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send written inquiries to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:
INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND,

EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions. Therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

Intellectual Property Dept. for Rational Software
IBM Corporation
1 Rogers Street
Cambridge, Massachusetts 02142
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results

may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Additional legal notices are described in the legal_information.html file that is included in your software installation.

Trademarks

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at www.ibm.com/legal/copytrade.html.

Windows is a trademark of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product or service names may be trademarks or service marks of others.

Index