



**User Guide**

*IBM Rational*  
*System Architect User Guide*  
*Release 11.3*

Before using this information, read the “Notices” in the Appendix, on page 11-2.

This edition applies to IBM® Rational® System Architect®, version 11.3 and to all subsequent releases and modifications until otherwise indicated in new editions.

© Copyright IBM Corporation 1986, 2009

US Government Users Restricted Rights—Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.



---

# Table of Contents

<b>Table of Contents</b> .....	<b>I</b>
<b>Introduction to Rational System Architect</b> .....	<b>1-1</b>
<i>Rational System Architect</i> Components and Features .....	1-2
Methodologies Supported in Rational System Architect .....	1-8
<b>Working with Rational System Architect</b> .....	<b>2-1</b>
Getting Started with Rational System Architect .....	2-2
Using ‘Example’ Project Encyclopedias .....	2-3
Creating/Opening a Project Encyclopedia.....	2-5
Selecting Available Diagrams and Target Languages .....	2-8
Configuration Dialog Options .....	2-10
Using the Advanced Configuration Dialog .....	2-16
Creating a Diagram.....	2-19
Creating or Modifying a Definition .....	2-21
Using the Toolbars .....	2-25
Customizing the Toolbars .....	2-27
Exiting from Rational System Architect .....	2-29
<b>Working with the Explorer</b> .....	<b>3-1</b>
The Explorer’s Functions .....	3-2
Viewing Items in the Explorer.....	3-3
The Properties Option .....	3-4
Docking/Undocking the Explorer .....	3-5
Diagrams – Creating, Editing, Deleting through the Explorer.....	3-6
Definitions – Creating, Editing, Deleting through the Explorer .....	3-8
Permanently Deleting a Definition from the Encyclopedia .....	3-10
<b>General Drawing Techniques</b> .....	<b>4-1</b>
Drawing in Rational System Architect .....	4-2
Drawing Symbols on a Diagram.....	4-3
Naming and Defining Symbols .....	4-5
Drawing Lines .....	4-6
Drawing Lines in an Entity Relation Diagram .....	4-12
Diagram Settings .....	4-15
Moving Symbols and Editing a Diagram.....	4-17
Undo Command .....	4-20

Fonts .....	4-21
Format File .....	4-24
Copying Diagrams .....	4-32
Keyboard Accelerators .....	4-34
<b>Working with Definitions.....</b>	<b>5-1</b>
What are Definitions in <i>Rational System Architect</i> ? .....	5-2
Symbols and Definitions.....	5-3
Browse, Select, and Drag .....	5-5
Using Grids .....	5-6
Handling Data Definitions.....	5-8
Attributes in Entities.....	5-9
Data Elements and Data Structures .....	5-10
Using Data Domains.....	5-12
Text, Descriptions, and Comments.....	5-17
Spell Check .....	5-19
Importing and Exporting Definitions.....	5-20
CSV and Text Import Export .....	5-21
Using the Clipboard .....	5-23
Importing/Exporting Via XML .....	5-24
<b>The Matrix Editor .....</b>	<b>6-1</b>
The Matrix Editor.....	6-2
“X” in Cell Matrix.....	6-5
Text in Cell Matrices.....	6-8
Multi-dimensional Matrices.....	6-12
Mirrored Matrices .....	6-16
Creating Matrices .....	6-19
<b>Requirements Tracking.....</b>	<b>7-1</b>
Handling Requirements in Rational System Architect.....	7-2
Built-in Requirements.....	7-3
Customizing Requirements.....	7-4
Adding Requirements to Definitions .....	7-6
Built-in Tracking Facility .....	7-7
Creating and Attaching a Requirement Specification.....	7-8
How to Build Requirements .....	7-12
<b>Leveling Diagrams through Parent/Child Links.....</b>	<b>8-1</b>
Linking Parent and Child Diagrams .....	8-2
Data Flow Organization Techniques .....	8-8
Data Store Leveling .....	8-11
<b>Reporting and Documenting System.....</b>	<b>9-1</b>

Internal Reporting System .....	9-2
Microsoft Word Reports.....	9-5
HTML Generator.....	9-6
<b>IBM Support .....</b>	<b>10-1</b>
Contacting IBM Rational Software Support.....	10-2
<b>Appendix: .....</b>	<b>11-1</b>
Notices.....	11-2
Trademarks.....	11-5
Copyright acknowledgements.....	11-6
<b>Index .....</b>	<b>iv</b>

# 1

---

## ***Introduction to Rational System Architect***

### **Introduction**

Welcome to IBM® Rational® System Architect®, the World's Leading Enterprise Modeling Tool. Rational System Architect provides extensive capabilities to perform business process modeling, UML modeling, relational data modeling, and structured analysis and design. These modeling capabilities are performed within a real-time, multi-user environment that is based on Rational System Architect's repository with customizable metamodel.

This chapter introduces you to *Rational System Architect* – providing an overview of its features and the modeling methodologies it supports.

<b>Topics in this chapter</b>	<b>Page</b>
Product Components and Features	1-2
Methodologies Supported in Rational System Architect	1-9



---

## ***Rational System Architect*** **Components and Features**

*Rational System Architect* is comprised of a rich set of components that enable the capture, design, modeling, and creation of enterprise systems. All design information is stored in a multi-user repository called the encyclopedia. An encyclopedia is created as a database in either SQL Server 2000, SQL Server 2005, Microsoft Server Desktop Engine (MSDE), SQL Express, Oracle9i or Oracle 10g.

### **Comprehensive Support for All Major Paradigms of Modeling**

Rational System Architect is the only tool on the market to provide integrated support for all four areas of analysis and design – business modeling, object- and component-based modeling, relational data modeling, and structured analysis and design.

Business process modeling support includes extensive coverage of the Zachman Framework, the IDEF methodology, and links to third-party simulation tools. Object- and component-based modeling is provided through comprehensive support of the UML notation, with forward and reverse engineering of multiple languages. Market-proven data modeling capabilities include Entity Relation models with subject-areas, separate physical models, schema generation, and reverse data engineering. Structured analysis and design is provided through coverage of the Gane & Sarson, Yourdon/DeMarco, Ward & Mellor, and SSADM methodologies. You may also purchase the following options to Rational System Architect:

- A Simulation option provides simulation of Business Process Charts, IDEF3 Process Flow, and BPMN diagrams.
- A Rational System Architect for IBM Rational DOORS®® Interface links Rational System Architect with IBM Rational's market-leading requirements management tool, enabling you to track requirements against design models.
- A Rational System Architect DoDAF option provides comprehensive support for the United States Government's Department of Defense Architecture Framework.

**Graphical  
Interface and  
Drawing  
Workspace**

Rational System Architect provides a drawing workspace to graphically build the models representing the business system, application, or database design being modeled. It provides many options for drawing, displaying, and viewing items in the workspace. You may also print diagrams, and automatically create certain types of models using Rational System Architect's various reverse engineering capabilities.

**The Explorer**

Rational System Architect's Explorer is a multi-purpose navigation interface that displays, in a hierarchal tree, encyclopedia diagrams and definitions. It opens automatically when you open the product. You can view diagrams and definitions, open them, edit them, make them read-only, or delete them through the Explorer. By using the filter option, you can choose which definition and diagrams type you want to display.

**Frameworks**

Rational System Architect provides a Framework Manager that enables users to view and access the models and artifacts they have developed in an encyclopedia through a framework interface. Each cell of the Framework Manager can be opened to view a filtered browser list of all diagrams and definitions in the encyclopedia that pertain to that cell of the framework. Users may use predefined, industry accepted, frameworks such as the Zachman Framework or the US Department of Defense Architecture Framework DoDAF that are provided by IBM, or build their own framework browser to support their own custom framework.

**Diagrams and  
Drawing  
Workspace**

Every diagram type has its own Toolbox, with the symbols appropriate for the methodology that the diagram supports. You use the cursor and mouse to select one of the tools, or symbols, which you then drop onto the diagram drawing area. You can have up to approximately 1850 symbols on any one diagram; you can expand the default drawing area of the diagram up to the equivalent of 12.5 feet square.<sup>1</sup>

**Definition  
Dialogs**

All definitions in the encyclopedia can be edited through Definition dialogs. The properties representing each type of definition can be tailored using Rational System Architect's extensible metamodel facility. See the on-line help or the *Extensibility Guide* for details.

---

<sup>1</sup> Look for MaxDrawArea in the on-line help for information on how to increase the drawing area.

**Matrix Editors**

Rational System Architect provides a suite of Matrix Editors that can be used to enter information on the models before a single diagram is drawn. By attacking analysis in this way, you concentrate on a wide view of the problem, and the dependencies of information, before delving into more detailed analysis and design.

Matrix Editors are provided for all types of modeling in Rational System Architect, including Business Enterprise and IDEF business modeling. Users may also create their own matrices and run reports against them.

**Multi-User  
Repository with  
Extensible  
Metamodel**

The repository stores the definitions of the components that make up a project. You can give some kind of information to each of the symbols on every diagram. You can also include explanations about non-graphical components: data elements, data structures, attributes, requirements, test plans, business objects, and a number of other non-graphical objects. Every graphic and non-graphic object in Rational System Architect has at least the property "Description," in which you can enter some informative text about the reason that object is in the project.

Some graphic objects, like entities and tables have a large number of properties. In some cases, the properties change. For example, if you're designing an Oracle database, you'll have different properties than if the system is a design of an Access database.

**Customizing the  
Repository  
Metamodel**

In Rational System Architect, definitions, diagrams and symbols are defined with properties. These properties are sometimes referred to as metadata. The property system is controlled by two files – SAPROPS.CFG and USRPROPS.TXT. These files are housed in the FILES table of a Rational System Architect encyclopedia. They may be accessed by exporting/importing them from/to the encyclopedia via the **Tools, Customize User Properties** command. Their purpose is to specify the list of properties associated with each diagram, symbol, and definition type in Rational System Architect.

SAPROPS.CFG contains properties specified by IBM to define the default metamodel of an encyclopedia. USRPROPS.TXT is shipped as an empty file, designed as a place for users to add properties for their modeling needs. When an encyclopedia is opened, the software first parses SAPROPS.CFG, and then USRPROPS.TXT. Any

properties in USRPROPS.TXT override similar ones in SAPROPS.CFG.

Users may make simple or sophisticated changes to an encyclopedia's metamodel in USRPROPS.TXT. These changes are stored in USRPROPS.TXT, and can be passed around in an organization, enabling creation of a corporate standard for definition properties. Just as important, the USRPROPS.TXT file is not overridden by new versions or installations of Rational System Architect.

Please see the on-line help and the USRPROPS Extensibility Guide for more details.

### **Importing and Exporting Information**

Rational System Architect provides many facilities to import and export information from and to its repository:

- To begin, there is a native import/export mechanism, through which you can import/export definitions in comma separated value (csv) and text format.
- There is also an XML import/export capability, wherein you export/import information to an XML instance document, that conforms to a Rational System Architect Document Type Definition (DTD), saxml.dtd (housed in <C>:\Program Files\IBM\Rational\System Architect Suite\11.3\System Architect\saxml.dtd).
- Rational System Architect provides native support for VBA, with a published object model. This is described below.
- Rational System Architect has an XMI add-in option that enables you to import/export UML information via XMI. You may interface with other UML modeling tools in this fashion.
- Rational System Architect provides a merge/extract capability to selectively exchange information between encyclopedias.
- Code generation and reversal of languages enables export or import of class diagram information.

- Schema generation and reverse data engineering enables export or import of data modeling information.

**Built-in  
Extensibility  
Through  
Microsoft VBA**

Built-in extensibility through Microsoft Visual Basic for Applications enables advanced Rational System Architect users to create scripts that run against Rational System Architect. The VBA scripts can be made to run when a user invokes them, or automatically after certain events in Rational System Architect itself (ie, opening an Encyclopedia, or saving a definition). The Microsoft VBA interactive development environment (IDE) is provided within Rational System Architect for creation or editing of VBA scripts. Please see the on-line help for more information.

**Reporting and  
Documentation**

Through the reporting and documenting system you can generate a variety of project monitoring and management reports on your designs. Rational System Architect provides three systems to generate reports and documentation:

**Reporting  
System**

Over 150 predefined reports are provided. Users can build their own using a report-building graphical user interface.

**Microsoft Word  
Reports**

A number of predefined Word templates are loaded into your Word Template directory when Rational System Architect is installed. To run the reports, you open a new or existing Word document, select one of Rational System Architect's predefined templates, and then run a variety of rich, nicely formatted Word template reports against the open Rational System Architect encyclopedia. The Word report templates themselves can be tailored using Word's Visual Basic for Applications.

**HTML Generator**

A built-in HTML generator enables automatic creation of HTML-formatted reports on some or all diagrams in a project encyclopedia, with context-sensitive pictures of each diagram. The reports are automatically opened in your browser of choice.

## **Screen Painter**

You can use the Screen Painter facility to create Windows-style graphical user interface (GUI) windows and menus. Screen Painter provides three diagram types: Graphic Screen, Character Screen, and Menu diagrams. The Graphic Screen and Menu diagrams come with toolboxes that include standard Windows controls. You can also drag-and-drop previously defined objects, such as entities, data elements, and data structures onto a Graphic Screen – those definitions will automatically create controls. You can copy existing dialogs and menus. From the created Graphic Screens and menus you can generate .DLG, .MNU, and .H files for Windows projects. Or you can create Visual Basic .FRM files.

Character Screen diagrams provide for creation of character screens for COBOL applications. Rational System Architect generates COBOL .WKS and .SCS files from Character Screen diagrams.

---

## Methodologies Supported in Rational System Architect

### **Business Process Modeling**

Rational System Architect provides extensive business process modeling through a variety of techniques, most notable of which are integrated support for Enterprise Business Modeling and also the IDEF methodology.

### **Activity Based Costing Mechanisms**

Business process modeling activity provides based costing mechanisms.

### **Simulation**

Business process modeling diagrams provide links to an external simulation product – Witness.

### **UML Modeling**

Rational System Architect provides object-oriented and component-based modeling using the Unified Modeling Language (UML). UML is the current, yet still evolving, standard for object-oriented analysis and design of systems.

### **Code Generation**

Code generation is provided from object-oriented class diagrams, such as the UML Class diagram type. Code generation enables you to automatically translate a class diagram design to the following languages: C++, Java, Visual Basic, CORBA IDL, and JavaScript.

### **Reverse Engineering of C++ or Java**

Existing code in C++ and/or Java can also be reverse engineered into Rational System Architect, to automatically create a class diagram.

### **Relational Data Modeling**

Rational System Architect provides comprehensive data modeling that includes conceptual data modeling, logical data modeling that is provided through models and synchronized subject areas, and separate physical models.

### **Schema Generation**

The Rational System Architect Schema Generator creates data definition language (DDL) statements for a wide variety of SQL and non-SQL DBMS products, or links directly to an existing database through ODBC drivers. It also creates data definitions for inclusion into COBOL and C language programs. Schema is generated from a Physical Data Model. The **DB Schema Generator** menu is



under the **Tools** menu and is only listed as an option when a Physical Data Model diagram is in the active window.

**DB Reverse Engineering**

The Rational System Architect DB Reverse Engineer reverse engineers from a wide variety of SQL schema and databases into existing or new Rational System Architect encyclopedias, providing a relatively easy way to upgrade legacy systems. Output from RDE includes dictionary definitions and physical data models. It also creates diagrams from Windows-based .DLG and .MNU files. The **DB Reverse Engineer** command is under the **Tools** menu.

**DB Synchronize™**

**DB Synchronize™** is a Rational System Architect tool that helps data modelers effectively manage physical models and database schema. Using **DB Synchronize™**, you can compare and selectively synchronize a physical data model with a DBMS schema. The DBMS's supported at the time of this manual's printing are SQL Server 7, SQL Server 2000, SQL Server 2005, Teradata, DB2/UDB v8, Oracle 8x, and beyond. You may also compare and synchronize two physical data models, or two databases.

**Structured Analysis and Design**

Rational System Architect provides a rich set of diagrams to support traditional structured analysis and design techniques. Supported methodologies include Gane & Sarson, Yourdon/DeMarco, Ward & Mellor, SSADM, and Information Engineering (IE). Support includes methodology-specific data flow diagrams that can be decomposed and leveled, automatic synchronization between data stores and entities in Entity Relation data models, decomposition diagrams, context diagrams, state transition diagrams, and structure charts.



# 2

---

## *Working with Rational System Architect*

### **Introduction**

This chapter introduces you to working with Rational System Architect – how to select the methodology, diagram types, and properties that you will model with, how to create a diagram, use the Explorer and Toolbars.

<b>Topics in this chapter</b>	<b>Page</b>
<b>Getting Started with Rational System Architect</b>	<b>2-2</b>
<b>Selecting Available Diagrams and Properties</b>	<b>2-8</b>
<b>Creating a Diagram</b>	<b>2-19</b>
<b>Creating or Modifying a Definition</b>	<b>2-21</b>
<b>Using the Toolbars</b>	<b>2-25</b>
<b>Exiting from Rational System Architect</b>	<b>2-29</b>

---

## Getting Started with Rational System Architect

### The Audit ID

When you open Rational System Architect for the first time, you are presented with a dialog prompting you for an Audit ID, which is a user identification to distinguish users modeling with Rational System Architect.

Your Audit ID can consist of any combination of alphanumeric characters, up to a maximum of 7 characters. The first character cannot be blank. Your System Administrator may have established guidelines for establishing Audit IDs. Generally, an abbreviation of a person's name is used; for example, Bill Smith might use Audit ID BSmith, or Bills.

The Audit ID stamps all your work in Rational System Architect session. If you move symbols around on a diagram, for example, the diagram record in the encyclopedia is stamped with your Audit ID, the time and the date. This information can be printed out on a report, indicating what work Bills did on May 12, for example, or what definitions were changed by JohnC since January 1.

The Audit ID you enter is written to the sa2001.ini file. The next time you run Rational System Architect, that saved **Audit ID** is displayed as the default value, on the lower right corner of the application window. To overwrite the default, click the **File** menu and select **Audit ID**. You can change the Audit ID in the middle of a Rational System Architect session.

If you're working on an Enterprise encyclopedia, implemented via SA Catalog Manager utility, your network ID is automatically used as the **Audit ID**.

---

## Using 'Example' Project Encyclopedias

In Rational System Architect, all project work is stored in a project Encyclopedia. The encyclopedia is a database in SQL Server 2000, SQL Server 2005, SQL Express (replaces MSDE), and Oracle9i or Oracle 10g. In SQL Server and SQL Express, one encyclopedia corresponds to one database; in Oracle 9i and Oracle 10g an encyclopedia corresponds to a schema object. Rational System Architect provides several 'example' project encyclopedias, located in the subfolder <C>:\Program Files\IBM\Rational\System Architect Suite\11.3\System Architect\Encyclopedias.

### Attaching 'Example' Project Encyclopedias to Your Server

To open one of these 'example' encyclopedias, you must first 'attach' it to your server. The methods you can use to attach the encyclopedias are as follows:

- For an **SQL Server 2000 and SQL Server 2005** server, a Database Administrator can attach the encyclopedia with Microsoft's Enterprise Manager.
- For an **SQL Express** (replaces MSDE) server, you can attach the encyclopedia with IBM Rational's SAEM (SQL Server) utility.
- Rational System Architect does not provide 'example' Oracle9i or Oracle 10g encyclopedias. However, users in Oracle9i and Oracle 10g environments can also use an SQL Express server, since it can be installed locally on any computer. Another alternative is to create a new encyclopedia in Oracle9i or Oracle 10g and merge any of the 'example' encyclopedias into that Oracle9i or Oracle 10g encyclopedia.

Instructions for attaching a prebuilt encyclopedia to your server are provided in the on-line tutorials (either the Quick Start tutorial, or

the method tutorials). You can also find information in the online help for SAEM (SQL Server) and SAEM (Oracle).

To begin work on a new project of your own, you may create a new encyclopedia.

---

## Creating/Opening a Project Encyclopedia

### Creating a Connection

The first step in creating any encyclopedia is to establish a connection. A Connection serves as a pointer between your server and the encyclopedia that you are creating. First time users will not have any available connections. You'll need to create a new connection and specify the Server Name, Server Type, and Security. When creating a connection use the preferred name of the server connection. To create a Connection, perform the following steps:

1. Start **Rational System Architect**.
2. Select **File, Open Encyclopedia** (or click on the **Open Encyclopedia** icon on the toolbar
3. Click on the **New** icon.
4. Select the '...' (ellipsis) button, next to the **Connection** field. The **Connection Manager** dialog appears.
5. Select the **New** icon in the **Connection Manager** dialog.
6. Specify the name for the connection in the **Connection Name** field, preferably the same name as the server name. If your not sure of the server name you may type in the name of your computer.
7. You can find the name of your computer by going to your desktop and right clicking on the "**My Computer**" icon and choosing "**Properties**".
8. Depending on your version of Windows you are running, click on the "**Network Identification**" or "**Computer Name**" tab. The "**Full Computer Name**" will be displayed.
9. Type in the name of your computer +\TLOGICSA, i.e. **USNYC-SUSWTLOGICSA**.

10. Place your cursor inside of the **Server Name** field and specify the name of your server from the drop-down list.
11. Specify the type of server in the **Server Type** drop-down list, **SQL** or **Oracle**.
12. Select the type of Security you are using – either **Windows**, **SQL Server**, or **SQL (Persist Password)**. Your administrator should be able to advise which method you should use. If you are running SQL Express off of your local machine, then you can use Windows Security for Windows 2000, Windows 2003, and Windows XP machines.
13. Select **OK** to create the available connection.

### Creating a New Encyclopedia

To create a new encyclopedia, perform the following steps:

1. Select **Open Encyclopedia** from the **File** menu (or click on the **Open Encyclopedia** icon on the toolbar).
2. Click the **New** icon in the **Open Encyclopedia** dialog

Optionally, you can check the **Enterprise encyclopedia** checkbox to subject the encyclopedia to access control as implemented via SA Catalog Manager utility. Checking on the **Allow others to access this enterprise encyclopedia** checkbox makes it accessible to other users.

3. Specify the **Connection**. The **Connection** field serves as a pointer between the server and the encyclopedia (database) that you are creating. To select a connection, you can:

Click the drop-down arrow and select your server from the list

Or

Click the ‘...’ (ellipsis) button, next to the drop-down field. This method displays the **Connection Manager** dialog, where you can view existing connections or create new connections.

4. Select an available Connection and click **OK**.
5. In the **New encyclopedia name** field, type in the name of the encyclopedia that you want to create. If one already exists



with that name, you will be prompted to open that encyclopedia.

6. If not already selected, toggle on **Open this encyclopedia at startup**, located at the bottom left of the **Open Encyclopedia** dialog. This will make this encyclopedia the default encyclopedia that will automatically open each time you start SA.
7. Click **OK** to create the encyclopedia
8. The System Architect **Property Configuration** dialog will appear. Make choices on what diagram and property types you want to have available upon initial creation of the encyclopedia. You may change these choices at any time during a project, later on. The **Property Configuration** dialog is explained in more detail in the next section.
9. Click **OK** to the **Property Configuration** dialog to create the encyclopedia.

### **Opening an Existing Encyclopedia**

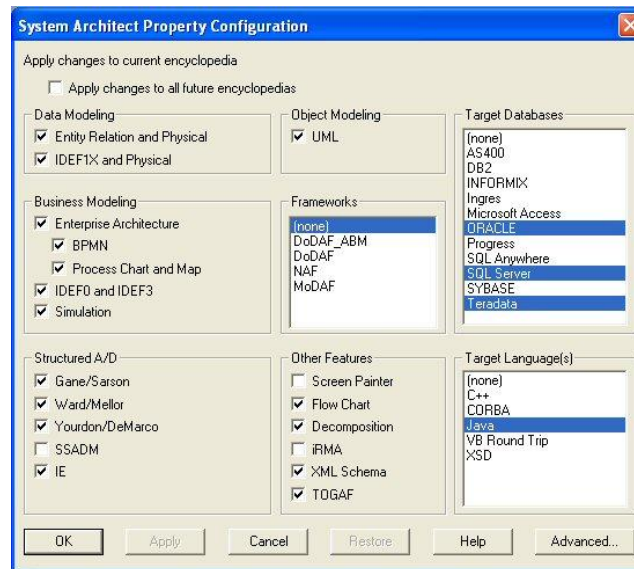
1. Two tabs are available to open an existing encyclopedia in the **Open Encyclopedia** dialog: the **Existing** tab and the **Recent** tab. You may use either tab to open existing encyclopedias encyclopedia, which exists as a database on either a SQL Server 2000, SQL Server 2005, SQL Express, and Oracle 9i or 10g servers.

## Selecting Available Diagrams and Target Languages

The **System Architect** Property Configuration dialog enables you to select the diagrams and definition properties that are available at any time during a session.

1. From the **Tools** menu, select **Customize Method Support, Encyclopedia Configuration**. The **System Architect Property Configuration** dialog appears.

Figure 2-1. The **System Architect Property Configuration** dialog



2. Toggle on the diagram types and methodology types that you want to work with.
3. You may select one or more target languages for code generation, selecting from C++, Java, Visual Basic, and CORBA IDL. You may also select None.

4. You may select one or more target database property set(s). You may also select None. A database property set is not required for logical data modeling with Entity Relation models. It is required for modeling with a physical data model; however, the choice of database can be selected when creating the physical model. (see also, setting Default DBMS type.)
5. Click on **OK** to close the dialog.
6. Select **File, Open Encyclopedia** (or click on the **Open Encyclopedia** icon on the toolbar) to reopen the encyclopedia for the changes to take effect.

## Configuration Dialog Options

The contents of the **System Architect Property Configuration** dialog are described in the table below:

Option/Button Name	Description
<b>Apply the changes to all future encyclopedias</b>	Toggle on <b>Apply changes to all future encyclopedias</b> if you want every new encyclopedia to start with the same choices. SADECLAR.CFG in the executable subdirectory is modified.
<b>Data Modeling</b>  <b>Entity Relation</b>          <b>IDEF1X</b>	Toggling on <b>Entity Relation</b> in the <b>Data Modeling</b> section provides the following diagrams: <b>Entity Relation Model diagram</b> <b>Entity Relation Subject Area diagram</b> <b>Physical diagram</b>  Toggling on <b>IDEF1X</b> in the <b>Data Modeling</b> section provides the following diagrams: <b>IDEF1X Model diagram</b> <b>IDEF1X Subject Area diagram</b>

Option/Button Name	Description
<p><b>Business Modeling</b></p> <p><b>Enterprise</b></p> <p><b>BPMN</b></p> <p><b>Process Chart and Map</b></p> <p><b>IDEF0</b></p> <p><b>IDEF3</b></p>	<p>Business Enterprise modeling provides the following diagrams:</p> <p><b>Business Concept</b>  <b>Decision Chart</b>  <b>Flow</b>  <b>Functional Hierarchy</b>  <b>Network Concept</b>  <b>Organizational Chart</b>  <b>Process Decomposition</b>  <b>Process Hierarchy</b>  <b>Relationship Map</b>  <b>System Architecture</b>  <b>System Area Map</b>  <b>System Context</b>  <b>System/Subsystem Structure</b></p> <p>Toggling on <b>BPMN</b> provides the following diagram:  <b>Business Process Diagram</b></p> <p>Process Chart and Map provides the following diagrams:  <b>Process Chart</b>  <b>Process Map</b></p> <p>Toggling on <b>IDEF0</b> enables the following diagrams:  <b>IDEF0</b>  <b>Node Tree</b></p> <p>Toggling on <b>IDEF3</b> enables the following diagrams:  <b>IDEF3 Process Flow</b>  <b>IDEF3 Object State Transition</b></p> <p style="text-align: right;">2-11</p>

Option/Button Name	Description
<b>Business Enterprise</b>  <b>Simulation</b>	Toggling on <b>Simulation</b> enables simulation property sets for the following diagrams:  <b>BPMN</b> <b>Process Charts</b> <b>IDEF3 Process Flow</b>
<b>Object Modeling</b>  <b>UML</b>	Selecting UML provides the following diagrams: <b>Activity</b> <b>Use Case</b> <b>Sequence</b> <b>Collaboration</b> <b>Class</b> <b>State</b> <b>Component</b> <b>Deployment</b>
<b>Structured A/D</b>  <b>Gane/Sarson</b>        <b>Ward/Mellor</b>         <b>Yourdon/DeMarco</b>	Toggling on <b>Gane/Sarson</b> enables the following diagrams: <b>Data Flow Gane &amp; Sarson</b> <b>State Transition</b> <b>Structure Chart</b> In addition, <b>Flow Chart</b> and <b>Decomposition</b> diagrams are enabled.  Toggling on <b>Ward/Mellor</b> enables the following diagrams: <b>Data Flow Ward &amp; Mellor</b> <b>State Transition Ward &amp; Mellor</b> <b>Structure Chart</b> In addition, <b>Flow Chart</b> and <b>Decomposition</b> diagrams are enabled  Toggling on <b>Yourdon/DeMarco</b> enables the following diagrams:

Option/Button Name	Description
<p><b>SSADM</b></p>	<p><b>Data Flow Yourdon/DeMarco</b>  <b>State Transition</b>  <b>Structure Chart</b>                      In addition, <b>Flow Chart</b> and <b>Decomposition</b> diagrams are enabled</p> <p>Toggling on SSADM enables the following diagrams</p> <p><b>SSADM Context</b>  <b>SSADM Data Flow</b>  <b>SSADM Data Structure</b>  <b>SSADM Dialogue Structure</b>  <b>SSADM Document Flow</b>  <b>SSADM Effect Corr</b>  <b>SSADM Enquiry Process</b>  <b>SSADM Entity Life History</b>  <b>SSADM I/O Structure</b>  <b>SSADM Logical Database Process</b>  <b>SSADM Menu Structure</b>  <b>SSADM Physical Database Process</b>  <b>SSADM Resource Flow</b>  <b>SSADM Update Process</b></p>
<p><b>IE</b></p>	<p><b>Toggling on IE (Information Engineering)</b> enables the following diagrams:  <b>Data Flow Gane &amp; Sarson</b>  <b>State Transition Ward &amp; Mellor</b>  <b>Structure Chart</b>  <b>Entity Relation Diagrams</b>                      In addition, <b>Flow Chart</b> and <b>Decomposition</b> diagrams are enabled.</p>

Option/Button Name	Description
<b>Other Useful Diagrams</b>	
<b>Screen Painter</b>	Use the <b>Screen Painter</b> option to create the required dialogs if you are designing a GUI interface or character screens for your project. The following diagrams become available: <b>Graphic Screen, Character Screen, Menu.</b>
<b>Flow Chart</b>	Toggle the <b>Flow Chart</b> option on to make the <b>Flow Chart</b> diagram available.
<b>Decomposition</b>	<b>Decomposition</b> diagrams are used to diagrammatically represent an organization or a system in a tree hierarchy. Toggling this option provides the <b>Decomposition</b> diagram and the <b>Auto-Decomposition</b> diagram (creates hierarchies automatically based on provided information).
<b>iRMA</b>	<b>iRMA - integrated Reference Model Architect</b> is used to assist Departments and Agencies in incorporating and using the Office of Management and Budget (OMB) Reference Models.
<b>XML Schema</b>	Toggling on <b>XML Schema</b> option enables you to model XML designs in an XML hierarchy diagram.
<b>TOGAF</b>	Selecting <b>TOGAF</b> (The Open Group Architectural Framework) enables the following diagrams: <ul style="list-style-type: none"> <li>Business Architecture</li> <li>Technical Architecture</li> <li>Help</li> </ul>



Option/Button Name	Description
<b>Advanced</b> button	Click the <b>Advanced</b> button to select more specific diagram types and property sets for your project. See the next section in this chapter.
<b>Restore</b> button	Click the <b>Restore</b> button to return the configuration options to their default settings.
<b>Frameworks</b> <b>DoDAF</b> <b>DoDAF ABM</b> <b>NAF</b> <b>MoDAF</b>	Toggling on this option enables you to model in the US Department of Defense's Architecture Framework, The NATO Architecture Framework (NAF) is based on the U.S. DoD Architecture Framework (DoDAF), and The MOD Architectural Framework (MoDAF) is the UK Ministry of Defence Architecture Framework.  Available Frameworks are:  <b>DoDAF, DoDAF ABM, NAF, and MoDAF</b>

---

## Using the Advanced Configuration Dialog

Once you've chosen your methodologies, target databases, and other options on the **System Architect Property Configuration** dialog, you can refine those choices further by clicking on the **Advanced** button. This brings you to the **Configure Property Set** dialog, where you may add or subtract diagrams from the sets selected by methodology, or add/subtract property sets.

There are special properties that can only be added or removed from the **Configure Property Set** dialog. For example, Map AS/400 data types, Map dBASE data types, Map Paradox data types, and Map Progress data types.

To access the **Configure Property Set** dialog, perform the following steps:

1. Click the **Tools** menu and select **Customize Method Support, Encyclopedia Configuration**.
2. Click the **Advanced** button.

**To Change the  
Diagrams  
Selected List**

You may refine the selections you have made for diagrams by performing the following steps:

1. In the **Available** Diagrams list on the left, highlight the diagram(s) you want to use.

**Note:** To highlight consecutive items in a list, hold the **Shift** key while clicking the first and last item the list. To highlight non-consecutive items in a list, hold the **Ctrl** key while clicking each item in the list.

2. Click on **Add** to move them to the **Selected** Diagrams list.

or

In the **Selected** Diagrams list on the right, highlight the diagram items you do not want to use.

Click on **Remove** to move them back to the **Available** Diagrams list.

### To Change the Property Sets Selected List

You may refine the selections you have made for properties by performing the following steps:

1. In the **Available Property Sets** list on the left, highlight the property set(s) you want to use and click **Add** to move them to the **Selected** Property Sets list.

or

From the **Selected** Property Sets list on the right, highlight the property set(s) you do not want to use.

Click on **Remove** to move them back to the **Available** Property Sets list.

To make changes to the Advanced Configuration dialog take effect:

2. Click on **OK** to save the selected diagram and property sets and close the **Configuration Property Set** dialog.
3. Click on **OK** to close the **Rational System Architect Configuration** dialog.
4. Reopen the encyclopedia for the changes to take effect (click the **Open Encyclopedia** icon in the Toolbar or select the **Open Encyclopedia** command from the **File** menu).

---

## Creating a Diagram

You may create a new diagram from the Main menu or the Explorer. In the Explorer, if the **All Methods** tab is in view, a list is displayed of all available diagrams or definitions (depending on the methodology and property sets you have chosen for the project). If you are in a specific methodology tab (ie, UML), the list of diagrams or definitions is limited to those available within the chosen methodology.

### Using the File Menu to Create a New Diagram

To create a new diagram from the Main menu, perform the following steps:

1. Click **File**, select **New Diagram**, or click the **New Diagram** icon.
2. Click on the desired diagram type.
3. Type the name of the diagram, up to 80 characters, in the **Name** text box.
4. Click on **OK**, or press **ENTER**.

### Using the Explorer to Create a New Diagram

To create a new diagram from the Explorer, perform the following steps:

1. In the Explorer, select the tab that contains the diagram type you want to add (ie, **UML**), or select the **All Methods** tab.
2. Right-mouse-click on the **Diagrams** selection and select **New** from the floating menu. Select a diagram type from the drop-down list and type in a name for the new diagram. Press **OK**.

OR

Expand the **Diagrams** group (double click on **Diagrams** or click on its expand indicator) and right-mouse-click on a particular diagram type (ie,

**UML Class**). Select **New** from the floating menu. Type in a name for the new diagram and press **OK**. The new diagram is created.

---

## Creating or Modifying a Definition

Rational System Architect diagram symbols contain definitions. Definitions also exist within an encyclopedia that are not represented by symbols (requirements, for example, or attributes, or methods).

Rational System Architect offers you a number of ways to add a definition to an encyclopedia. You may add one definition at a time or add multiple definitions of a particular type in one function. Definitions may be added through the following ways:

- Add a definition to a symbol drawn on a diagram. To open the symbol's definition dialog, either double-click on the symbol, right-mouse-click on the symbol and choose **Edit** from the drop-down list, or select the symbol and choose the symbol's name from the **Edit** menu. (See the next section of this chapter.)
- Add a definition through the Explorer. (See the next section of this chapter.)
- Through the **Dictionary** menu, **New Definition** command. This command will open an **Add Definition** dialog, and also open the Explorer if it is closed. (See the next section of this chapter.)
- The **Dictionary** menu, **Import Definitions** option, which enables you to add multiple definitions of the same type by importing them into the project encyclopedia from various sources (ie, Word documents, Excel projects, etc).

- The **Matrix Editor** may be used to enter definitions into the encyclopedia. Matrix editors are provided for certain definition types in Rational System Architect, especially business modeling definitions. New matrices may be added by the user.
- The **Tools** menu, **Merge** option which adds definitions in from another encyclopedia.
- Import C++ or Java code into a class diagram (UML Class diagram, Object model). This import mechanism is accessed by selecting **Dictionary, Forward and Reverse Code Engineer** with one of the class diagram types listed above, open, and in focus. (See the on-line help, UML Modeling folder.)
- Import DDL code from supported RDBMS's via the **Reverse Data Engineer** (accessed via the **Dictionary** menu). (See the on-line help, Data Modeling folder.)

### Preferences on Adding Definitions

If you prefer to define while you're drawing, make sure *[Auto] Define* is toggled on in the Tools menu, Preferences dialog. The Dictionary Object [type] dialog is automatically presented every time a new definition is added to the encyclopedia.

### Creating a New Definition

There are a number of ways to create a new definition. You may create one from the menu or from the explorer, or draw a symbol on a diagram and define it.

### To create a new definition from the menu:

1. Select **Dictionary, New Definition**.
2. Double-click on a definition type from the dialog that appears.



3. Type in the name of the new definition (up to 80 characters) and click **OK**.

**Adding and Saving Definitions to the Repository**

Changes to definitions in the Dictionary are written to the repository just as soon as you click the **OK** button on any **Dictionary Object <Type > <Name>** dialog.

Contrast this procedure to the procedure of modifying and saving diagrams. With a diagram, you are allowed to make change upon change, but to finally decide in the end to discard all changes without writing them to the database. Definition changes do not give you this option. When you close a **Dictionary Object <Type > <Name>** dialog, you must either click on **OK** or **Cancel**; the former causes an immediate database update.

**Creating/Modifying a Definition**

With a diagram open, you may create a new definition by drawing a symbol of a particular type on the diagram, and adding a definition to it, or modify an existing definition for a symbol already drawn (a table on a physical diagram, or a class on a class diagram, for example).

**To create/modify  
a new definition  
for a symbol on a  
diagram:**

1. Select the symbol.
2. From the **Main Menu**, click **Edit**, and select **Edit “Symbol Name”**,

OR

Right-click on a symbol and select **Edit** from the drop-down list,

OR double-click on the symbol.

**To create/modify  
a definition  
through the  
Explorer:**

1. Use the Explorer to find the definition.
2. Open the definition by simply double-clicking on it or right-mouse clicking on it and selecting **Open** from the drop-down list.

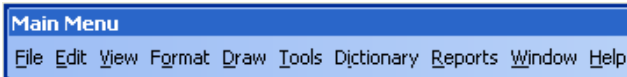
## Using the Toolbars

Rational System Architect provides a main menu and a number of toolbars that enable you to work the tool. All menus and toolbars are able to be undocked from their default position, and 'floated' on your desktop, or docked to other areas of the Rational System Architect product frame. All toolbars can be edited, to add or remove tools from them. You can also create your own, custom toolbars.

In addition, each specific diagram type in Rational System Architect provides its own drawing toolbar. This toolbar can also be undocked, floated, docked elsewhere.

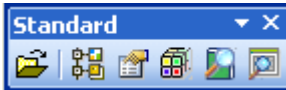
### Main Menu

Use the main menu to initiate most of the functions contained in Rational System Architect. The default main menu is pictured below.



### Main Toolbar

Use the main toolbar to initiate general functions in Rational System Architect, such as opening existing diagrams, creating new diagrams, saving diagrams, opening the explorer, or launching Visual Basic for Applications.



### Edit Toolbar

The Edit Toolbar enables you to cut and paste diagram elements.



**Diagram Toolbar**

The diagram toolbar enables you to perform generic diagram functions, such as opening a diagram, saving a diagram, printing a diagram, zooming in or out on a diagram view, or finding something on the diagram.



**Moving a Toolbar**

To move a toolbar, click the move handle on a docked toolbar, or click the title bar on a floating toolbar, and drag the toolbar to a new location. If you drag the toolbar to the edge of the program window, it becomes a docked toolbar.

**Hiding or Showing a Toolbar**

To hide a toolbar, right-click on the toolbar (on any portion of it except the title bar), and then click the toolbar you want to hide on the shortcut menu.

If you hide all the toolbars and want to see one again, you can right-click the menu bar, and then click the toolbar you want on the shortcut menu.

To quickly hide a floating toolbar, click the **Close** button on the toolbar.

---

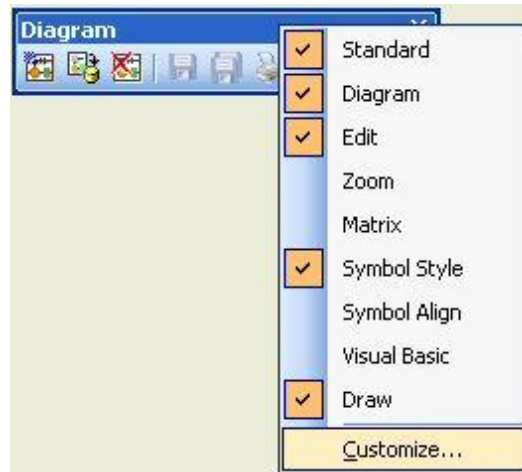
## Customizing the Toolbars

### Creating a Custom Toolbar

You can create one or more custom toolbars for your own use.

1. Right-click any toolbar (on any portion of it except the title bar), and then click **Customize** on the shortcut menu.

Figure 2-2. Customizing a toolbar.



2. Select the **New** button on the **Toolbars** tab of the **Customize** dialog.
3. Name the toolbar and click **OK** to add it to the toolbar list.
4. Select the toolbar from the toolbar list and close the **Customize** dialog. The new toolbar will be added to the Rational System Architect workspace. Since there are no tools in it as a default, it will show up as a small box.

5. Right-click the new toolbar (on any portion of it except the title bar), and click **Customize** on the shortcut menu.
6. Select the **Commands** tab in the **Customize** dialog; select a tool from the **Commands** list, and drag it onto the new toolbar.

### **Customizing an Existing Toolbar**

You can customize any of the toolbars, placing command buttons on it that you feel are most convenient for your use, or removing command buttons from it.

1. Right-click any toolbar (on any portion of it except the title bar), and then click **Customize** on the shortcut menu.
2. Select the **Commands** tab of the **Customize** dialog.
3. Select the category which contains the command you want to add. You can browse commands for each category in the **Commands** list box on the right of the **Customize** dialog, **Commands** tab.
4. Select the command button and drag it onto the toolbar that you wish to add it to.

### **Resetting a Toolbar to It's Default Settings**

To reset a toolbar to it's default settings:

1. Right-click any toolbar (on any portion of it except the title bar), and then click **Customize** on the shortcut menu.
2. Select the toolbar in the Toolbars list box, located on the **Toolbars** tab of the **Customize** dialog.
3. Click on the **Reset** button.

---

## Exiting from Rational System Architect

To exit from Rational System Architect, do one of the following:

- Pull down the **File** menu, and click on **Exit**.
- Click the **X** in the upper right-hand corner of the application window.
- Press **Alt-F4**.

If you have one or more diagrams open that have been altered, you are asked, one at a time, if you want to save them.

**Saving the Format File:** If you made any style changes, you are asked if you want to save them in a style sheet when exiting Rational System Architect. The name of the format file is **AUTOEXEC.STY**.

Note that you are never prompted for saving data dictionary entries, so long as you close the **Definition** dialog before exiting Rational System Architect. Changes to the dictionary are written immediately to the database as soon as you click on **OK**. Unlike diagrams, dictionary entries can never be in a state of "altered, but not yet saved." If you close Rational System Architect before closing the **Definition** dialog, you are prompted to save the open definition. See information on **Exit** command, **File** menu, in the on-line help for more details.





# 3

---

## *Working with the Explorer*

### **Introduction**

The Rational System Architect Explorer is the primary tool for viewing information in the project encyclopedia. Through the Explorer, you may also create, edit, and delete diagrams and definitions, and perform a variety of other functions. This section introduces you to the Explorer's many features.

<b>Topics in this chapter</b>	<b>Page</b>
The Explorer's Functions	3-2
Viewing Items in the Explorer	3-3
The Properties Option	3-4
Docking/Undocking the Explorer	3-5
Diagrams – Creating, Editing, Deleting through the Explorer	3-6
Definitions – Creating, Editing, Deleting through the Explorer	3-8
Permanently Deleting Items from the Encyclopedia	3-10

---

## The Explorer's Functions

Rational System Architect diagrams and definitions are listed within the methodology tab to which they belong, and within the **All Methods** tab. The Rational System Architect Explorer provides basic navigation through an encyclopedia, allowing you to:

- List encyclopedia diagrams, based on the selected tab and the diagrams types selected in the **System Architect Configuration Property** dialog.
- View diagram thumbnails or definition properties in the **Properties** window.
- Create, open, or delete a diagram.
- Create, modify, copy, or delete a definition.
- Choose to make a definition or diagram **read-only** to prohibit someone from modifying it.
- Set filter options to determine which diagrams or definitions, if any, you want to hide from the Explorer.
- Turn the view properties options on or off.

---

## Viewing Items in the Explorer

Models, diagrams, and definitions can be viewed in the Explorer. The Explorer filters the information shown through a number of tabs. The **All Methods** tab shows all models, diagrams, and definitions in the project encyclopedia. The **All Methods** tab is always available. The other tabs sort diagrams and definitions by methodologies – for example, Business Modeling, Data Modeling, and Object Modeling, etc.

Which tabs are displayed in the Explorer depends on the methodologies you select in Encyclopedia Configuration settings for the encyclopedia – for example, if UML is selected, the **UML** tab will appear on the Explorer.

### Configuration Settings

The Explorer only shows, and makes available to add, diagram and definition types that are currently selected for the project encyclopedia. Diagram and definition types depend on the methodology and property sets you have chosen for the project (**Tools, Customize Method Support, Encyclopedia Configuration**). See Chapter 2 for more information on setting encyclopedia configuration.

---

## The Properties Option

The **Properties** area in the lower half of the Explorer enables you to quickly browse the details of a diagram or a definition without (or before) opening it. You may display diagram pictures or properties, and definition details in the **Properties** area.

When a *diagram* is selected in the Explorer, the **Properties** area in the lower half of the Explorer displays:

- A thumbnail image of a selected diagram, or
- All the values of properties of a selected diagram

When a *definition* is selected in the Explorer, the **Properties** area displays:

- All the values of properties for a selected definition.

### Selecting Properties

To select **Properties** (if it is not showing), right-mouse-click on the Explorer whitespace, and select **Properties**. You may choose to close the Properties area if you wish to lengthen the list area.

### Diagram Properties Options

For diagrams, if the **Pictures** choice from the drop-down menu is toggled on, you will view a thumbnail image of the diagram, if it is toggled off, you will see values of properties for a selected diagram. Some diagrams do not have properties associated with them, so the Details will be empty.

**Note:** When the **Properties** command is toggled off (or not selected), SA does not show a diagram picture either even when the **Show Picture** command is on.

---

## Docking/Undocking the Explorer

The Rational System Architect Explorer can be docked at the left, right, top, or bottom of the SA workspace. Left is the default dock position. The Explorer may also be undocked and, in that state, may be closed. Once closed, there are several ways to open the Explorer.

### Undocking the Explorer

To undock the Explorer:

- Click on the Explorer title bar and drag toward the left, right, top or bottom of the workspace until the outline conforms to the workspace.


### Docking the Explorer

There are a number of ways to dock the Explorer. From an open and undocked (floating) Explorer:

1. Point your cursor at the floating Explorer's Title Bar.
2. Click and drag the Explorer toward the left, right, top or bottom of the workspace until the outline conforms to the workspace.
1. Release the mouse button to dock it.

### Opening a Closed Explorer

To open the Explorer:

- Select **Explorer** from the **View** menu (or select the Explorer toolbar button )

OR

- Select **File, New Diagram** or **Dictionary, New Definition** to open the Explorer, along with the specified dialog (**New Diagram** or **New Definition**).

---

## Diagrams – Creating, Editing, Deleting through the Explorer

### Creating a New Diagram

New diagrams are added to a project through the Explorer by right-mouse clicking on an appropriate area of the Explorer, described below. A list of items comes up to select from – the list is dependant on the configuration settings for the encyclopedia, what methodology tab you are pointing to, and where on the Explorer your cursor is pointing when you right-mouse click.

To create a new diagram from the Explorer, perform the following steps:

1. In the Explorer, select the tab that contains the diagram type you want to add (i.e., **UML**), or select the **All Methods** tab.

If you are in the **All Methods** tab, you get a list of all available diagrams or definitions, dependant on the methodology and property sets you have chosen for the project. If you are in a specific methodology tab (i.e., **UML**), the list of diagrams or definitions is limited to those available within the chosen methodology.

2. Right-click on the **Diagrams** icon, select a diagram type from the drop-down list, type the diagram name and press **OK**. The new diagram is created.

OR

Expand the **Diagrams** group (double click on **Diagrams** or click on it's expand indicator) and right-click on a particular diagram type (i.e., **Class**). Type the diagram name and press **OK**. The new diagram is created.

### Editing a Diagram

Editing a diagram consists of opening it and graphically editing it. The same rules described above for adding a diagram, pertain to editing a diagram – that is, the configuration settings must have

been set to include the methodology of the diagram type you wish to open/edit, for it to appear in the Explorer.

To open and edit a diagram:

1. Find the diagram type using the **All Methods** tab or the tab of the particular methodology you are using (e.g., **UML**).
2. Find the particular diagram name. Use the **Properties** section at the bottom of the Explorer to help in locating the diagram you are after.
3. Once the diagram is selected in the Explorer, double-click on it or right-mouse-click on it and choose **Edit** to open it.

### **Editing Diagram Properties**

Some diagrams in Rational System Architect contain definition properties for the diagram itself (for example, a Sequence diagram contains a diagram property for showing or not showing *Focus of Control* lines on the diagram). These diagram properties cannot be directly edited through the Explorer – you must open the diagram, and select **Diagram Properties** (right-mouse click on the diagram workspace and select **Diagram Properties**, or select **Edit, Diagram Properties** from the main menu).

### **Deleting a Diagram**

To delete a diagram through the Explorer, first find and select it using the techniques described above for editing or adding a diagram. Then right-mouse-click on the diagram and choose **Delete** from the drop-down menu choice.

You will be presented with a dialog to confirm you want to delete the diagram and all of its dependencies. You are also given the option to preview the dependencies affected by your proposed deletion. You can still cancel the deletion at this point.

---

## Definitions – Creating, Editing, Deleting through the Explorer

### Creating a New Definition

New definitions are added to a project through the Explorer by right-mouse clicking on an appropriate area of the Explorer, described below. A list of items comes up to select from – the list is dependant on the configuration settings for the encyclopedia, what methodology tab you are pointing to, and where on the Explorer your cursor is pointing when you right-mouse click.

To create a new definition from the Explorer, perform the following steps:

1. In the Explorer, select the tab that contains the definition type you want to add (e.g., for a class method, select **UML** tab), or select the **All Methods** tab.

If you are in the **All Methods** tab, you get a list of all available diagrams or definitions, dependant on the methodology and property sets you have chosen for the project. If you are in a specific methodology tab (i.e., Object), the list of diagrams or definitions is limited to those available within the chosen methodology.

2. Right-click on the **Definitions** icon, select a definition type from the drop-down list, type in a name for the new definition and press **OK**. The new definition is created.

OR

Expand the **Definitions** group (double click on **Definitions** or click on it's expand indicator) and right-mouse-click on a particular definition type (i.e., **Component**). Type the name for the new definition and press **OK**. The new definition is created.



**Editing a Definition**

To edit a definition through the Explorer, first find and select it using the techniques described above for adding a definition – for example, use the **All Methods** tab or the tab of the particular paradigm you are using (e.g., **UML**), find the definition type, and then the particular definition.

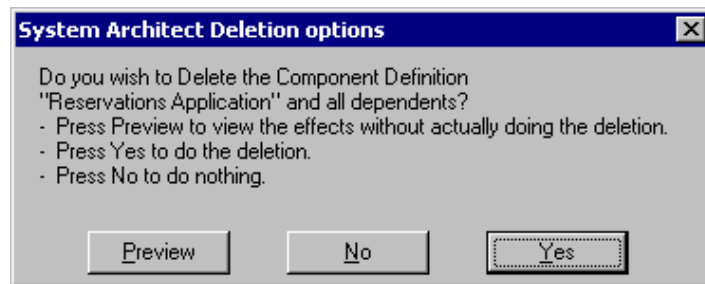
Select the definition in the Explorer window. The **Properties** window displays the selected definitions properties. You can edit directly on the properties window, or click the **Edit Object** link to display the definition dialog.

**Deleting a Definition**

To delete a definition through the Explorer, first find and select it using the techniques described above for editing or adding a definition. Then right-mouse-click on the definition and choose **Delete** from the drop-down menu choice.

You will be presented with a dialog to confirm you want to delete the definition, and all of its dependencies. You are also given the option to preview the dependencies affected by your proposed deletion. You can still cancel the deletion at this point.

Figure 3-1. Deletion options message box.



---

## Permanently Deleting a Definition from the Encyclopedia

### 'Shrinking' the Database

When you use Rational System Architect commands to delete a diagram, symbol, or definition from a project, the information will still exist in the database, but be unused.

To permanently remove the items, you must *shrink* the database – for encyclopedias on MSDE, this function is performed using IBM Rational's SAEM (Encyclopedia Manager) tool; for SQL Server 2000, this function is performed using the Enterprise Manager tool provided with SQL Server 2000. For network encyclopedias accessed by multiple users, this function is normally performed by the systems administrator at various stages of the project, coinciding with versioning of the project.

Instructions for shrinking a database using SAEM are provided in the SAEM help file; likewise for SQL Server's Enterprise Manager.



# 4

---

## ***General Drawing Techniques***

### **Introduction**

This section describes how to draw diagrams using Rational System Architect, and covers many of the drawing and viewing options available in the product.

<b>Topics in this chapter</b>	<b>Page</b>
Drawing in Rational System Architect	4-2
Drawing Symbols on a Diagram	4-3
Drawing Lines	4-6
Diagram Settings	4-15
Moving Symbols and Editing a Diagram	4-17
Fonts	4-21
Format Files	4-24
Copying Diagrams	4-32
Keyboard Accelerators	4-34

---

## Drawing in Rational System Architect

Rational System Architect is a visual modeling tool for the Enterprise. It provides a multitude of preferences for drawing and viewing the graphical diagrams that represent an organization's analysis and design models.

### Drawing Symbols

There are many options for drawing symbols in Rational System Architect. You may use a diagram's toolbar to draw symbols, or drag existing definitions from the explorer onto the diagram workspace. You may choose to draw and define symbols as you go along, or choose to simply draw and create a list of undefined items, so that you can brainstorm and draw, and define everything later.

### Drawing Lines

As you draw lines to connect symbols, there are preferences for types of lines (always straight, always orthogonal, or elliptical), line drawing algorithms, grids that lines adhere to, etc. Most diagram types purposely do not force you to connect both ends of a line to a symbol, however, data modeling diagrams in Rational System Architect provide "on-the-fly" referential integrity, enforcing the rules of data modeling that prohibit you from leaving a line unconnected to a symbol on either end.

### Viewing Diagrams

There are a number of preferences to select on viewing the diagram. You may select to show page marks, grids, rulers, etc. You may print diagrams to various percentages, to paper of various sizes. You may print very large diagrams in whole to printer plotters.

---

## Drawing Symbols on a Diagram

### Drawing Versus Selecting

Drawing in Rational System Architect begins by selecting a symbol from a diagram's toolbar. Once selected, you can draw. The default setting for selecting versus drawing is *either/or*. That is, the cursor is set to draw a symbol or to select a symbol. If it is set to draw, the shape turns into a pen with a square next to it, or a pen with a line next to it, and so on, depending on the symbol type. In order to use the cursor to select, rather than draw, do one of the following:

- select the cursor in the upper left-hand corner of the Toolbox, or
- click on **Select Mode** in the **Draw** menu,
- or hit the **Escape** key.

### Simultaneous Select/Draw Mode

If you prefer to use the cursor for selecting and drawing, without going back and forth to **Select Mode**, toggle on **Simultaneous Select/Draw** in the **Preferences** dialog from the **Tools** menu. In **Simultaneous Select/Draw** mode, the user can select a symbol type from the toolbar, drop one down, then select any symbol already on the diagram, then perform an action on the selected symbol (either edit its definition or move it or resize it), then drop down another new symbol of the selected type, then select and edit other existing symbols, etc, without ever having to go to the toolbar and change select modes.

The **Simultaneous Select/Draw** mode may seem unintuitive at first to the new user, but once harnessed, can be a very efficient drawing mode. With **Simultaneous Select/Draw** mode turned on, Rational System Architect uses both mouse *clicks* and mouse *presses*, according to the following behavior:

- A *press* onto the diagram workspace drops a new symbol of the selected type down.
- A *click* on an existing symbol or line selects it for a further action. Once selected, you may perform an action on the symbol or line, such as editing its definition (right mouse click, Edit), resizing it (select one of its handlebars

and move it), or moving it on the diagram (press on the selected symbol and then move).

- Pressing on a selected symbol, as mentioned above, enables you to move it. But the symbol **must** be selected first.

### **Dragging Symbols from the Explorer**

Definitions may be dragged from the Explorer onto the diagram workspace to create the appropriate symbol representing the definition. The definitions that can be dragged are those that apply to the diagram in focus. For example, if a UML Class diagram is open, you may drag classes, objects, components, etc, onto it. You are restricted from dragging a Gane & Sarson Data Flow onto it, however.

For Entity Relation diagrams (both Models and Subject Areas), dragging an entity onto the diagram workspace will also cause appropriate relationships to be drawn on the diagram, from the entity being dropped to other entities appearing on that diagram that are related in the underlying project model.

---

## Naming and Defining Symbols

**On-Screen Editing**

Rational System Architect provides on-screen editing of symbols. On-screen editing is automatically enabled if you are viewing a diagram close enough so that the letters on symbols are legible. On-screen editing is automatically disabled if your view of a diagram is wide enough so that the letters on symbols become too small to read. With on-screen editing disabled, you will instead be presented with a **Name Symbol** dialog. The reasoning behind this behavior is that if the lettering is too small to read, on-screen editing would make it difficult for you to see what symbol name you are typing in.

**Preferences**

Select **Tools, Preferences** to set many drawing options in Rational System Architect. The **Preferences** dialog will open. If you want a symbol's definition dialog to pop up everytime you draw a symbol, toggle on the (Auto) **Definition** choice.

**Note:** Explanations for all of the selections in the **Preferences** dialog are contained in the on-line help.

**Format, Symbol Format**

The **Format, Symbol Format** menu choice unveils a number of commands to set the appearance of a selected symbol or line. To make a selection for appearance (line width, font, color, etc) stick for all future symbols or lines drawn, select **Format, Symbol Format, Symbol Style**.

**Note:** Explanations for all of the selections in the **Format, Symbol Format** menu are contained in the on-line help.



---

## Drawing Lines

In Rational System Architect, all drawing symbols are either "rectangular" symbols or "line" symbols.

All symbols other than lines are considered rectangular symbols. It may seem strange to consider circles, ellipses, and diamonds as "rectangular". However, if you select any of these symbols, you will see from its handles that the symbol is assigned a fully rectangular space on the screen. This is called its *bounding rectangle*.

### Drawing Lines

You draw a rectangular symbol by selecting the particular symbol from the toolbox or **Draw** menu and dropping it onto the diagram. Line symbols can be drawn between rectangular symbols (i.e., an entity relation line symbol is drawn between two entity symbols) or between white space on your diagram and a rectangular symbol (i.e., in data flow diagrams (DFDs)). Before drawing a line symbol, you may want to select the line style.

### Selecting Line Styles

There are three line styles you can choose from; the choice is made in the **Line Style** dialog box, accessed from the **Line** command under the **Format, Symbol Format** menu:



#### **Straight – orthogonal**

You may also select to have orthogonal lines with curved bends. You make this selection in the **Line Style** dialog box. You can also set the radius of curvature of the bends.



#### **Straight – any orientation**

lines are recommended for structure charts where lines need to be drawn at any angle.



#### **Elliptical arcs**

are available, but not used by any diagram type supported in Rational System Architect.

**Quitting Line Draw**

You can quit in the middle of drawing a line if you have not yet attached it to the target symbol. Press the **Esc** key. The line disappears and the mouse pointer changes to an arrowhead shape.

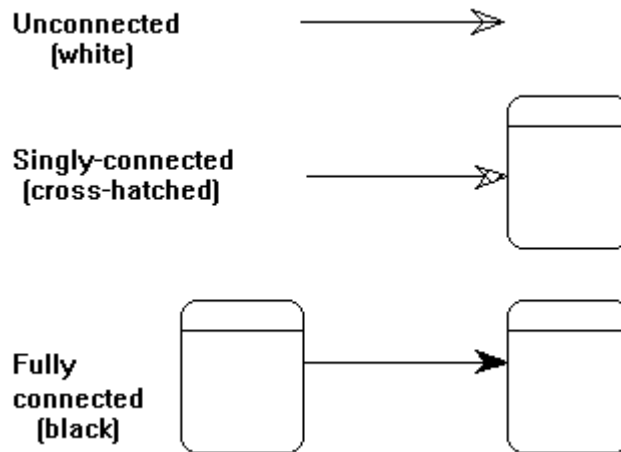
**Singly Connected Lines**

Lines aren't always attached at both ends. In a DFD, it is correct for a data flow line to start from the edge of the paper and feed data into a process symbol. This is a singly connected "interface" data flow. It is never correct for a data flow line to be unconnected at both ends.

Some types of line symbols in Rational System Architect have arrowheads on one or both ends (such as a data flow line); others do not contain arrowheads at all (such as connection lines on decomposition diagrams).

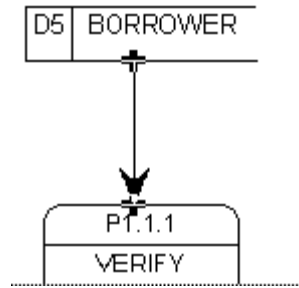
For lines with arrowheads, you can tell how well the line is connected by looking at the arrowhead itself. If it is solid, the line is fully connected to a rectangular symbol at both ends. If it is cross-hatched, at least one end is not connected to a rectangular symbol.

Figure 4-1. Line Symbol Arrowheads Tell How a Line Is Connected



You can also tell if the line symbol is connected by looking at the handlebar at each end of the line. If it is in the shape of a cross, it is attached to a rectangular symbol. If it is a simple, solid square handle bar, it is not attached.

Figure 4-2. Cross-shaped handles at line attached points



To draw a singly connected line when you are using the orthogonal line style, simply click the left mouse button once to start the line, and once again at the end point. Rational System Architect determines the best placement for the bend points.

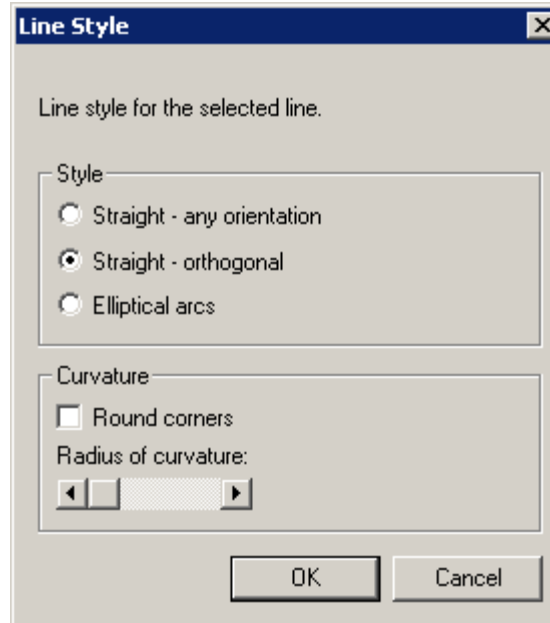
### From- and To-Symbols

If you are connecting two symbols, "A" and "B", it really *does* make a difference whether you start from "A" or from "B". Data flow arrows have their arrowheads pointing to the target symbol. Entity relation diagrams need to know where the line started and where it ended. The source entity is always the "parent;" the target entity is always the "child." Some reports also need to know which end of the line is which.

It is easiest to draw the line correctly the first time, but if you make a mistake, you may not have to delete and start again. The **Associative** command under the **Symbol** menu may be all you need to effect a swap on diagrams other than ERDs and Physical Data Models.

### Drawing a Curved Line

You can use the elliptical arc line style to draw a curving data flow line, or you can select Straight - orthogonal and toggle the Round Corners check box in the **Line Style** dialog (**Format, Symbol Format, Line**) as shown below. Rational System Architect draws a segment of an ellipse, which seems to work well for most users.

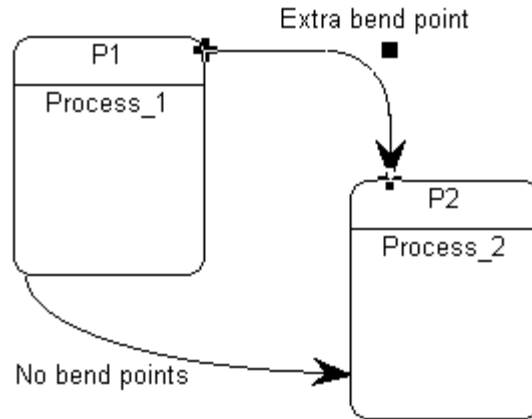


Elliptical arcs follow the same procedures as other lines, with this one addition: At any time, you can click the **right mouse** button. Each click transforms the line from concave to convex to straight.

After a line is in place and you want to come back to it later to change a concave/convex orientation, you must use both mouse buttons. Click to select the line, then point to any of its handles, except the one at its tail. While the **left** button is held down, click the **right** button to affect the line segment *behind* the handle.

There are times when you might like your line to have more curvature than a simple ellipse can provide. In the drawing below, you can see that an easy solution is to put an extra bend point into the line:

Figure 4-3. Putting an Extra Bend Point into a Curved Line



To add extra bend points to a line, use the **Insert Line Segment** Command from the **Format** menu.

### Line Splits and Joins

To create line splits and joins in Rational System Architect, you may use the AND or XOR ("exclusive or") connector provided on the specific diagram's toolbar.

You may also join the lines using the AND or XOR connector, and then make the connector invisible. AND connectors are made invisible by toggling off the **Always Display AND Connector** option in the **Notation** command under the **Format, Diagram Format** menu.

For the UML State Diagram specifically, you may use the AND connector to create splits and joins. Once the third line is connected, the AND symbol automatically disappears.

Figure 4-4. Two Examples of Data Flow Splits

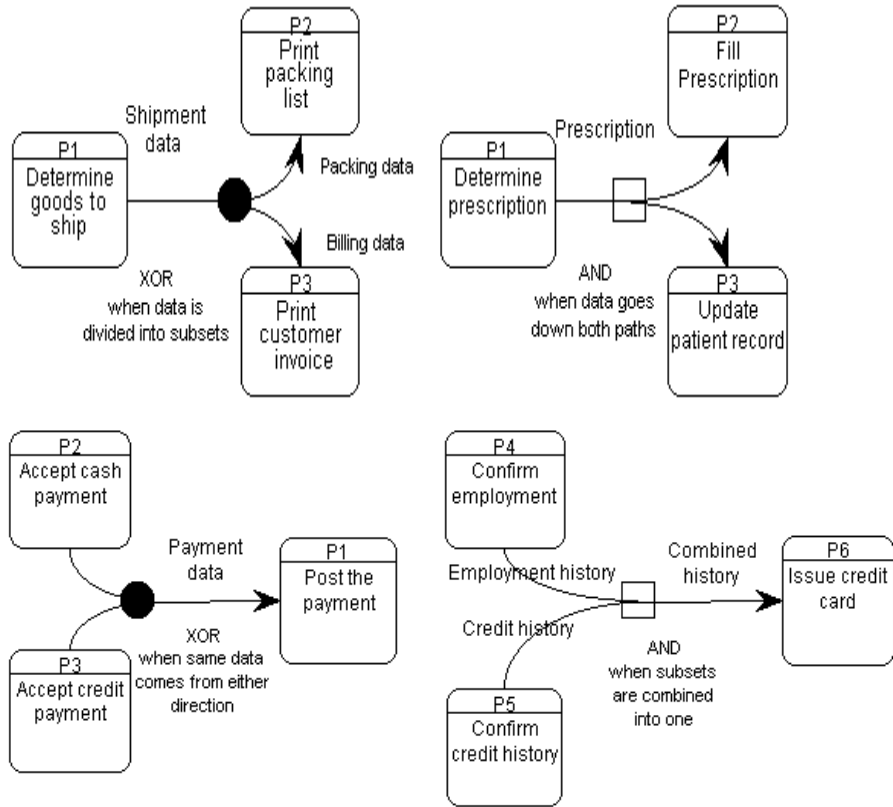


Figure 4-5. Two Examples of Data Flow Joins

---

## Drawing Lines in an Entity Relation Diagram

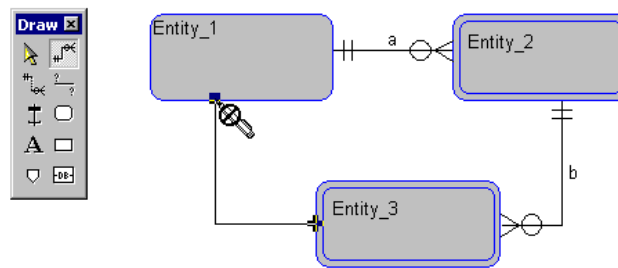
Automatic referential integrity checks mandate that you connect both ends of a relationship line in an Entity Relation diagram.

In addition, Rational System Architect applies several consistency checks when you draw a relation line on a diagram. These checks are designed to ensure that your model can be implemented in a database.

The following situations are illegal:

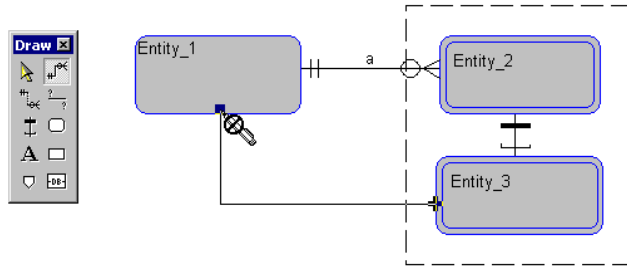
### Cyclic identifying relationships

Figure 4-6. Cyclic identifying relationship is prohibited.



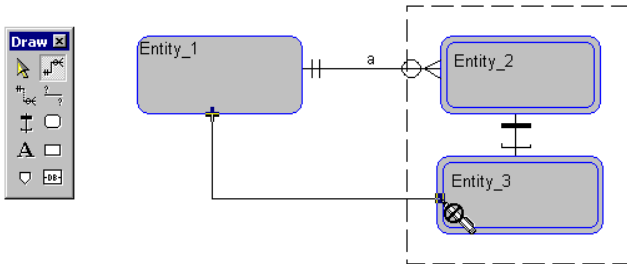
**Super-sub relationships are considered to be identifying and cannot participate in a cycle:**

Figure 4-7. Cyclical super/sub relationships are prohibited.



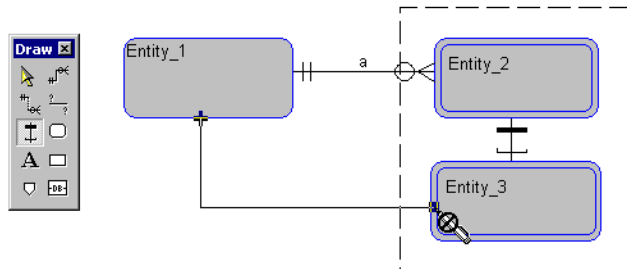
**Sub entities as the child of an identifying relationship:**

Figure 4-8. Sub-entities as the child of an identifying relationship is prohibited.



**Sub entities having more than one super entity:**

Figure 4-9. Sub-entities are prohibited from having more than one super entity.



Please note that these tests are only applied to the current diagram. If you have different relations between entities in different subject



area diagrams, there may be conflicts in the full model. When you create the full model or refresh it, consistency dialogues are displayed for you to resolve the conflicts.

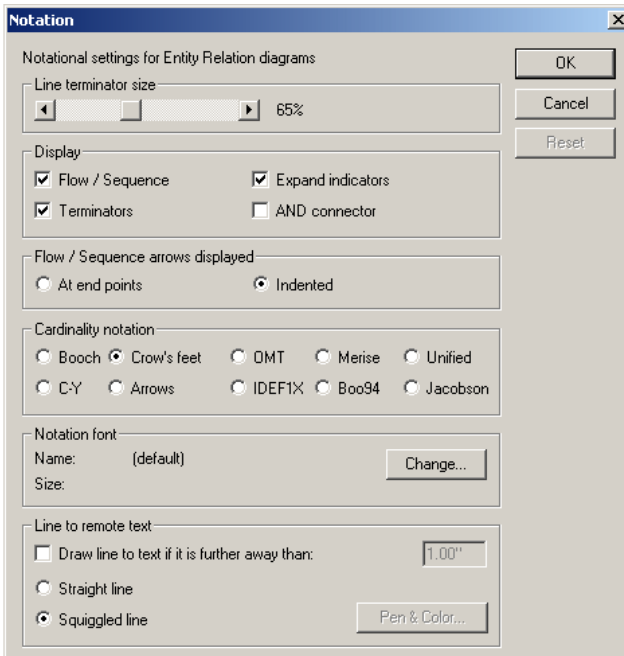
## Diagram Settings

### Diagram-Wide Style Settings

Diagram-wide settings can be made from three dialog screens, all accessed by selecting **Format, Diagram Format**.

The first is the **Notation** dialog, accessed by selecting **Format, Diagram Format, Notation**.

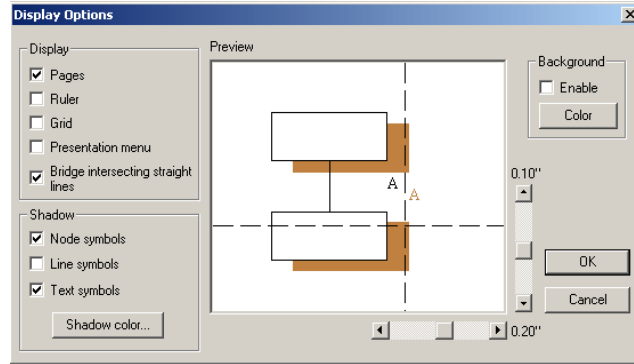
Figure 4-10. The Notation Dialog Affects an Entire Diagram



As you can see, the information on the Notation dialog is concerned with line terminators (arrowheads and crow's feet), the visibility of expand indicators (which show if there are child diagrams and if there are dictionary comments), and the visibility of AND connectors (which are allowed to become invisible when they lie at the junction of 3 or more lines).

The second screen that allows diagram-wide settings is the **Display Options** dialog, accessed by selecting **Format, Diagram Format, Display Options**.

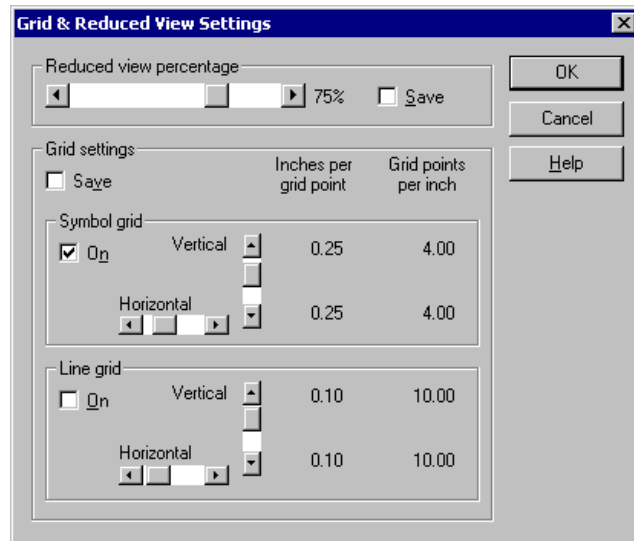
Figure 4-11. The Display Options Dialog Affects an Entire Diagram



The **Display Options** dialog enables you to show pages, rulers, grids, etc on the diagram. It also enables you to put shadows behind symbols, and select the shade percentage.

The third screen that allows diagram-wide settings is the **Grid & Reduced View Settings** dialog, accessed by selecting **Format, Diagram Format, Grid & Reduced View**.

Figure 4-12. The Grid & Reduced View Settings Dialog Affects an Entire Diagram



The grid on/off check boxes and the grid coarse/fine settings are saved in the Format File and affect all diagrams of this type.

## Moving Symbols and Editing a Diagram

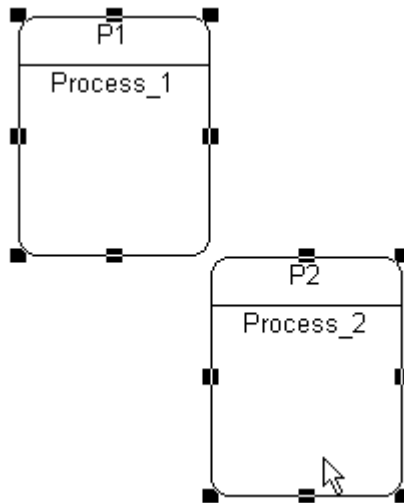
### Selecting and Moving Symbols

Once a symbol has been placed on a diagram, it may be moved, resized, or edited. To select a symbol, simply click on it with your mouse.

### Selecting Symbols

An unexpected result of using bounding rectangles is that two symbols, which seem to be separate when viewed on a diagram, in fact overlap, because their invisible bounding rectangles overlap. A common example is when one symbol's text field falls inside another symbol's boundary. When you click in an overlap area, the system arbitrarily selects one of the symbols. Click again to repeat the selection process, and the various overlapping symbols are selected in turn.

Figure 4-13. The Bounding Rectangles of These Circles Overlap



If the bounding rectangles of multiple symbols overlap one another, the desired symbol may be selected by pressing the **F2** key. The **F2** key selects one symbol after another on the diagram.

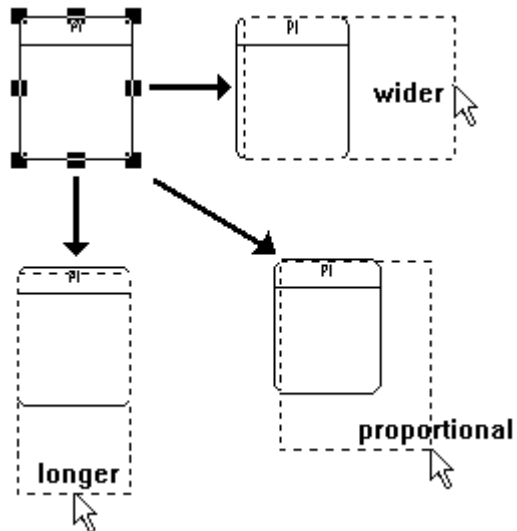
### Selecting Lines

You can select a line symbol by placing the tip of the mouse pointer anywhere along any line segment, or within the arrowhead, or within the line's name block. Elliptical lines are best selected at their endpoints or at their name or label.

## Resizing Symbols

To resize a symbol, drag on one of its handle bars. When you resize a symbol by moving a corner handle, Rational System Architect maintains its proportionality.

Figure 4-14. Example of Resizing a Rectangular Symbol



## Moving Symbols

You may move any symbol or group of symbols on a diagram to any location on the diagram:

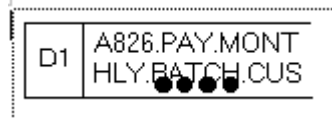
1. Select the symbol(s).
2. Hold the **left mouse** button down while you drag the symbol(s) to the new position.

All lines connected to symbol(s) remain connected. Lines will automatically bend in whatever way needed to retain connections.

## Truncation Indicator

The truncation indicator is a set of four black dots that appear near the bottom of a symbol. It tells you that there is more text to be displayed than there is room for:

Figure 4-15. A Symbol Showing a Truncation Indicator



There are three ways to get rid of a truncation indicator:

1. Make the symbol bigger.
2. Make the font smaller.
3. Suppress the display of all truncation indicators by clearing the Draw Truncation Indicator check-box in the **Preferences** dialog under the **Tools** menu.

---

## Undo Command

You may undo any action that you perform on a diagram by using the **Undo** command in the **Edit** menu. The **Undo** command remembers symbols added or moved on one or more diagrams.

The **undo** command remembers the last x amount of moves or symbol additions, where x is a varying number. The number x depends on the size of the Undo file, the types of moves made, and the size of the diagram or diagrams in which the moves were made.

The size of the Undo file is adjusted through the **Undo File Size** box in the **Preferences** dialog box. The default file size is 131072, which corresponds to 128k of memory ( $128 \times 1024 = 131,072$ ). An Undo file of this size will remember about 20 moves in a relatively small and uncomplicated diagram. You may set the size of the Undo file to any value within the range of 100,000 to 10M (10,000,000). You may also disable the Undo function by setting the size of the file to zero (0).

*Note: Undo will not undo the addition of new symbols on ERDs or PDMs.*

---

## Fonts

Fonts are a complex subject, and a full explanation is beyond the scope of this manual. However, there are some aspects associated with fonts that must be explained here, due to the confusion that they often cause among Windows users.

Much of the confusion is caused by the fact that there are two different types of fonts in Windows: *display fonts* and *printer fonts*. They are distinct from each other and must, therefore, be considered separately. For example, if you have a VGA monitor, and are printing to a Postscript laser printer, you must have at least one display font and one printer font working independently of each other at the same time, because each font is provided and managed by the VGA driver and printer driver, respectively.

### Display Fonts

When Windows was installed on your PC, a set of *display fonts* was installed that was suitable for your monitor. These fonts can be found on your disk with extensions like .FON, .FOT, .TTF, etc. Some fonts are bitmapped, such as Courier, Helvetica, and Times Roman. Bitmapped fonts come in point sizes from 24 down to 8, and are generally considered to look good on your screen. However, they cannot be scaled; the sizes contained in the font files on your disk are the only sizes available for display.

In addition, the installation process gave you some more display fonts, such as Arial, Modern, Script, and Roman. Unlike the non-scalable bitmapped fonts, these vector fonts can be scaled to point sizes ranging from 72 down to 4 (limits set by *Rational System Architect*). Unfortunately, vector fonts don't usually look as nice as the bitmapped ones, and so they are used when getting the correct size is more important than getting the best looking display.

Note that you can go out and buy additional display fonts and add them to your machine. They can be either bitmapped or vector; that is, either fixed size or scalable.

### Printer Fonts

*Printer fonts* are also installed with Windows. Like the display fonts, these can be either bitmapped or vector.

Printer manufacturers frequently give you a drivers and fonts on the installation cd for the printer.



**Display Versus  
Printer Fonts**

Note that the set of display fonts and printer fonts may not have all fonts in common, and, in fact, they rarely do. This means that the style of text you see displayed on your screen may easily not be the style of text that gets printed. However, MS True Type is based on the principle that display and printer fonts match exactly in appearance and name.

Some users solve this problem by choosing to do all their work in Courier, for example, which happens to be a font common to both displays and to many printers. Other users want their printed output to be as nice as possible, and so they choose to work in a good-looking printer font, even though their displays don't look nearly as good as their printouts. And, of course, vice versa is also true.

**Printing  
Diagrams**

As described in the section named **Format File**, each individual symbol on a diagram may have its font setting defined as being either:

- A specific named font and size
- The "default" font and size

When the setting is "default", *Rational System Architect* looks in SA2001.INI to see what the default is prior to displaying or printing that symbol. It look for an entry named `Font=xxxxxx,Display` or `Font=xxxxxx,Printer`, respectively. (If SA2001.INI has no entry, then Arial 10 is used instead. Font entries are made in the SA2001.INI file through the **Font** command under the **Format, Symbol Format** menu.)

**SA2001.INI  
entries**

One problem to be solved is what font and size to use when a diagram has to be reduced before printing, as it might when you specify **Reduced 1 Page**. When a scalable font is being used, this should not be a problem, because the printer driver ensures that the font is reduced correctly and in proportion.

On the other hand, when a bitmapped font is being used, *Rational System Architect* does its best to determine what font size to use. At this point SA goes into SA2001.INI to see if you have given any instructions on how to proceed.

If there are no SA2001.INI instructions on how to reduce, *Rational System Architect* reduces the bitmapped font in steps until it runs out, usually at 8. The font stays at 8 when further reduction calls for 7. It finally shifts over to the scalable Modern font in order to continue with 6, 5 and finally 4.

If you put an instruction in SA2001.INI, it was probably because the step-wise reduction was too coarse for your requirements. For example, 12 point might be too big, and 8 point too small. You need to tell Rational System Architect that Helvetica is acceptable for 12 and 8, but that it should shift to Modern for all the in-between reductions of 11, 10, 9 and 7, 6, 5, 4.

The instruction you put into SA2001.INI is:

```
FontModern=nn, nn, nn, nn
```

The number you use for each *nn* tells what percentage error you are willing to tolerate. `FontModern=1, 1, 1, 1` means that you want the sizes on the printed diagram to be as exact as possible, and that you want Rational System Architect to definitely switch to the scalable Modern font any time it can't find an exact size among its set of bitmapped fonts. Note that normally this entry is not necessary.

---

## Format File

The Format File determines the look of symbols that are drawn on diagrams. In particular, it refers to a symbol's size, shape, line thickness, font, text justification, etc. You can change these symbol characteristics to fit your project's needs.

It is a good idea to establish all your project symbol style standards at the beginning of your project before you start drawing diagrams. Then apply those standards to all your project work in order to give it a consistent look right from the outset. Create one "master" Format File with all desired style settings. Having done this, you may do either of the following: import the Format File into the **FILES** table of each encyclopedia, using Rational System Architect's **Tools, Encyclopedia File Manager**.

1. Put it in a central directory and then specify that path in the SA2001.ini file for each user. To do this you open the SA2001.ini file using a text editor such as Notepad (the sa2001.ini file is stored in the C:\Document and Settings\FORMATFILE = <C>:\Program Files\IBM\Rational\System Architect Suite\11.3\System Architect\Autoexec.sty. In this case the name Autoexec.sty is arbitrary; you may name it something else, as long as the extension remains .sty (ie, Payroll.sty).
2. Name the format file AUTOEXEC.STY and import a copy into the FILES table of all your project encyclopedias (using Rational System Architect's **Tools, Encyclopedia File Manager** command). The format file loaded in the FILES table of the encyclopedia is auto-loaded upon opening of the encyclopedia. It must be named Autoexec.sty.

A given Format File has one slot reserved for every possible symbol that can be drawn with Rational System Architect. Of course, like most users, you use only a small subset of those symbols, and may make your own style settings for an even smaller subset. Thus, if

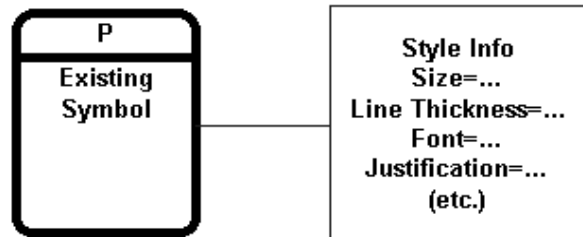
you should create a Format File, it will probably contain just a few settings among a large number of empty slots.

In general, Format Files work as follows:

- If you assign a new Format File to an encyclopedia, and then open an existing diagram, the Format File settings are **ignored** until you select all of the symbols on that diagram, and choose **Format, Format File, Impose Style**.
- Once a new Format File is saved, its settings are applied to the appropriate symbols every time you draw a new symbol of that type on a diagram.
- While most style settings apply to symbols, a few settings apply to an entire diagram.

For example, in the drawing below, a process symbol has been read in from the encyclopedia and is being displayed. You can see that the symbol has a series of style settings that always accompany it. These Format File settings determine the appearance of the symbol on the diagram:

Figure 4-16. An Existing Process Symbol is Being Displayed

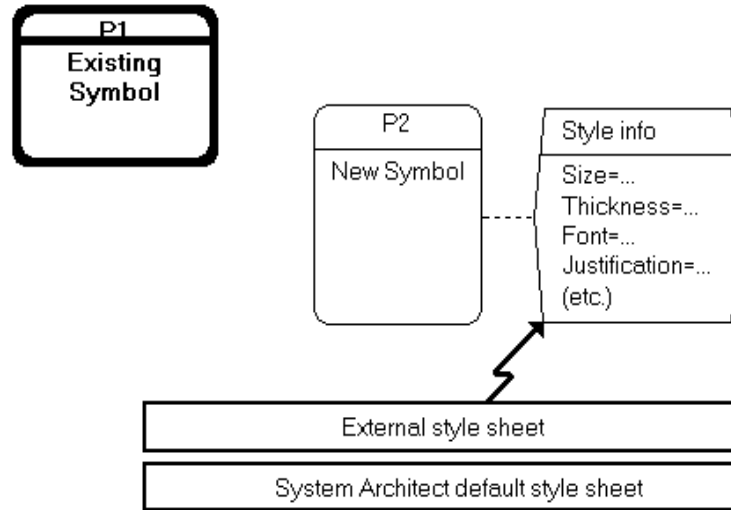


Because each existing symbol carries with it all the information needed for proper display, the presence or absence of a Format File has no effect at all on the display of an existing symbol.

#### When Format Files Have Effect

A Format File comes into play every time you draw a new symbol on a diagram, as shown below:

Figure 4-17. A New Symbol Takes its Settings from the Format File

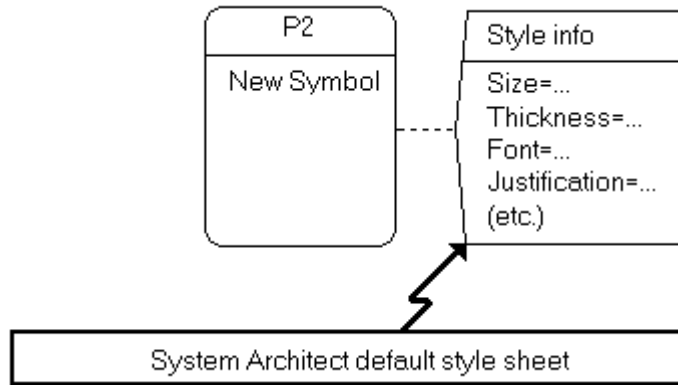


In the drawing, you can see that style settings are immediately attached to the new symbol when it is first drawn, and accompany the symbol from then on. You can also see that those settings were copied from the currently active external style sheet, or Format File.

### The Default Format File

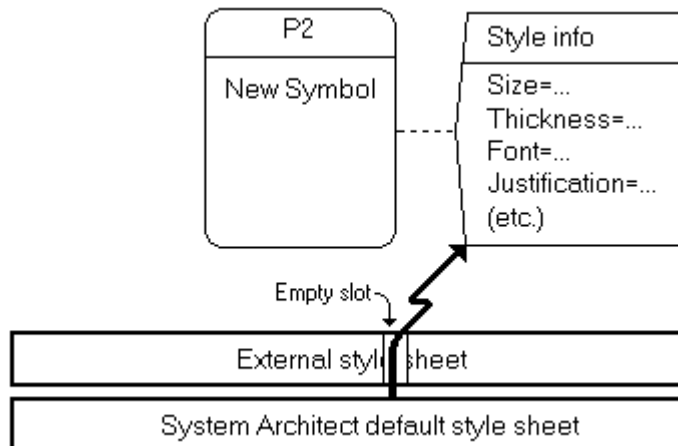
If no external Format File has been loaded, then the Rational System Architect default Format File takes over. Settings for every possible symbol have been hard-coded into Rational System Architect, reflecting the preferences of most users. Except for "font", the settings of the default Format File cannot be altered. If you don't like one or more of the default settings, then you must load in an external Format File with an override setting more to your liking.

Figure 4-18. The Default Format File Takes Over When no External Sheet Has Been Loaded



It is also possible that an external Format File was loaded, but that it has an empty slot for the symbol just drawn. Here, again, default settings are used, as shown in the drawing below:

Figure 4-19. Default Settings are Used When There is an Empty Slot in the External Sheet



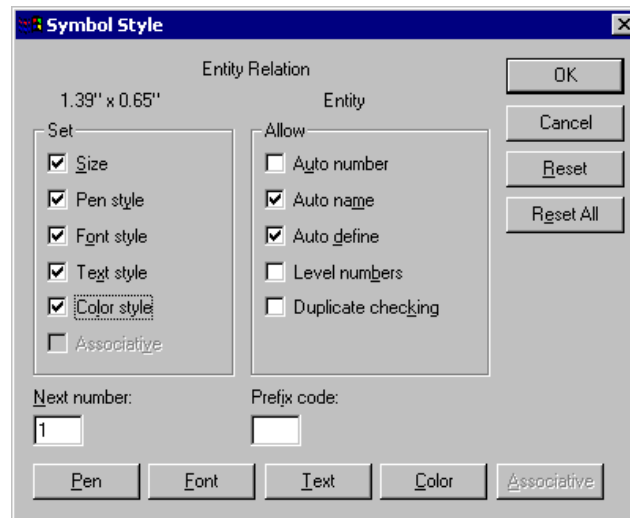
To summarize, every time a new symbol is drawn, it picks up settings taken from the active Format File. If no external Format File has been loaded, settings are taken from the hard-coded default Format File.

When you save the diagram, the new symbols are saved in the encyclopedia with their style data attached. They maintain their size and shape whenever they are displayed in the future, even if no Format File is present.

### Creating a New Format File

Assume you want to tailor the style of entity symbols for your project. Perform the following steps:

1. Draw an entity symbol and alter its size, shape, thickness, and font as required.
2. Click on the entity symbol to select it. Select **Format, Symbol Format, Symbol Style**.
3. Toggle on the check boxes shown in the picture below.



4. Click **OK**.

Figure 4-20. Check the Settings to be Added to the External Format File

A new Format File is initialized with its *Process* slot filled in (other slots are left empty). At this point, you can save the Format File, by selecting **Format, Format File, Save As...** You may also wait and save the Format File at the end of their session. Rational System Architect automatically prompts you if you wish to save the Format File when you close the encyclopedia.

### Applying the Format File to Existing Symbols

### Clearing a Symbol's Style Settings

With a Format File loaded, simply select one or more old symbols to be altered. Then pull down the **Format** menu and click on **Format File, Impose Style**.

If needed, you can clear out or "reset" any symbol's style settings and change them back to the system defaults. It is a two-step process:

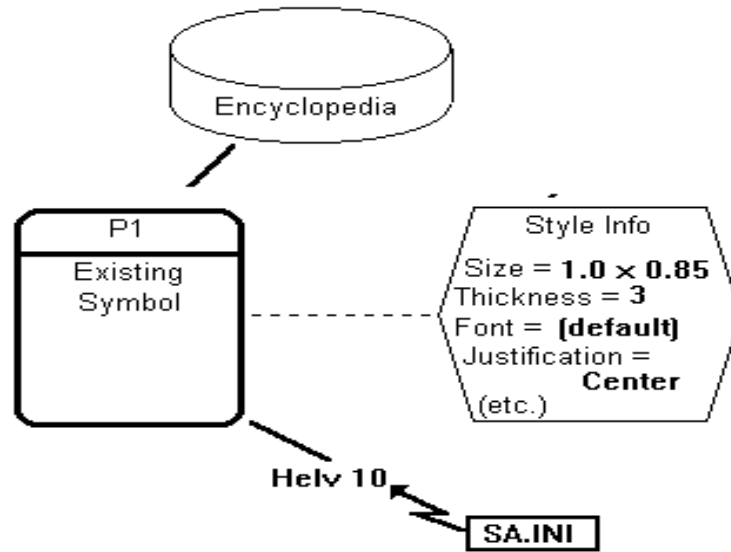
1. First ensure that no external Format File is present. Do this by pulling down the **Format** menu and clicking on **Format File, Reset**. If a Format File was present, it is cleared from the computer's memory, leaving behind only the hard-coded default Format File.
2. With only the default Format File present, you can select one or more symbols. Pull down the **Format** menu and click on **Format File, Impose Style**. The symbols' prior settings are overlaid with the default Rational System Architect settings.

### Format File Fonts are Special

Take another look at this symbol coming in from the encyclopedia prior to its being displayed. All its attached style information is known in detail, except for "font", which is simply marked as "default":

Figure 4-21. This Symbol's Style Settings are Known in Detail Except for its Font



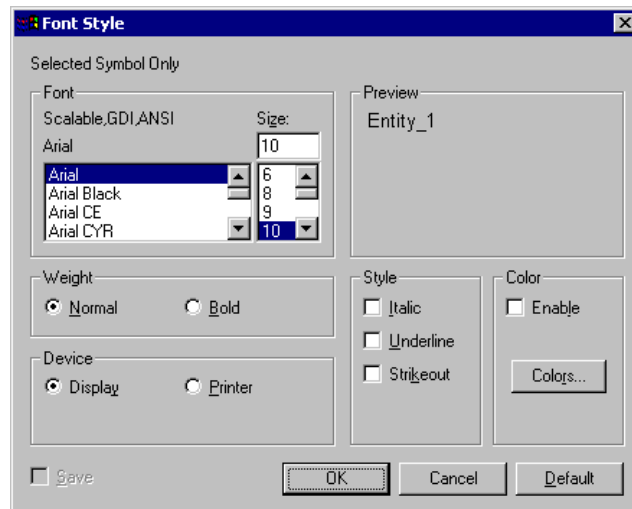


In this context, "default" means, "I'll decide later what this symbol's font should be." *Later* is when a diagram is about to be displayed or printed. At that point, Rational System Architect looks in SA2001.INI to see what the "default font" is and displays/prints the symbol with that font.

If you were to change the default font in SA2001.INI, this symbol's appearance on the diagram would change immediately. If there is no entry in SA2001.INI, Rational System Architect uses either Arial 10 or Helvetica 10.

If a symbol has a specific font attached to it, and you want that symbol to use the default font instead, invoke the **Font** dialog from the **Symbol Style** option under the **Format** menu and click on **Default**:

Figure 4-22. Click on Default to Clear a Symbol's Font Setting



### Loading and Saving Format Files

Invoke the **Format, Format File, Import Style Sheet** command to import any Format File into the FILES table of the current encyclopedia.

### AUTOEXEC.STY

The best name to give a Format File is AUTOEXEC.STY. If a sheet with this name is present in the encyclopedia's FILES table, it is loaded in automatically as soon as you open that encyclopedia. A Format File with any other name must be loaded manually with the **Format, Format File, Import Style Sheet** command.

---

## Copying Diagrams

The easiest way to make a copy of a diagram is by way of the clipboard:

1. Use **Select All** under the **Edit** menu or **CTRL+A** to select all symbols on the source diagram.
2. Use the **Copy** command under the **Edit** menu or the **Copy** icon in the Toolbar to copy the selected symbols into the Clipboard.
3. Use the **New Diagram** command under the **File** menu or the **New Diagram** icon in the Toolbar, to start up a new, empty target diagram. Normally the new target should be the same type as the source, but see below for a possible exception.
4. Use the **Paste** command under the **Edit** menu to populate the target diagram with the Clipboard symbols.

*You cannot copy a diagram by a command line `COPY` of its `.DGX` file. A `.DGX` file contains most, but not all, of a diagram's contents. A command line copy causes encyclopedia files to get out of synch.*

At some sites, a library of diagram templates is maintained. These templates are set up at the beginning of a project, and analysts are instructed to always start their work from these diagrams. This is meant to ensure adherence to company standards. In this case, the Rational System Architect "glossary" is used in place of the Clipboard. Please refer to the on-line help system for details on how to copy a diagram from the glossary (**Tools** menu, **Diagram Glossary, Open**).

You can drag all the symbols from one diagram onto another by selecting them on the Docking Explorer. However, they will not be displayed as they are on the source diagram.

<b>Using the Clipboard</b>	<p>You can copy, cut, and paste to and from the Windows Clipboard.</p> <p>Copying and cutting require that you first select the item. You can select a block of text without a mouse by holding down the <b>shift</b> key while pressing the left and right arrow keys.</p> <p><b>Copy:</b> Ctrl + C</p> <p><b>Cut:</b> Ctrl + X</p> <p><b>Paste:</b> Ctrl + V</p>
----------------------------	--

## Keyboard Accelerators

Many of the most commonly used commands have shortcut keys, officially known as "accelerator keys". When an accelerator key is pressed, it is the exact equivalent of using your mouse to select the command.

Most users of Rational System Architect rely on mouse operations, and rarely refer to the keyboard accelerators tables. However, there are a few accelerator keys that are quite useful and really should be learned. These are marked with an asterisk.

The accelerator key assignments are listed below.

Table 4-1. Keyboard Accelerators by Key

Command	Menu	Key
HELP	HELP	F1
PRINT (Diagram)	FILE	Ctrl + P
NEW DIAGRAM	FILE	Ctrl + N
CLOSE DIAGRAM	FILE	Ctrl + W
SAVE DIAGRAM	FILE	Ctrl + S
RESTORE POINTER		Esc
UNDO	EDIT	Ctrl + Z
REDO	EDIT	Ctrl + Y
CUT	EDIT	Ctrl + X
COPY	EDIT	Ctrl + C
PASTE	EDIT	Ctrl + V
DELETE	EDIT	Del
SELECT ALL	EDIT	Ctrl + A
SELECT NEXT	EDIT	F2
FIND SYMBOL	EDIT	Ctrl + F
REDRAW	VIEW	F3
REFRESH EXPLORER	VIEW	F5
ACTUAL SIZE	VIEW (Zoom)	F8

<b>Command</b>	<b>Menu</b>	<b>Key</b>
REDUCED 75%	VIEW (Zoom)	F9
FULL PAGE	VIEW (Zoom)	F6
ZOOM IN	VIEW (Zoom)	F11
ZOOM OUT	VIEW (Zoom)	F12
TEXT POSITION	FORMAT (Symbol Style)	Ctrl + T
ALIGN LEFT	FORMAT (Align)	Shift + F2
ALIGN RIGHT	FORMAT (Align)	Shift + F3
ALIGN TOP	FORMAT (Align)	Shift + F4
ALIGN VERTICAL CENTER	FORMAT (Align)	Shift + F7
ALIGN HORIZONTAL CENTER	FORMAT (Align)	Shift + F6
ALIGN BOTTOM	FORMAT (Align)	Shift + F5
RUN MACRO	TOOLS (Macro)	Alt + F8
VBA EDITOR	TOOLS (Macro)	Alt + F11



# 5

---

## *Working with Definitions*

### **Introduction**

This chapter describes how to create and work with definitions in *Rational System Architect*.

<b>Topics in this chapter</b>	<b>Page</b>
What are Definitions in <i>Rational System Architect</i> ?	5-2
Symbols and Definitions	5-3
Browse, Select, and Drag	5-5
Using Grids	5-6
Handling Data Definitions	5-8
Text, Descriptions, and Comments	5-17
Importing and Exporting Definitions	5-20



---

## What are Definitions in Rational System Architect?

A Rational System Architect project encyclopedia contains **diagrams**, **symbols**, and **definitions**. A diagram contains symbols, each of which has an underlying definition.

Many symbol types share a single definition type. For example, there are several types of entity symbols (entity, associative entity, weak entity) that are all served by a single entity definition type.

On the other hand, some definitions in the encyclopedia are not represented by any symbol, such as requirements, attributes, methods, columns, data elements, etc. These are referred to as non-symbol definitions.

### Diagrams, Symbols, and Definitions Have Properties

Each diagram type, each definition type, and each symbol type has one or more underlying **properties**. For example, a Class diagram has a diagram property for *Navigation Presentation Mode*, a class symbol on a class diagram has a property called *Hide Details*, and a class definition has properties such as *attribute* and *method*. An attribute is itself a definition which contains properties such as type, access, etc.

Each diagram type, definition type, and symbol type in Rational System Architect has a *default* set of properties declared for it; you may modify these properties, or add to them, using Rational System Architect's extensibility mechanism (see the *Extensibility Guide* or the on-line help concerning USRPROPS.TXT).

### Diagrams, Symbols, and Definitions Can Be Worked on Separately

Diagrams, symbols, and definitions can be created and worked on quite separately. You are free to design an entire system, entering the requirements, business goals and objectives, business rules, data, entities, minispecs for processes, classes, and methods *without ever drawing a single diagram*. Or, you may draw diagrams and place symbols without defining them.

## Symbols and Definitions

Whenever you place a symbol on a diagram, and add a definition to it (right-mouse click on it and select **Edit**), you will see within its Edit dialog a tab labeled **Symbol**. Information you place within properties on the **Symbol** tab are for the particular symbol you have just dropped on the diagram. Each symbol that you drop on a diagram is considered an instance of the definition. Information added to the **Symbol** tab belongs to that instance only; if you drop down another symbol of the same type and name on the diagram, or another diagram, the information added to the **Symbol** tab of the first instance will not appear in subsequent instances.

The rest of the tabs within the Edit dialog represent the symbol's **definition**. A change to any of the properties within these tabs applies globally to the definition within the encyclopedia.

Figure 5-1 The Class Symbol is defined by Class Definition.

The screenshot shows the 'Model Object - Class - Reservation' dialog box. The 'Symbol' tab is selected, showing a class symbol with properties: status, date, duration, roomType, create(startDate, duration, roomType, customerName), calculatePrice(discountPercent), getDetails(startDate, duration, roomType, customerName), allocateCustomer(customerRef), allocateRoom(roomNumber, guestName), allocateRoom(roomType, guestName), makeProvisional, confirm, exercise, cancel, and persistent. The 'Operations' tab is also visible, showing a table of operations.

	Name	Formal Parameters	Category	Concurrency
1	create	startDate, duration, roomType,		
2	calculatePrice	discountPercent		
3	getDetails	startDate, duration, roomType,		
4	allocateCustomer	customerRef		
5	allocateRoom	roomNumber, guestName		
6	allocateRoom	roomType, guestName		

Annotations in the image include:

- Class Symbol (instance of the class definition that appears on this diagram)**: Points to the class symbol on the diagram.
- Class definition, partitioned by tabs**: Points to the dialog box.
- Symbol property tab**: Points to the 'Symbol' tab.
- Class symbol defined by Class definition, some properties of which are displayed on the symbol**: Points to the list of operations in the 'Symbol' tab.
- Properties of the class definition**: Points to the 'Operations' tab.
- Values for the properties entered by the user**: Points to the 'Operations' table.

### What's Going On Under the Covers

When you define a symbol, that definition is added to the encyclopedia and is bound to the symbol, making a bound pair. The bond relies on the entries in the encyclopedia's underlying **Relationship** table, which has two rows for that relationship:

- RESERVATION [*class definition*] **defines**  
RESERVATION [class symbol]
- RESERVATION [class symbol] **is defined by**  
RESERVATION [*class definition*]

Note that no matter how many class symbols you have with the name RESERVATION, there is only one definition. Each symbol has a unique 32-bit ID, as does each definition.

**Note:** Understanding the relationships between items in Rational System Architect's metamodel, and the Relationship table, is not necessary for modeling in Rational System Architect. However, familiarity with the relationships is necessary if you want to define your own reports, using either the internal reporting system or the link to Microsoft Word, or if you want to modify the meta-model using USRPROPS.TXT. All the information you'll need is in the on-line help for the reporting system, under *Relationships between Objects*.

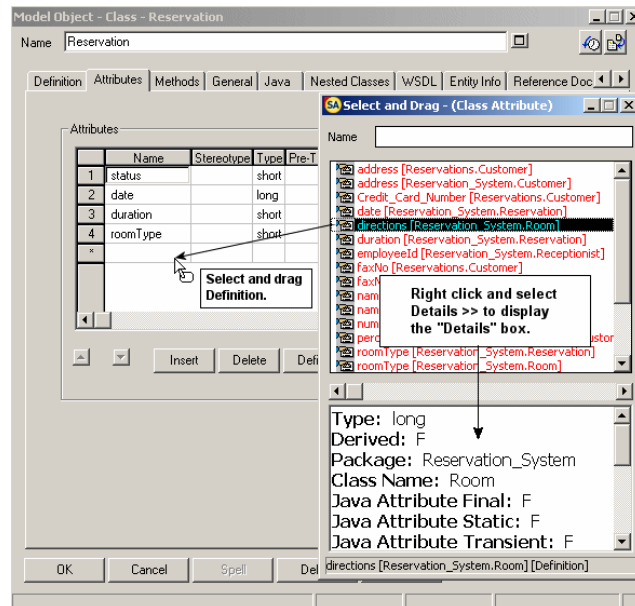
## Browse, Select, and Drag

A **Select and Drag** dialog is opened whenever the **Choices** button is clicked on any definition dialog.

To move the listed objects into the edit box in the definition dialog box, select one or more objects and drag them with your mouse. If you want to select contiguous items, hold down the SHIFT key and click on the top-most and bottom-most of the items to be selected. If you want to select non-contiguous items, hold down the CTRL key, and click on each item in turn.

Certain information about a selected item in the **Select and Drag** dialog is displayed in the details box. To activate it, right-click on the area in the **Select and Drag** dialog, and from the drop-down list, select **Details>>**; to close it, select **No Details** from the drop-down list. If a number of items are selected, the last one selected is displayed.

Figure 5-2 The Dictionary Object Dialogs for Names and Other Information



---

## Using Grids

Some Rational System Architect definition dialogs are displayed within grids. These grids display much of the information of a definition, all in one table that can be easily viewed and edited. See Figure 5-3 below.

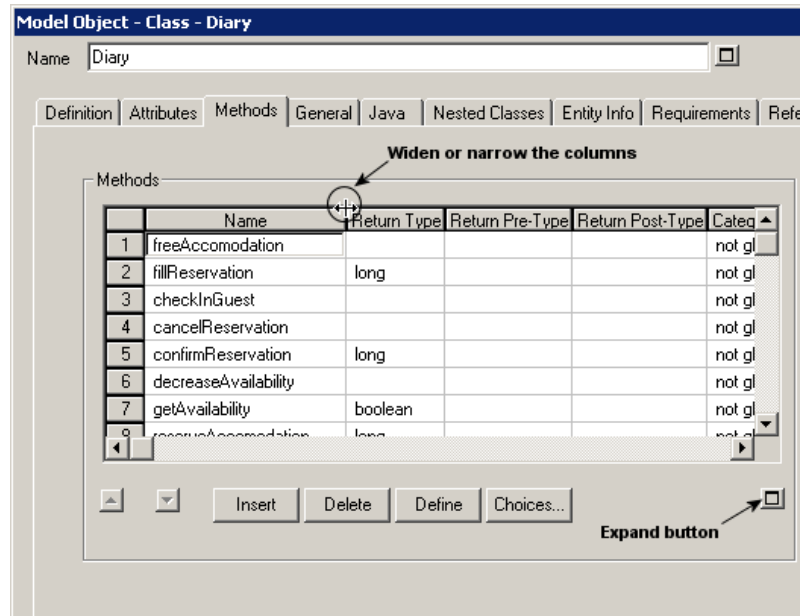
You may edit a definition through its representation in the grid, or edit the definition through traditional dialogs by putting your cursor in any part of the definition's row in the grid, and clicking on the **Define** button.

### What Is Not Shown in the Grid

Most definition properties that are specified as being a text string greater than 100 characters are not shown in the grid. To get at a definition's full set of properties, select any element of the definition in the grid and click on **Define**.

List properties are also shown in the grid; to display the choices of a list, click on the grid element in question and a drop-down list selector will appear within the grid element.

Figure 5-3 The Dictionary Object Dialogs for Names and Other Information



### Adjusting the Grid

Columns can be widened or narrowed in the grid by selecting a column separator in the grid title bar and dragging it to the right or left.

The grid area itself can be temporarily expanded by clicking on the expand button (displayed as a square) in the bottom right-hand corner of the grid. The grid will expand to the full size of the dialog. If any properties exist under the grid, they will temporarily be hidden from view, until the grid is unexpanded, or the user moves off that tab, and then back onto it.

To move a definition up or down in the grid, select the entire row of the definition and click on the up or down arrows in the bottom left-hand corner of the grid. Selecting the entire row of a definition is best achieved by selecting the number of the row, listed in the left-hand-most column in the grid.

---

## Handling Data Definitions

Rational System Architect provides for the creation of an underlying data dictionary containing data elements, data structures, and data domains. Within each entity of a data model, there are also attributes.

In general, the data in an entity is defined by data elements. They can be created in the entity, or copied in from the underlying data dictionary. An attribute is regarded as an instance of a data element. It resides in an entity and provides certain instance information about the data – for example, whether or not it is a primary key. A data structure is a grouping of data elements. A data element can optionally report to an underlying data domain.

### Unique Names in the Data Dictionary

The names of all entries within the dictionary are not necessarily unique within individual entry types. There is one exception to this rule: the names for data elements and data structures must be unique across both types within the dictionary.

Here are some examples:

Allowed:

Data Store definition:"Customer"  
Entity definition:"Customer"  
Data Element definition:"Customer"

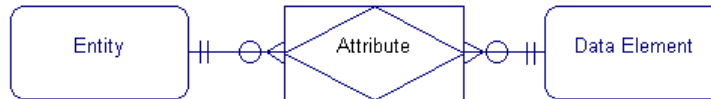
Not Allowed:

Data Element definition:"Customer"  
Data Structure definition:"Customer"

## Attributes in Entities

An attribute is defined in Rational System Architect as the association between the entity and the data item.

Figure 5-4 How an attribute is defined in Rational System Architect.



The attribute definition maintains the characteristics that apply only to the instance of the data within the entity:

- key status - is this instance of the data a primary key component?
- nullity - can the column formed by this data item contain a null value in the database?
- uniqueness - does this attribute, on its own, represent a candidate key for the entity?
- column name
- database comments
- extended attributes (PowerBuilder)

### Primary Key

The *Primary Key* is an attribute, or set of attributes, that is used to uniquely identify an instance of the entity in which it resides. Since the primary key component(s) are the identifier, their values may not be null.

### Foreign Key

A *Foreign Key* is an attribute of an entity that is a primary key component in a related entity. The foreign key is a product of a parent/child relationship; it resides in the child entity, and is dependent on the primary key of the parent entity.



---

## Data Elements and Data Structures

### Data Elements

The data element definition specifies the characteristics that apply to all instances of the data, regardless of what model or entity it is used in:

- data type
- type qualifiers - length, precision
- default value
- check constraints
- data owner
- domain

It is important to note that, since the physical characteristics belong to the data element, they are *always* inherited by *all* attributes used to represent the data.

*If the property values of a data element are modified, all related attributes in all models, are affected by the change.*

### Data Structures

Another type of data entry in the data dictionary might be "CustomerOrder", which is defined as "Information taken over the phone when a customer places an order". This is an example of a **data structure**. It is defined at a fairly high-level, and clearly hides a number of details.

If you are a vice-president, you might be satisfied with this definition as is. But if you are a programmer, you want to know more details: "Exactly what is the information that makes up a given customer order?" You are expecting an answer such as:

```
CustomerOrder consists of
CustNo +
CustName +
CustAddressBlock +
OrderDate +
```

Thus, a **data structure** is a group of other data items. A structure can always be expanded to a more detailed level. Most of the items listed above are **data elements**, meaning you reached the lowest level of detail that is meaningful for this project. One exception is the item named `CustAddressBlock`, which is in fact a structure within a structure. At the next level down `CustAddressBlock` is defined:

```
CustAddressBlock consists of
CustHouseNo +
CustStreet +
CustCity +
CustState +
CustZip
```

In *Rational System Architect*, you may have structures within structures without limit. However, try to be careful not to inadvertently create a recursive structure, where A contains B contains C contains A. *Rational System Architect* watches for such conditions and points them out when it finds them. When you have recursive structures, results can be unpredictable.

Data elements and data structures are global to the encyclopedia. They are not restricted to a specific Project Data Model, but can be used by any entity in any diagram in any Project Data Model to create attributes. Data elements and structures can also be used by other objects in the encyclopedia that are defined as "data," including data stores, data flows, and processes on data flow diagrams.

Search on *Data Modeling* in the on-line help for more information.

---

## Using Data Domains

### What is a Data Domain?

A **domain** is used to supply common (physical) properties to a number of data elements. Things like *social security number*, *phone number*, *zip code*, are domains. All social security numbers, whatever their use, have the same format and the same editing rules, and would probably be entered in fields of the same type on an input form.

The same is true of phone numbers (at least in the U.S.) and of zip codes. Domains, therefore, may be used, at the discretion of the analyst/designer, to reduce effort by supplying one place in which common information may be stored and maintained.

Data domains act as a place to store project-wide standards and rules about data formats. For example, if your project were to decide in advance that all dates must be stored in the form "yyyymmdd" (e.g., December 31, 1997 = 19971231), the data domain named *Standard-Date* must be added to the encyclopedia.

Later, as various date-oriented data elements are being added to the encyclopedia, such as *Date-Entered*, *Date-Of-Birth*, *Date-Hired*, etc., each one can be given the value *Standard-Date* for the property *Data Domain*. Therefore, if *Standard-Date* were to be given the properties of 8 numeric characters, all other dates derived from it would also automatically be numeric 8.

### Data Domains Vs Data Elements

The data domain allows you to specify the domain that a data element belongs to. When you specify a domain for a data element, all of the properties that are defined for the domain are applied to the data element.

When using the default data domain configuration options, the allowable properties that can be defined for a data domain conform to the SQL '92 Standard:

- data type
- type qualifiers - length, precision
- default value
- check constraints

This style of data domain corresponds to a user defined data type in the physical data model.

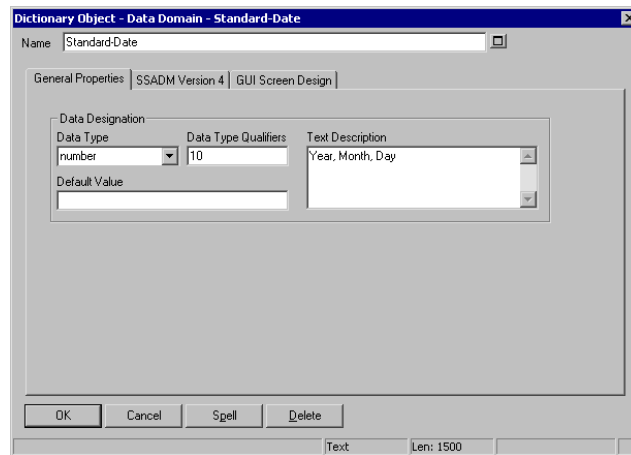
Rational System Architect provides an advanced configuration option, which enables you to specify full inheritance from data domains. When this option is selected, all properties of data elements can be derived from data domains and multi-level domain inheritance is permitted.

**Example 1:  
Defining  
Standard-Date**

To define Standard-Date, follow the steps described below:

1. Click on the **Dictionary** menu, **New Definition** command.
2. In the Browser, select *Data Domain* as the definition type.
3. Click with the **right mouse** button, and select **New**.
4. In the **Dictionary Object <Type> <Name>** dialog, enter *Standard-Date* in the name text box.
5. Click on **OK**.
6. In the expanded **Dictionary Object <Type> <Name>** dialog, select the Data Designation *Data Type*, enter the *Data Type Qualifiers* (length), *Default Value*, and *any Text Description* that might be useful.

Figure 5-5. Sample Dialog for adding Data Domain definitions

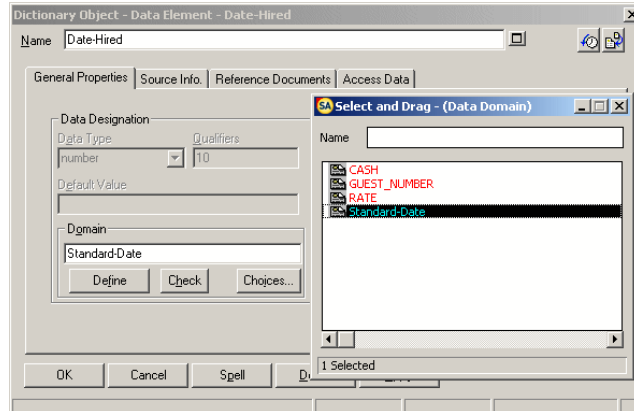


**Example 2:  
Defining *Date-  
Hired***

1. Click on the **Dictionary** menu, **New Definition** command.
2. In the Browser, select *Data Element* as the definition type.
3. Click with the **right mouse** button, and select **New**.
4. In the **Dictionary Object <Type> <Name>** dialog, type *Date-Hired* in the **Name** text box.
5. Click on **OK**.
6. In the expanded **Dictionary Object <Type> <Name>** dialog, click the **Choices** button in the text box labeled **Domain - (Data Domain)**.
7. Select *Standard-Date* from the list of defined **Data Domains**, and drag it into the **Domain - (Data Domain)** text box.
8. Alternatively, you may simply type *Standard-Date* in the text box, but in general it is easier to drag a defined object than to remember exactly how it is spelled.

9. Notice that as soon as you drop the data domain name into the **Domain - (Data Domain)** text box, the **Data Designation Data Type** and **Data Type** qualifiers on the data element change. If you change the values in the data domain, the values in the data element will change, too.

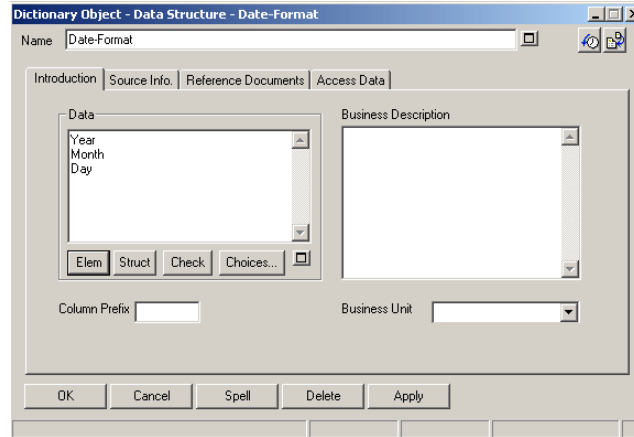
Figure 5-6. Sample Dialog for Adding a New Data Element Definition, Based on a Data Domain



10. It is only necessary to enter those values of the properties of the Data Element that are different from those of the Data Domain from which it is derived, such as Business Unit, which is not a meaningful property for Data Domain.

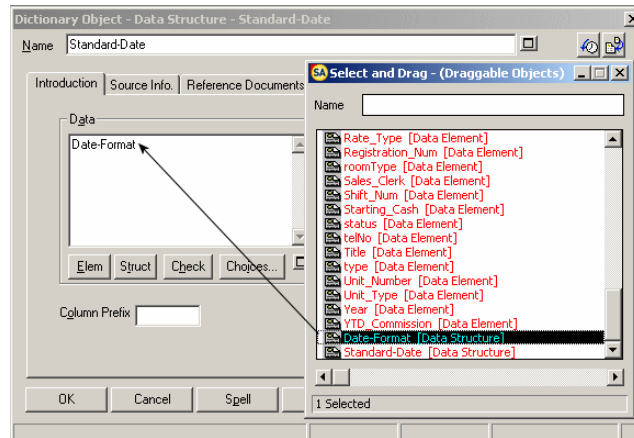
Actually, the above example with *Standard-Date* as a date object of no length was a bit too simple. In reality, it should have been numeric 4 + 2 + 2. In order to achieve this, we first need to create a Data Struct Domain, *Date-Format*, which is defined by three data elements, *Year*, *Month*, and *Day*. Data struct domains are to data structures as data domains are to data elements.

Figure 5-7. The Data Struct Domain *Date-Format*, Defined as Consisting of Three Data Elements



We can now define a data structure, *Standard-Date*, completing the value of the property Domain with *Date-Format*.

Figure 5-8. The data structure *Standard-Date*, defined as inheriting values from the data struct domain *Date-Format*



In addition to dates, other candidates for standardization are people's names (e.g., they should always be last + first + middle initial) or phone numbers (e.g., they should always be 3 + 3 + 4 digits).

---

## Text, Descriptions, and Comments

In *Rational System Architect*, there is a dictionary entry for **each** instance of any one **symbol**, but there is only **one** dictionary entry for that symbol's **definition**. If the data store symbol *Customer* appears 3 different times, the dictionary has 3 separate entries of the symbol *Customer*, and one entry of the definition *Customer*.

### Attaching Text to a Symbol

Four types of textual data can be attached to symbols in *Rational System Architect*:

#### 1. Dictionary Definition

The primary place for any value of any property is the symbol's **definition**. It is the first place that someone would look to learn more about the symbol.

If a given symbol, such as the *Customer* data store, has to appear on several different diagrams, only one *Customer* dictionary definition is required, since that one definition can serve all instances of the data store.

#### 2. Graphic Descriptions – the Graphic Comment

The Graphic Comment is for annotations that you want to actually appear on the diagram.

Note that each instance of the *Customer* data store can have its own graphic comment. Thus, a typical usage for graphic comments to show how each instance of the symbol on a diagram varies from another instance of that same symbol on another diagram. Enter the graphic comment text via the Symbol tab on the **Definition** menu.

Graphic comments can be displayed inside the symbol or outside. Use the **Format, Symbol Format** menu, **Symbol Style, Text Position** command.

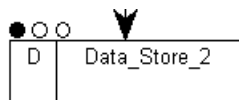
#### 3. Dictionary Comments

Dictionary Comments are virtually identical to Graphic Comments: there is one comment for each instance of the symbol. The difference between the two is that dictionary comments don't appear on the diagram. Their existence is signaled by the appearance of the



"traffic lights" on the upper left-hand corner of the symbol. The left-most dot of the three is filled in. To add a comment to a symbol, select **Comment** from the symbol's floating menu or the **Comment** command on the **Dictionary** menu.

Figure 5-9. The "traffic lights" adornment on a symbol



#### 4. Adding Text from the Draw Menu

There is a fourth type of text. Each **Draw** menu has a **Text** command that allows you to place text on any diagram. Unlike the three types of text described above, this text is not attached to any symbol. An example is a title you place at the top of a diagram.

For instructions on including textual data on reports, please refer to the on-line help. Search on *Text, Draw Menu*.

---

## Spell Check

The *Rational System Architect* spell check feature is available through a **Spell** button located in all **Diagram** and **Definition** dialog boxes. To use the spell check feature, perform the following steps:

1. Place your cursor in a field within the dialog box that you wish to check.
2. Click on the **Spell** button.
3. If a potentially misspelled word is encountered, a dialog appears showing possible alternatives. Select the replacement word from the list or type a correction in the **Replace With** field.
4. Click on the **Replace** button to change the word, or click on **Ignore** to leave the word unchanged.

More information on the spell checker is available in the on-line help. Search on *spell check*.

---

## Importing and Exporting Definitions

Rational System Architect provides a number of facilities for importing and exporting information to and from an encyclopedia. Information in a Rational System Architect encyclopedia consists of two basic components: the graphical diagram information and the underlying symbol definitions.

SA's Import/Export facilities include:

- Importing/Exporting Definitions via **CSV** or **Text** Format
- Importing/Exporting graphical diagram and definition information via **XML**
- Using SA's **native VBA support and published Object Model** to interface to other tools
- Merging/extracting between other Rational System Architect encyclopedias
- Importing database designs into a physical data model through the Reverse Data Engineer and generating schema from a data model using the Schema Generator
- Importing Java, C++, or Visual Basic code into a Class diagram and generating code from a class diagram (C++, Java, VB, CORBA IDL, Smalltalk, Delphi, etc)
- Reverse engineering an XML file representing a Document Type Definitions (DTD) or BizTalk design into an XML Design diagram

---

## CSV and Text Import Export

Rational System Architect provides a generic import/export mechanism that enables definitions to be import/export from/to external files in either **Text** format or **Comma Separated Value (CSV)** format. This generic facility is accessed by selecting **Dictionary, Import Definitions** or **Dictionary, Export Definitions**.

A graphical design represented by a diagram in Rational System Architect consists of two basic components: the graphical diagram and the underlying symbol definitions. The generic import/export facility enables import/export of the definitions only – not the graphical components of the diagrams.

**Note:** Rational System Architect provides a number of other facilities for importing and exporting definitions to and from an encyclopedia, which facilitate import/export of both the definitions and the graphical components of diagrams. This includes merge/extract between other Rational System Architect encyclopedias, importing database designs into a physical data model through the DB Reverse Engineering, generating schema from a data model using the DB Schema Generation, importing C++ or Java code into a Class diagram, generating code from a class diagram (C++, Java, VB, CORBA IDL, Smalltalk, Delphi, etc), or importing/exporting definitions to the Microsoft Repository. Please see the on-line help for more information on any of the above topics.

### Using the Import Function

To import a definition into Rational System Architect through CSV format, it is imperative that the columns of the data in the CSV file match one-to-one with the properties of the definition that the information is being imported into.

To import a .CSV file into Rational System Architect, perform the following steps:

1. Determine which non-DBMS-related portions of the definitions of the entities, attributes, and so on in the source tool you want to export to Rational System Architect.

2. Modify USRPROPS.TXT to add those properties to the definition type. For example, let's assume the source tool has the properties "Notes" and "General Information" in the entity definitions. The modification to USRPROPS.TXT might look like this:

```
DEFINITION "Entity"  
{  
  PROPERTY Notes { Edit Text LENGTH 1500 }  
  PROPERTY "General Information" {Edit Text  
LENGTH 750 }  
}
```

3. In the source tool, export the non-DBMS-related portions of the definitions.

In most tools, you can run a report with output as a Comma Separated Value (CSV) file.

4. In Rational System Architect, select **Import Definitions** from the **Dictionary** Menu. Use the **Browse** button to pick the path of the CSV file.
5. Select **Entity** as the type. Click on **OK**. You'll get a report of all the entities that were imported. You can print it, or save it as a .TXT file.
6. Set the **Browser** to definitions. Select **Entity** as the Type.

As you select the entities that were reverse engineered, the values from Notes and "General Information" should appear.

### Using the Export Function

If you need to export Data Dictionary Definitions, please refer to the on-line help for information on *Dictionary* menu, **Export Definitions** command. The **Export Definitions** command allows you to export files in ASCII or in CSV format. Otherwise, if you have other requirements for exported data, continue below.

---

## Using the Clipboard

If you have a diagram currently displayed, simply select one or several or all of its symbols, and use the **Copy** command under the **Edit** menu to copy them into the Clipboard. (Or use the **Copy** icon on the Toolbar.) From there, you should be able to insert the diagram into any product that lets you paste in data. Diagrams can be copied to the Clipboard in either bitmap or metafile format, using the **Clipboard Format** command under the **Edit** menu. Most applications can accept one of these choices.

Textual data can also be copied to the clipboard from most dialog windows. One example might be the contents of the **Dictionary Objects <Type> <Name>** dialog, which might contain a lengthy minispec. Another example might be draft output from a report. First, use your mouse to select the text so that its colors become inverted to white-on-black. If the dialog has a **Copy** button, click it. Otherwise, press the **Ctrl-C** key combination.

Graphical data can also be captured by pressing the **Print Screen** button on your keyboard. This takes a "snapshot" of *everything* currently being displayed on your monitor. The snapshot can then be pasted into a paint program where it can be retouched and annotated. *Alt-Print Screen* can also be used to take a snapshot of a smaller portion of the monitor display. This snapshot consists of just the active front-most window or dialog.

---

## Importing/Exporting Via XML

XML is the eXtensible Markup Language that provides a standard format for data exchange. Rational System Architect allows you to export all information from its repository, or encyclopedia, to XML format, and to import information in XML format into an encyclopedia. What is generated (or imported) is called an XML instance document.

The instance document contains the data, or information, you are concerned with – in this case all of the information you and other users have modeled in Rational System Architect. The instance document generated from Rational System Architect relies on a DTD (Document Type Definition) for its structure. Rational System Architect's DTD is located in the main software directory, <C>:\Program Files\IBM\Rational\System Architect Suite\11.3\System Architect\SAXML.dtd.

### Exporting To XML

To export a diagram or definition to XML, perform the following steps:

1. Using the browser, select a single diagram or multiple diagrams, or a single definition or multiple definitions, using standard Windows techniques of selecting while holding down the Shift key.
2. Right-mouse click and select **Export xml...** from the drop-down floating menu. The **Export XML** dialog will open.
3. Specify whether or not you want to generate the selected objects (diagrams, definitions, or symbols) only, or if you want to include subordinate definitions or child diagrams.
4. Toggle on whether or not you want to generate to a **Single File**, or to **Multiple Files**. If you choose multiple files, every diagram or definition that you have selected will be generated to a file of the same name, with a naming prefix that you specify.
5. Specify whether or not you want to include pictures of diagrams in your XML output.

6. Specify an output directory and file name (if you are generating to a single file) or naming prefix (if you are generating to multiple files).
7. Click **Next**.
8. Confirm your export options and click **OK**.

The XML will be output to the path you have specified, and you will receive a message telling you the XML generation was successful.

## Importing XML

To import XML containing Rational System Architect information (and conforming to the saxml.dtd), perform the following steps:

1. From the browser, select the diagram or definition **type**, right-mouse click and choose **Import XML** from the drop-down floating menu. (An example of *diagram type* would be **UML Use Case**, *not* the specific name of a UML Use Case diagram, such as Make Reservations.)
2. In the **Import XML** dialog, browse to the directory that contains your XML file(s). Select the file(s) and click **Open**.
3. Specify collision options in the **Rational System Architect – Import XML** dialog.
4. Click **OK**.

The diagram(s) or definition(s) will be added to the Rational System Architect encyclopedia





# 6

---

## *The Matrix Editor*

### **Introduction**

The Rational System Architect Matrix Editor can be used to enter information on the models before a single diagram is drawn. By attacking analysis in this way, you concentrate on a wide view of the problem, and the dependencies of information, before delving into more detailed analysis and design.

The Matrix Editor can be used for all types of modeling in Rational System Architect, including Enterprise and IDEF Business Modeling. You may also create your own matrices and add them to the Matrix Browser.

<b>Topics in this chapter</b>	<b>Page</b>
The Matrix Editor	6-2
“X” in Cell Matrices	6-5
Text In Cell Matrices	6-8
Multi-Dimensional Matrices	6-12
Creating Matrices	6-19

---

## The Matrix Editor

### Matrix Editor

Matrix Editor provide a different entry point for entering information into the project encyclopedia, in addition to the normal techniques of using diagrams and definitions.

From an analysis standpoint, entering data through matrices provides the user with a wide view of the problem domain and the information that the models can/will contain, and the intersections of dependent definitions.

### Synchronized Definitions

Matrices provide another view of the same information contained within symbol definitions. Information entered into the Matrix Editor is automatically entered into symbol definitions that appear on diagrams; changes to information in either place are synchronized – the Matrix Editor simply provides another view of the information.

The Matrix Editor is synchronized with definitions in Rational System Architect in either of the following ways, depending on the specific definition and Matrix Editor:

- **Input through Matrix Editor and Definition Dialogs:** Most matrices allow information to be entered either through the matrix editor or within the definition dialog for one or either of the definition types.
- **Input Forced to Matrix Editor Only:** Some definitions have been restricted so that they can only be entered through matrices – the corresponding property dialog for entering a new definition within a symbol, is read only. Many matrices provided for Business Process methodologies behave this way. The reasoning behind this is that these methodologies require certain definitions to be entered only on the global level.

### Where Cross-Reference Definitions Reside

The Matrix Editor presents a two-dimensional view of data. To say a definition of type X is related to a definition of type Y means that:

- The definition dialog for X contains a list of Y definitions, or
- The definition dialog for Y contains a list of X definitions, or
- Both of the above, or
- None of the above, that is, although the two definitions are cross-referenced, this view of the information is not (for some methodological reason) presented to the user in the definition dialog of either definition.

### Matrix Types

There are three matrix types available in Rational System Architect:

**“X” in Cell** – restricts input so you may only click on or off an “X” in the cell where column and row definitions intersect. The presence of an “X” in a cell indicates a relationship exists.

**Text in Cell** – enables you to enter text in the cell where column and row definitions intersect. The presence of text in a cell indicates a relationship exists.

**Multi-dimensional** – enables you to view, in a tabular layout, multiple “X” in cell type matrices that contain overlapping definitions. For example, a matrix in one tab cross-references definitions A and B, a matrix in a second tab cross-references B and C, and a matrix in a third tab cross-references A and C.

### Adding a New Definition Through the Matrix Editor

You may add a new definition to either the column or the row of the Matrix editor.

To add a new definition of the type represented by the column, right-mouse click in any column in the matrix. You are presented with a dialog that enables you to add a new definition of that type.

To add a new definition of the type represented by the row, right-mouse click in any column in the row and add the new definition in the dialog presented.

The newly entered definition is added to the bottom of the list of matrix column or row definitions. You may move a definitions position within the list by left-mouse clicking on it, and keeping your mouse down, moving it to another position in the list.

#### **Modifying a Definition Through the Matrix Editor**

You may modify a definition via the matrix editor. Right-mouse click on a definition in a column or row, and select **Modify Row** or **Column Definition**.

#### **Sorting the List of Definitions in a Row or Column**

As mentioned above, you may move a definition's position within the list by left-mouse clicking on it, and keeping your mouse down, moving it to another position in the list.

#### **Sorting By Property Values**

You may also sort a column or row based on property values of definitions. Select **Format, Row** (or **Column**), **Sort**. A dialog will open that lists all of the properties of the respective definition type. You may select a property and choose whether to sort the definitions via ascending or descending order of property values.

For example, you could choose to sort Elementary Business Processes by a property such as Initial Audit – the Business Processes would be listed by the name of the users who first created them.

#### **Printing Matrices**

You may print matrices via standard Windows **Print** and **Print Preview** functions available via the **Matrix** menu.

---

## "X" in Cell Matrix

"X" in Cell matrices indicate whether a relationship exists between two definitions. The existence of a relationship is indicated by an "X" in the cell that intersects two definitions – an "X" indicates a relationship exists; a blank indicates no relationship exists.

"X" in Cell matrices are simple cross-reference matrices that place an "X" in the intersection cell between two definitions. The result of the cross-reference can have the following implications: Each Row definition may store a list of the Column definitions. Each Column definition may store a list of the Row definitions. If each Row definition stores a list of Column definitions, and each Column definition stores a list of Row definitions then the intersection information is duplicated. Because of this, you will typically see that the 'related definition' properties in both definitions are read-only.

### Sample "X" in Cell Matrix: Role to Competency

An example of an "X" in cell matrix is the **Role to Competency**, as shown in figure 6-1 below. The steps that follow describe how to open this matrix and assume that the **Samples** encyclopedia is open and that you have enabled the **Business Modeling** methodology. If it is not enabled, click the **Tools** menu, select **Customize Method Support, Encyclopedia Configuration** and toggle on the **Business Modeling** checkboxes.

Figure 6-1. Example of an "X" in Cell Matrix – Role to Competency.

Role	Competency	Computing	Filing	Oral	Organizing	Telephony	Typing
Account Computer 1							
Accounts Clerk							
Booking Coordinator		X	X	X	X		X
Conference Organizer				X	X		X
Customer							
Customer Services Rep				X	X	X	
Receptionist							
Receptionist 1							
Receptionist 2							
Sales Clerk			X			X	X
Sales Person				X		X	X

To open the **Role to Competency** “X” in cell matrix proceed as follows:

1. With the Samples encyclopedia opened, in Rational System Architect, select **View, Matrix Browser**.

2. The **Matrix Browser** appears, on the **Organization tab** select **Role to Competency**.
3. Click **Next** to bypass the **Matrix Filters** dialog.
4. Click **Finish** to bypass the Matrix Filters dialog (leaving the default – all columns and row definitions selected). The **Role to Competency Matrix** is presented.
5. To cross-reference a column and a row definition, click on the intersecting cell between the two definitions. This places an "X" in the cell. To remove an "X" in the cell, click on the cell.
6. When finished viewing click the X in the upper right-hand corner of the matrix dialog to close the matrix. The matrix will automatically be saved for you.



---

## Text in Cell Matrices

Text in Cell matrices enable you to enter text in the cell that intersects two definitions. An intersecting cell has a definition in its own right in the repository. Usually, it is named by appending the names of the definitions and separating them with forward slash (e.g., EBP/Role, Function/Activity, Message/Stimulus etc.).

In a Text-in-Cell matrix, the “**X**” (for Show Intersections) button in the upper row of the matrix toolbar will be disabled (see figure below). This enables you to enter and see text in the cells of the matrix. Clicking the “**X**” button on will reload the matrix and any cell with text in it displays as an ‘**x**’ instead.

Matrix columns and rows may be dragged on to widen them, enabling you to show as much of the text as you like.

The result of a cross-reference can have all the implications of the “**X**” in cell matrix plus the following:

- An intersecting Cell definition stores the text, up to 255 characters
- The intersecting Cell definition is named according to the Row definition, and is keyed by the Row and Column definitions.
- Each Row definition stores a list of related Column definitions
- Each Column definition stores a list of related Row definitions

**Sample Text in  
Cell Matrix: EBP  
to Entity**

An example of a Text matrix is the **Elementary Business Process to Entity** matrix, as shown in Figure 6-2 below. You may enter text in the cells to describe how a process works on a particular entity. You may, for instance, specify if it **Creates** it, **Reads** it, **Updates** it, or **Deletes** it. Using verbs such as these makes this what is often called a **CRUD** matrix. The steps that follow describe how to open this matrix and assume that the **Samples encyclopedia** is open and that you have enabled the **Business Modeling** methodology. If it is not enabled, click the **Tools** menu, select **Customize Method Support, Encyclopedia Configuration** and toggle on the **Business Modeling** checkboxes.

Figure 6-2. Example of  
a Text Matrix --  
Elementary Business  
Process to Entity.

Dictionary Reports Window Help

39%

'X' button enables you to type text in cell. If turned on, text is represented by an 'X'.

Elementary Business Process to Entity

Elementary Business Process	Entity	Customer	Guest	Overseas Customer	Owner	Property	Receptionist	Reservation	Room	Shift	Travel Agent	UK Customer	Vehicle
Check Payment Details				Read				Read				Read	
Check Reservation Details				Read				Read	Read			Read	
Check Room Availability								Read	Read				
Check Room for Availability													
Customer Agrees to Terms													
Customer Rejects Terms													
Examine Period to Accom Date								Read					
Inform Customers Company								Read					
Make Cancellation Charge	Update							Update					
Make Full Room Rate Charge	Update							Update					
Make Percentage Room Rate Charge	Update							Update					
Notify Customer of Credit Problem													
Notify Customer of Inavailability													
Notify Customers of Inavailability													
Process Room Booking													
Provide Client with Reservation Number													
Provisionally Book Room													
Release Room	Update							Delete	Update				
Reservations and Booking Funcs													
Reserve Room	Update							Create	Update				
Store Customer Details	Update			Read				Update	Update			Read	
Take Customer Details													
Take Payment Details	Update							Update					

You may drag on column or cell sides to increase/decrease widths

In this matrix, we see CRUD (create, read, update, delete) verbs specifying how a process uses an entity.

Reserve Room process 'creates' Reservation entity.

To open the **Elementary Business Process vs. Entity** Text in cell matrix proceed as follows:

1. With the Samples encyclopedia open in Rational System Architect, select **View, Matrix Browser**.
2. The **Matrix Browser** appears, on the **Business Enterprise tab**, select **Elementary Business Process to Entity**.
3. Click **Next** to bypass the **Matrix Filters** dialog.
4. Click **Finish** to bypass the Matrix Filters contents dialog (leaving the default – all column and row definitions selected). The **Elementary Business Process to Entity matrix** is presented.
5. Click on the **'X'** (for Show Intersections) button in the upper row of the matrix toolbar. To cross-reference a column and a row definition, click on the intersecting cell between the two definitions and enter text. This text can be output later to a Word report.
6. You can alternate between an "X" in Cell and Text in cell view by clicking the **'X'** (for Show Intersections) button in the Matrix toolbar.

---

## Multi-dimensional Matrices

Definitions in Rational System Architect can be part of more than one matrix. Consequently, a cross-reference in one matrix can infer a cross-reference in another. A Multi-dimensional matrix (MDM) displays, on a single Matrix Editor window, inter-related matrices that share definitions and where a cross-reference in one matrix infers a cross-reference in another.

In a sense, a MDM is not really a matrix type. It is actually a collection of related "X" in cell type matrices viewed simultaneously. Each matrix within a MDM can also be viewed on its own. Because MDMs are created by linking individual "X" in cell matrices, any "X" in cell matrix can be part of part of a MDM.

### The Infer Cross-Reference Option

Multi-dimensional Matrices provide the Infer option, which automatically places a temporary "X" in cells where a cross-reference may be *inferred*, based on a cross-reference made in another matrix that is part of the same MDM.

Inferred relationships are temporary until you perform one of the following:

- Click the **Save** button in the toolbar—this changes the color of the "X" in the cell from blue to black, thereby making the relationships permanent.
- Click the **Reload** button in the toolbar—this removes all inferred relationships from the matrix currently in focus.
- **Close** the MDM— this discards relationships that are still in the Inferred state.

**Sample Multi-dimensional Matrix:  
Elementary Business Process to Application**

An example of Multi-dimensional Matrix is the **Elementary Business Process to Application matrix** as shown in Figure 6-3 below. In Rational System Architect the following matrices are linked via multiple dimensions: Elem Bus Process to Application, Elem Bus Process to Technology and Technology to Application. The steps that follow describe how to open this matrix and assume that the **Samples** encyclopedia is open and that you have enabled the **Business Modeling** methodology. If it is not enabled, click the **Tools** menu, select **Customize Method Support, Encyclopedia Configuration** and toggle on the **Business Modeling** checkboxes.

Figure 6-3. Example of a Multi-dimensional Matrix.

ctionary Reports Window Help

39%

Clicking on the 'Infer Relationships' button automatically fill in cells of inferred relationships with an 'X' colored blue.

Application to Technology

App to Tech EBP to Tech EBP to App

Elementary Business Process	Technology	E-Mail	Fax	Letter	PC	Terminal	Windows 95	Windows98
Advise Customer								
Calculate Room Price		X	X	X	X		X	X
Check Customer Credit								
Check Payment Details		X	X	X	X	X	X	X
Check Reservation Details		X	X	X	X			
Check Room Availability		X	X	X	X			
Check Room for Availability								
Customer Agrees to Terms								
Customer Rejects Terms								
Examine Period to Accom Date					X			
Inform Customers Company		X	X	X	X			
Make Cancellation Charge		X	X	X	X			
Make Full Room Rate Charge		X	X	X	X			
Make Percentage Room Rate Charge		X	X	X	X			
Notify Customer of Credit Problem								
Notify Customer of Inavailability								

If you Save the matrix, this inferred relationship will become a formal relationship --the 'x' will turn color from blue to black, the technology '.NET' will be added as a reference property value to the EBP definition 'Calculate Room Price', and Calculate Room Price' will be added as a reference property value to the technology definition '.NET'.

To open the **Elementary Business Process to Application** matrix proceed as follows:

1. Click the **View** menu, **Matrix Browser**.
2. The **Matrix Browser** appears, on the **Business Enterprise** tab select **Elem Bus Process to Application** . All three matrices above are

represented in the same Matrix Editor window, within tabs.

3. Click **Next** to bypass the **Matrix Filters** dialog.
4. Click **Finish** to bypass the Matrix Filters contents dialog (leaving the default – all column and row definitions selected). You'll need to do this for each matrix to load into the window.
5. Click **Finish. Elementary Business Process to Application** matrix is presented.
6. Click the **In** button on the toolbar to Infer cross-references. Inferred relationships will display a blue-colored "X" in the intersecting cells. This text can be output later to a Word report.
7. Click **Save** button to save inferred cross-references (optional).  
OR  
Click the **Reload** button on the toolbar to remove inferred cross-references (optional).



---

## Mirrored Matrices

A Mirrored Matrix can have the same row and column definition types displayed as mirrored. When you click in the cell of the matrix, the opposite relationship will also have the cell marked with an 'x'. The relationship becomes a two-way relationship. When using a non-mirrored matrix which has the same row and column definition types, you can only specify one way relationships.

An example of a Mirrored matrix is the Application to Application Interface matrix.

To open the Application to Application Interface matrix proceed as follows:

With the **Samples** encyclopedia open in Rational System Architect, select **View, Matrix Browser**.

The **Matrix Browser** appears, on the **Application tab**, select **Application to Application Interface**.

Click **Next** to bypass the **Matrix Filters** dialog.

Click **Finish** to bypass the Matrix Filters dialog (leaving the default – all column and row definitions selected). The Application to Application Interface matrix is presented.

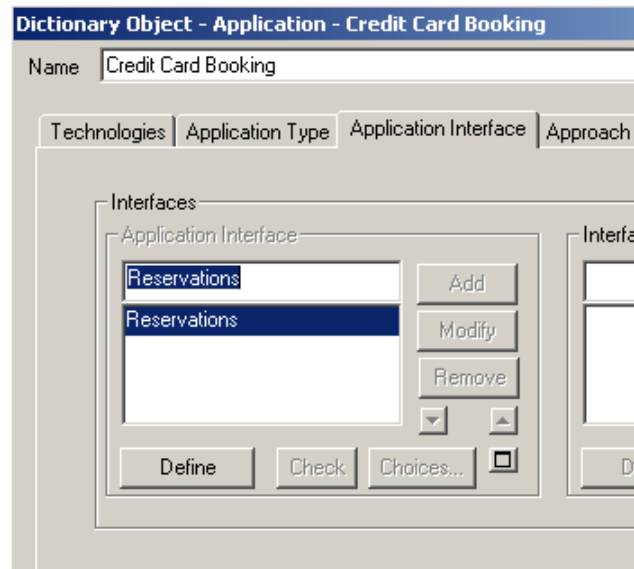
Select the cell between **Credit Card Booking** and **Reservations**. An 'X' in cell will mark the relationship. Automatically an 'X' appears in the opposite relationship between **Reservations** and **Credit Card Booking**.

Application	Credit Card Booking	Customer Maintenance	Reservations
Application			X
Credit Card Booking		X	
Customer Maintenance	X		
Reservations	X		

Save the matrix by selecting **Matrix, Save** or by clicking the 'x' in the upper-right hand corner of the dialog.

From the **Encyclopedia Browser**, select the + mark next to **Definitions** and open the **Application** definition called **Credit Card Booking**.

Select the **Application Interface** tab and notice that **Reservations** has been added. The opposite will appear in the Reservations definition.



Select **OK** to close the definition dialog.

---

## Creating Matrices

Matrices created by users are referred to as User-defined matrices since they contain row and column definitions specified by a user. You can create any matrix type and select any definition type, as long as they can cross-reference each other. User-defined matrices are placed in the Matrix Browser, under the User Defined tab.

The type matrix and the level of customization you want to apply determine the steps you need to take to create your matrix. For example, you can create a definition that acts as a container for your relationships. Suppose you create a matrix with definitions X and Y. By customizing SA, you can create a definition type named X/Y. This way, for every X and Y definition that you cross-reference, an X/Y definition type is created.

Some variables to consider when creating matrices include the following:

- Definitions to be cross-referenced
- Matrix type and function – will you force input matrix only, or will you allow cross-referencing to be done through definition dialogs and the matrices
- Customization -- Should the dialog for definition X contain a field for related Y definitions, and vice versa? Will only one definition dialog display its related definition, or will neither?
- If a definition dialog will display its related definition, what is the name of that property? For our example, the **Component** definition dialog will contain a property drop-down field called **Related Requirements**, listing **Requirement** definitions.
- Is one or both of the definitions keyed to other properties?

## Creating a User-Defined Matrix

Creating matrices in general is a three step process in which you perform the following tasks:

**Part I** – Plan the Matrix as stated above.

**Part II** — Edit **USRPROPS.TXT** to support the definition relationships of the matrix. Depending on the type of matrix, you may also need to create an intersecting cell definition. See online help topic *What is USRPROPS/SAPROPS*.

**Part III** – Creating the Matrix Using the Matrix Designer

Once you complete these steps, you can run your matrix. Each part is described in the sections that follow.

## Part II – Edit USRPROPS.TXT

The second step to building a matrix is to create the USRPROPS.TXT entries that will support the definition relationships of the matrix. For example we will create a matrix of UML Components vs System Requirements. The first thing we do is open the USRPROPS.TXT file.

In Rational System Architect, select **Tools, Customize User Properties, Export USRPROPS.TXT (Encyclopedia)**.

In the **Export User Properties** dialog, select a target folder and click **Save**. The file will be opened automatically from that location.

The Component definition type already exists as a default in all Rational System Architect encyclopedias, however the System Requirement definition type is a new definition type that we will add to the encyclopedia.

Add the following code to USRPROPS.TXT:

**Rename Definition "User 2" To "System Requirement"**

**Definition “System Requirement”**

```
{
PROPERTY "Related Components"
{ ZOOMABLE EDIT ListOf "Component" LENGTH 1200
READONLY}
}
```

Although the Component definition type already exists, we want to add a property called Related Requirements into the definition of a Component, which will contain a list of the System Requirement definitions.

Adding the following to USRPROPS.TXT:

**Definition “Component”**

```
{
PROPERTY "Related Requirements"
{ ZOOMABLE EDIT ListOf "System Requirements" LENGTH
1200 READONLY}
}
```

Now we need to create a definition that will contain information concerning the intersection of the two related definitions.

Add the following lines to USRPROPS.TXT:

At the top of the file, add:

**Rename Definition “User 3” To “Component/System Requirement”**

At the bottom of the file, add the following:

```
DEFINITION “Component/System Requirement”
{
PROPERTY “RowDefinition”
{KEY EDIT OneOf “Component” RELATE BY “is part of”}

PROPERTY “ColumnDefintion”
```

{KEY EDIT OneOf “System Requirement” RELATE BY “is part of”}

PROPERTY “Description”

{EDIT Text LENGTH 255 HELP “Appears in the cell of a matrix”}

PROPERTY “Intersection?”

{ EDIT Boolean LENGTH 1}  
}

Save and close the USRPROPS.TXT file.

Import USRPROPS.TXT from the **Tools**, menu, **Customize User Properties, Import USRPROPS.TXT ( Encyclopedia)** command.

Reopen the encyclopedia (File, Encyclopedia Open) in order for the changes to take effect.

### **Part III – Creating the Matrix Using the Matrix Designer**

Run the **Matrix Designer** by selecting **Tools, Matrix Designer**. The Matrix Designer is a form presented within Rational System Architect’s diagram workspace frame – it will appear within the diagram workspace areas, and by default be attached to the diagram workspace frame.

**Note:** To detach the Matrix Designer from the workspace frame, select **Window, Cascade**, or **Tile Horizontally**, or **Tile Vertically**.

### General Matrix Settings

In the General tab of the Matrix Designer, specify the name of the matrix, the name of its menu command that will invoke it off the Tools menu, and the type of matrix, by filling out the following properties:

- **Menu Caption** – Specify the name of the matrix’s menu command that will invoke it off of Rational System Architect’s **Matrix Browser, User Defined tab**. You may place an ampersand (‘&’) anywhere in the name, directly preceding any letter that you might want to act as a ‘hotkey’ to invoke the matrix via your keyboard. Menu entry, appears in the sub user menu.
- **Matrix Title** – Specify the name of the matrix – it will appear as a heading in the matrix itself.
- **Matrix Type** – Specify the type of matrix:  
SAGenericMatrixProc – standard generic matrix
- **Short Title** – Specify a short name for the tabs that will appear on a Multi-Dimensional matrix.

Additional Commands of the Matrix Editor:

**Scroll Matrix** – Scrolls through all the user defined matrix entries. The number of the currently selected matrix is shown to the immediate right of the scroll bar.

**Delete Entry** – Deletes the currently selected matrix.

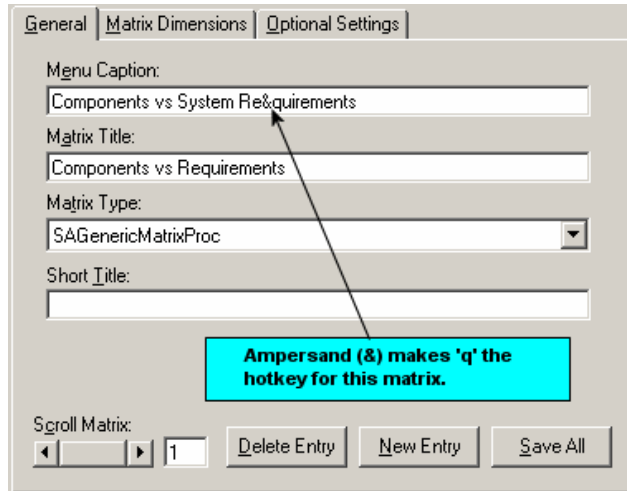
**New Entry** – Adds a new matrix

**Save All** – Saves all of the user defined matrices.



### Example (continued)

In our example, we fill in the following properties of the **General** tab of the Matrix Designer:



The screenshot shows the 'General' tab of the Matrix Designer. The 'Menu Caption' field contains 'Components vs System Re&quirements'. The 'Matrix Title' field contains 'Components vs Requirements'. The 'Matrix Type' dropdown menu is set to 'SAGenericMatrixProc'. The 'Short Title' field is empty. A red callout box with the text 'Ampersand (&) makes 'q' the hotkey for this matrix.' points to the ampersand in the 'Menu Caption' field. At the bottom, there is a 'Scroll Matrix' section with a left arrow, a right arrow, a '1' in a box, and buttons for 'Delete Entry', 'New Entry', and 'Save All'.

### Matrix Dimensions

In the **Matrix Dimensions** tab of the Matrix Designer, specify the definitions that will be involved in the matrix, and the properties within each that will list the related definitions. You may also specify the definition type that will hold cross-reference cell information:

**Column** – specify the definition that will be in the columns of the matrix.

**Property** – specify the name of the property in the column definition that will store the list of related (row) definitions (ie, the property that will store cross-reference information).

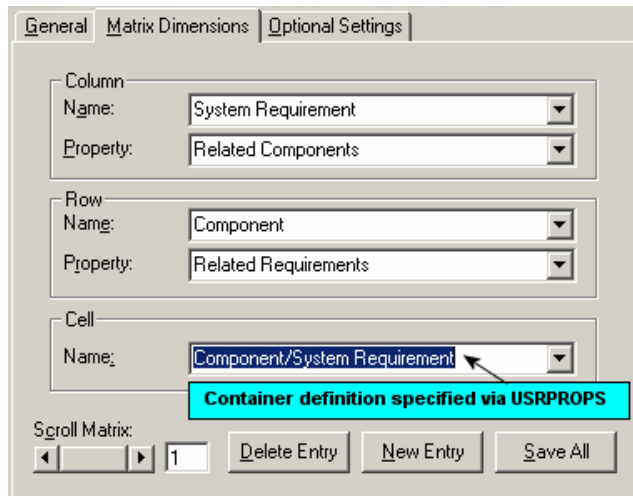
**Row** – specify the definition that will be in rows of the matrix.

**Property** – specify the name of the property in the row definition that will store the list of related (column) definitions (ie, property that will store cross-reference information).

**Cell** – specify the name of the definition type that will contain (as text) the contents of the column and row intersection. You might call this the ‘container’ definition.

**Example (continued)**

In our example, we will fill in the following properties of the Matrix Dimensions tab of the Matrix Designer:



## Optional Settings

In the **Optional Settings** tab of the Matrix Designer, specify optional properties of the matrix, per the following explanations:

**MatrixXinCell** – A checked setting indicates that the matrix should only be displayed in a simple cross-reference mode, regardless of whether it is possible to display text in cells. In this mode, if the Column property and Row property are empty strings, then the CellDef property must be filled with valid values. A non-checked setting indicates that text should be shown in cells. In this mode, the CellDef must be given valid values.

**MatrixShowPickList** – A checked setting will cause the pick list of Column and Row definitions to be shown before the matrix is presented. The user also has the option at this point to save the matrix configuration.

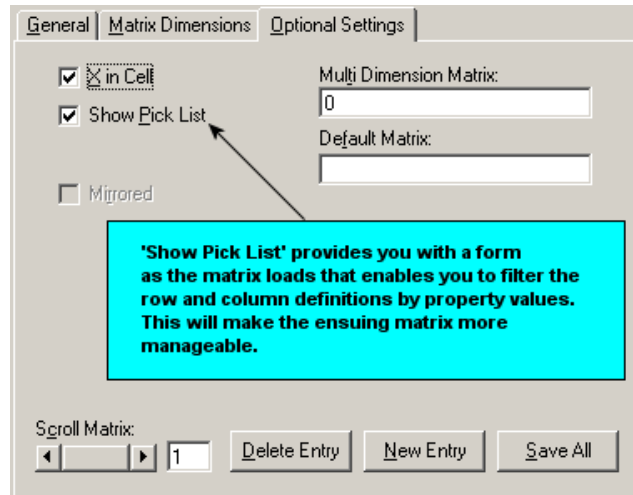
**MatrixMirrored** –A matrix that has the same row and column definition types can be displayed as mirrored matrices.

**MatrixMDM** – This represents the ID of the Multi-Dimensional Matrix group to which that particular matrix belongs.

**MatrixDefaultMatrix**- The name of the matrix, which should be selected from the list of located matrix configurations, by default. On loading User Matrices i.e. (Matrices of type SAMatrixSelectProc) a list of matrices saved by the user will be presented. In case the field ‘SAMatrixDefaultMatrix’ contains a matrix name, then that matrix is opened up by default instead of asking for the user to specify a preference through the list.

### Example (continued)

In our example, we fill in the following properties of the Matrix Dimensions tab of the Matrix Designer:



### Restart Rational System Architect for the Matrix to Be Loaded

Select **SaveAll** and close the Matrix Designer (click on the 'x' in the upper right hand corner of the Matrix Designer – if the Matrix Designer is maximized (not a floating window), you will find the 'x' in the upper right hand corner of Rational System Architect workspace).

Restart Rational System Architect. Select **View, Matrix Browser**. The matrix will be available in the **Matrix Browser** under the **User Defined tab**.



# 7

---

## *Requirements Tracking*

### **Introduction**

This chapter details how to use Rational System Architect as your design bed for capturing and tracking requirements, change requests, test plans, and the like during the design life cycle of your project. Rational System Architect also provides a direct interface to the IBM Rational DOORS requirements tool. Please see the on-line help of that interface for more information.

<b>Topics in this chapter</b>	<b>Page</b>
Handling Requirements in Rational System Architect	7-2
Built-in Requirements	7-3
Customizing Requirements	7-4
Adding Requirements to Definitions	7-6
Built-in Tracking Facility	7-7
How to Build Requirements — Discussion	7-12

---

## Handling Requirements in Rational System Architect

Rational System Architect enables you to incorporate on-going input from the outside world into the systems development process. This input can take the form of the initial statement of requirements, requests for changes as the system progresses, or verification of proper testing at each stage of development. In addition, you can use Rational System Architect to define and record business objectives, business processes, critical success factors, functional organization descriptions, and organizational goals.

### **Types of Requirements**

Rational System Architect provides a number of built-in requirement type definitions, and a number of ways for you to store and track requirement types against models. You may use these requirement types as is, or customize their definition properties through Rational System Architect's extensibility mechanism. You may also create new requirement definition types through this mechanism.

### **Handling Requirements**

There are two basic ways to handle requirement type definitions created, and correlate them with the analysis and design models. You may choose to use Rational System Architect's extensibility mechanism again to actually add requirement type definitions as properties of other model definition types. Alternatively, you may use a built-in tracking facility that enables requirement-type definitions to be assigned (or addressed) to symbols in models.

### **Reporting on Requirements**

You may run reports against requirement types that have been made properties of other definitions, or that have been addressed to symbols, using Rational System Architect's reporting facility and SA/Word link. These reports enable you to see how your system stacks up against your requirements, and to analyze the effects changing a requirement will have on your model.

---

## Built-in Requirements

Rational System Architect provides a number of built-in definitions to enable you to enter and track business objectives, goals, and requirements for your system. You may use these requirement-type definitions as is, you may customize the properties of each requirement-type definition to suit your needs for the information you are trying to capture, or you may create your own, new requirement types.

### Built-in Requirement Types

Rational System Architect's default set of built-in requirement types is as follows:

- Business Objective**
- Business Process**
- Change Request**
- Critical Success Factor**
- Current Data Collection**
- Data Class**
- Deliverable**
- Functional Organization**
- Geographic Location**
- Information Requirement**
- Organization Goal**
- Requirement**
- Test Plan**

Each definition type listed above contains properties. Rational System Architect installs them by default. You can add new properties, tailor default properties, or hide properties you do not need. This enables the definition to capture more precisely the information you are modeling.

**Note:** The Extensibility Manual and the on-line help provide extensive details on modifying the properties through USRPROPS.TXT.

### Addressables

Each of the built-in requirement types listed above is also considered an *addressable* definition – meaning a symbol on a diagram *addresses* a requirement. New requirement types that you add to the encyclopedia through USRPROPS.TXT can also be designated as addressables.



## Customizing Requirements

Customizing the properties of a definition type is performed using Rational System Architect's extensibility mechanism, USRPROPS.TXT.

The Extensibility Manual and the on-line help provide details on the full range of functionality provided by this mechanism. This section presents examples of some commands you can use to edit a requirement definition or add a new one.

### Modifying USRPROPS.TXT

In brief, every encyclopedia contains a USRPROPS.TXT file that overrides the default property file (SAPROPS.CFG) provided by Rational System Architect. To modify USRPROPS.TXT, perform the following steps:

1. Select **Tools, Customize User Properties, Export USRPROPS.TXT (Encyclopedia)**. The USRPROPS.TXT file is opened in Notepad.
2. In the **Export User Properties** dialog, select a target folder and click **Save**. The file will be opened automatically from that location.
3. Add information/make modifications to the USRPROPS.TXT file and Save the file.
4. Import USRPROPS.TXT from the **Tools menu, Customize User Properties, Import USRPROPS.TXT (Encyclopedia)** command.
5. Reopen the encyclopedia for the changes to take effect.

### Adding a New Property

You may add a new property to an existing definition type. For example, to add a new property to the **Requirement** definition which will hold a string of up to 20 characters, add the following code to USRPROPS.TXT:

```
DEFINITION "Requirement"
{
  PROPERTY "Approval Authority"
  { EDIT Text LENGTH 20 }
}
```

**Tailoring an Existing Property**

You may like the fact that the definition of Requirement contains a property "Source Entity." However, you may want to change it from a 20-character to a 30-character string and set "IEEE-1220-1994" as the default value. To do this, you must restate the old property, but give it new information, such as adding the following code to USRPROPS.TXT:

```
DEFINITION "Requirement"
{
  PROPERTY "Source Document"
  { EDIT Text LENGTH 30 DEFAULT
    "IEEE-1220-1994"}
}
```

**Hiding an Existing Property**

You cannot delete existing properties, but you can hide them. To hide the property "Impact Statement" from the definition of **Requirement**, add the following statements to USRPROPS.TXT:

```
DEFINITION "Requirement"
{
  PROPERTY "Impact Statement"
  { INVISIBLE}
}
```

**Adding a New Definition**

To add a new definition, you must actually rename one of 50 generic "User Defined" Definitions provided specifically for this purpose. So, for example, to create a new definition such as "Software Requirement", with the properties "Status" and "Programmer Responsible", add the following statements to USRPROPS.TXT:

```
RENAME DEFINITION "User 1" TO "Software Requirement"
DEFINITION "Software Requirement"
{
  PROPERTY "Status"
  { EDIT Text LENGTH 30 }
  PROPERTY "Programmer Responsible"
  { EDIT Text LENGTH 30 }
}
```

**Note:** These are only representative samples of what you can do to requirement-type definitions through USRPROPS.TXT. Please refer to the *Extensibility Guide* and the on-line help for full coverage.

---

## Adding Requirements to Definitions

The Extensibility mechanism (USRPROPS.TXT) enables you to add a requirement definition as a property of another definition. For example, you can make the definitions Requirement and Test Plan properties of a Use Case definition.

Attaching requirements as properties of definitions is an alternative to attaching requirement as addressables to symbols. Using this method attaches requirement definitions to the symbol's definition. This provides greater flexibility in manipulating information.

For example, to add the definition type "Requirement" as a property of a Use Case definition, add the following code to USRPROPS.TXT (see previous section for steps on how to do this):

```
Definition "Use Case"
{
  PROPERTY "Requirement"
  { ZOOMABLE EDIT ListOf "Requirement"
    LENGTH 1000 DISPLAY { FORMAT List } }
}
```

The EDIT ListOf statement above makes the property appear as a list box that you can drag existing definitions into, or add new ones to. The ZOOMABLE command makes the list box something you can expand or contract.

**Note:** The Extensibility Manual and the on-line help provide extensive details on modifying the properties through USRPROPS.TXT.

---

## Built-in Tracking Facility

### Addressables

A generic term for requirements, test plans, and other definitions you use to track your project is *addressables*. Rational System Architect provides a built-in tracking facility for addressables. Addressables may be attached to symbols within diagrams, and tracked throughout the project using Rational System Architect's reporting system. To enter Address-type definitions, click the **Dictionary** menu, then select the **Addresses** submenu.

With a symbol on a diagram selected, you may select **Dictionary, Addressables** to view the dropdown list of pre-defined **Addressable** definitions. Keep in mind, however, that you may create as many additional objects as you require using USRPROPS.TXT.

---

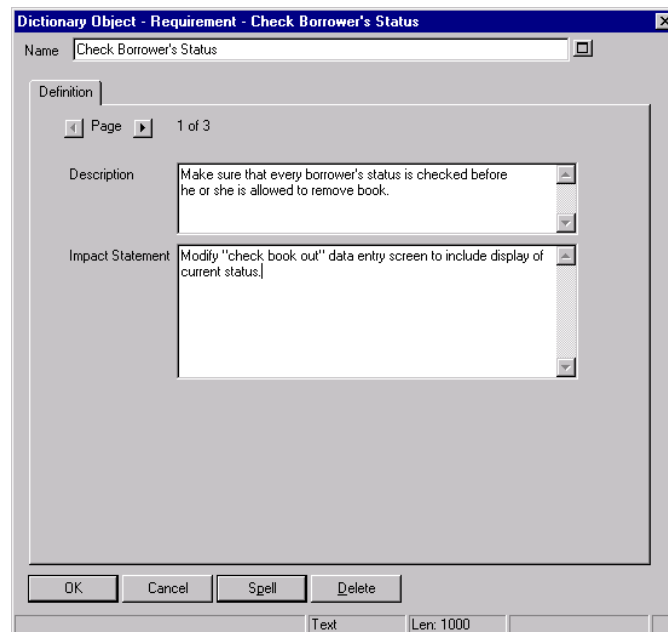
## Creating and Attaching a Requirement Specification

### Creating a Requirement Specification

Follow these steps to create a requirement specification. We'll use Requirement as the example:

1. Click the **Dictionary** menu, and select **New Definition**.
2. Enter the requirement's name in the **Dictionary Object** dialog (in our example: **Check Borrower's Status**).
3. Click **OK**.
4. Enter Description, Impact Statement, and pertinent documentation references in the definition dialog.

Figure 7-1. Dictionary Object dialog to add a definition



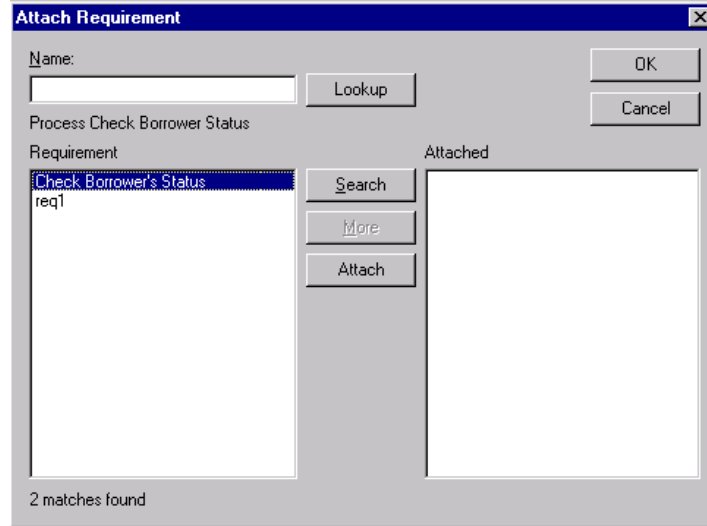
5. Click **OK** to store your definition and exit the **Definition** dialog.

**Attaching a  
Specification**

Follow these steps to attach a specification. We'll continue to use Requirements as our example:

1. Open a diagram and select a symbol that meets the Requirement, **Check Borrower Status**. In this example, we will use a process symbol on a Gane & Sarson Data Flow diagram.
2. From the **Dictionary** menu, select **Addresses**. Select the specific type; in the case of our example, **Requirement**. The **Attach Requirement** dialog is displayed.
3. To limit the number of requirements to choose from, you can type in the exact name and click **Search**, or type in the first one or two characters of the name, and click **Search**. All matches are displayed in the top list box. To scroll through the complete list of requirements, leave the Name field blank, and click **Search**.
4. Select a requirement from the list and click on the **Attach** button, or double-click on the requirement.
5. The name of the selected requirement is now moved from the left list box to the right list box.
6. Click **OK**.

Figure 7-2. Attach  
Requirement Dialog



### Looking Up a Specification

To see the definition of the requirement you can highlight its name in the upper scroll box and click the **Lookup** button. You are now looking at the **Dictionary Object <Type> <Name>** dialog containing the definition of the selected requirement.

### Adding a Specification During Attach Process

If you discover you need a new requirement, you can add it during the Attaching process.

1. Enter the name of the new requirement.
2. Click **Lookup** without having a requirement selected. The **Add Requirement** dialog opens.
3. Enter the requirement name and description (and any properties that you wish).
4. Click **Save** to store your new requirement and return to the **Attach Requirements** dialog.
5. Enter the first few letters of your new requirement and click **Search** (or simply click the **Search** button with no name entered to see the whole list). Your new requirement now appears in the upper box.

6. To attach it to the selected symbol on the diagram, highlight the requirement by clicking on it, and click the **Attach** button.

Your new requirement now appears in the lower box of attached requirements.

### **Detaching a Specification**

To detach a requirement:

1. Click in the lower box of attached requirements to highlight the requirement to detach. The **Attach** button becomes the **Detach** button.
2. Click the **Detach** button (or double-click your selection). The requirement is removed from the lower scroll box, and placed in the upper scroll box.



---

## How to Build Requirements

Incorporating this input must be done in such a way that it becomes a useful and verifiable part of the various models, and it must adhere to the methodological rule of centralized specification and accessibility. To meet these requirements the input must take the form of individual statements that have a name for reference purposes. A statement can be a description of what must be addressed (requirement, test plan, change request, etc.), or it can be a group of properties that describe the specification (requirement, test plan, change request, etc.).

Once the text document has been broken down into this format, the individual statements can be attached to various parts of a model (or multiple models) to be cross-referenced with the corresponding element of the model at any point in the project life style. A statement can be associated with multiple elements of the model that pertain to the statement or are needed to fulfill the statement. And any one element might have multiple statements attached to it that the element either completely or partially relates to or fulfills.

### Requirements Tracking

The first step in building (or rebuilding) a system is a statement or description of the functional requirements for the eventual automated system. This "description" is commonly a text document called the functional specification.

Requirements stated in this text document can be separated into individual, identified requirements and named. These requirements can either be imported into Rational System Architect , or entered directly one at a time through the dictionary interface.

Requirements can then be attached to parts of the analysis and design, allowing cross reference reports of both a requirement as it pertains to one or more elements of a model, and an element of a model as it pertains to one or more requirements.

The cross reference reports can answer important questions about the status of a project:

- Where are the requirements met?
- Are all requirements met?

- Does every part of the design meet a need? (For each element of the analysis, what requirements are met?)

Requirements may be broken down into a series of properties, just as a record in a database is defined by multiple fields. A requirement may have properties such as the reference to a source document and paragraph number, or it might list the "owner" of the requirement.

Requirements have their own type category in the data dictionary. Each individually named requirement derived from the source document can be imported to the Rational System Architect data dictionary directly using the Import facility (**Dictionary** menu, **Import Definitions**), or entered by hand (**Dictionary** menu, **Edit**).

### Test Plans Tracking

A project design needs a series of tests to verify all aspects of the system implementation. Tests can range from initial inspection through checking the detailed results of a transaction. As with the Requirements cross references, cross referencing the Test Plans with the diagrams can answer important questions:

- Where are the test plans met?
- Are all test plans met?
- Does every part of the design have a test?
- What Test Plans are provided for each part of the analysis?

Test plans, like requirements, can start as a text document produced outside of the modeling tool environment that then becomes incorporated with the models. The test plans from this text document can be stated as individual, identified and named tests. These tests can either be imported into Rational System Architect , or entered directly one at a time through the dictionary interface.

Test plans, like requirements can then be attached to parts of the analysis and design, allowing cross reference reports of both a test as it pertains to one or more elements of a model, and an element of a model as it pertains to one or more tests.

**Change Request Tracking**

When changes are made to the existing design of a system, change requests, like requirements and test plans, can be associated with the various aspects of the design which are affected by the change. Cross referencing the change requests with the diagram elements they are attached to can be a powerful management tool, and answer questions about the evolution of the system:

- How many changes has the system gone through?
- How many changes to the particular module or function?
- What were they?

# 8

---

## *Leveling Diagrams through Parent/Child Links*

### **Introduction**

Diagram leveling involves linking diagrams together in parent-child relationships. Typically, the child diagram is directly related to a symbol on a higher-level diagram; that symbol is referred to as the "parent." The parent symbol and the child diagram are functionally equivalent: everything on the child diagram is assumed to be in the parent symbol. Therefore, leveled diagrams provide the ability to express complexity in easy-to-understand chunks.

Rational System Architect provides the ability to link multiple child diagrams to a single parent symbol.

<b>Topics in this chapter</b>	<b>Page</b>
Linking Parent and Child Diagrams	8-2
Organization Techniques	8-8
Data Store Leveling	8-11

---

## Linking Parent and Child Diagrams

### Leveling Diagrams

The concept of leveling applies to many diagram types within many methodologies. UML notes the presence of invisible hyperlinks between Use Cases and Sequence diagrams, in which the Sequence diagram provides greater detail of each Use Case scenario. Business modeling encourages decomposition of functional models with process flow diagrams. Structured analysis and design techniques specify decomposition and leveling of data flow diagrams.

Leveling allows the user to divide the specification into a series of levels, with different diagrams at each level. We think of the printed diagrams forming a stack of paper, with the simplest diagrams at the top and the most complex at the bottom. With this technique, the parent level hides details in order to have its diagrams be more easily understood. Those details can be found at the child level, where more information is exposed.

### Multiple Children

Rational System Architect allows you to level any supported diagram type. Diagrams can be linked together so that a symbol on one diagram can be the parent of the other. In addition, you may introduce different methodologies at each level. For example, a process symbol can be linked to a flowchart showing the process logic.

Rational System Architect provides the ability to link multiple child diagrams to a single parent symbol. Any symbol type may be linked to child diagrams of any type. For example, a Use Case may be linked to a child Sequence diagram, as well as an Activity diagram, as well as another Use Case diagram.

A diagram may also serve as a child to multiple parent symbols. So for example, you could have an Activity diagram that has as a parent a Use Case symbol on a Use Case diagram, and an Elementary Business Process symbol on a Process Chart, etc.

### Attaching a Child Diagram

To attach an existing diagram as a child diagram to a symbol:

1. Open the diagram containing the symbol you wish to attach a diagram to.

2. Right-mouse click on the symbol and select **Child Attach** from the drop-down list, or select the symbol and choose **Child Diagram, Child Attach** from the **Edit** menu.
3. Click **OK** to the dialog that asks if you wish to save the current diagram. You must save the current diagram at this point to attach the child diagram.

The diagram will be attached as a child and opened.

### Creating Multiple Children

There is no preset limit to the number of child diagrams that you may create for a given symbol. To create multiple child diagrams:

1. Open the diagram containing the symbol you wish to attach the child diagrams to.
2. **Create** or **attach** a child diagram to the selected symbol.
3. Move back to the parent diagram and choose the parent symbol again.
4. Right-mouse click on the symbol and select **Child Create** or **Child Attach** to create or attach a second child diagram. Select **OK** to the message that asks if the current diagram should be saved.
5. The second child diagram will be created and opened. To verify that there are now two children to the parent, perform the following steps:
6. Move back to the parent diagram and choose the parent symbol again.
7. Right-mouse click on the symbol and select **Child Open**, or select the symbol and select **Edit, Go to Diagram, Child**.
8. A dialog will open which shows the child diagrams attached to the parent symbol. Select a child diagram and click the **Open** button to open it.

### Creating Multiple Parents

To create multiple parents, perform the following steps:

1. Right-mouse click on a symbol on an open diagram and choose **Child Attach** (or **Child Create**). Select a diagram to attach to this symbol as a child and click **OK**. Select **OK** to all dialog prompts.

2. Open a third diagram, right-mouse click on a symbol on the diagram, and choose **Child Attach** (or **Child Create**). Select the same diagram to attach to this symbol as that selected above. Select **OK** to all dialog prompts. The diagram now has two parents.
3. With the child diagram open, right-mouse-click on the diagram workspace and choose **Parent**. A dialog will open that enables you to select which parent diagram to navigate to. (Note: You may select **Top** to go to the highest parent diagram in a chain.)

### **Synchronization Between Parent and Children**

In Rational System Architect, the synchronization between parent symbol and child diagram is not restrictive. In general, work you do on the parent diagram has no effect on the child, and vice versa. Here are some examples of the non-restrictive linkage:

- Add a new data flow going into or out of the parent symbol, or onto the child diagram. That same data flow does **not** automatically appear on the other diagram.
- Rename a data flow on the child diagram. That same data flow is **not** automatically renamed on the diagram of the parent symbol, and vice versa. There are ramifications to consider when renaming a data flow on a diagram or on a parent symbol. Please refer to the sections in the on-line help called *Tools Menu*, *Global Change Command*, and *Symbol Name* for detailed information.
- Change the name of the parent symbol. The name of the child diagram is **not** automatically changed to match.

Here are some examples of tight linkage:

- Change the number of the parent symbol. The decimal numbers of symbols on the child diagram **automatically** change to match.
- Delete the child diagram. The three-dot expand indicator above the parent symbol **automatically** disappears.

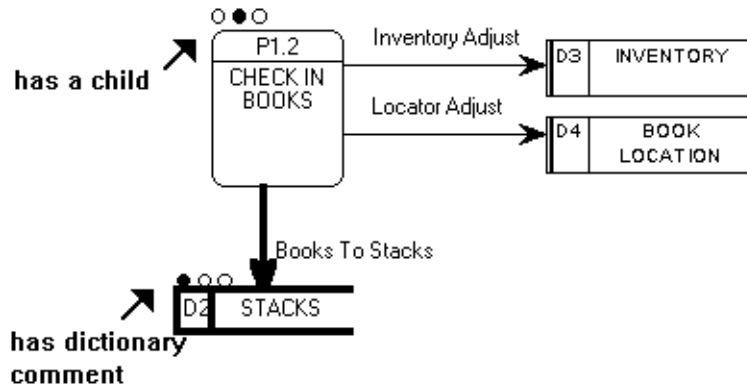
**Expand Indicators**

Expand indicators are the three dots that sometimes appear above a symbol. One or more of the dots are blackened with the following meaning:

- **left:** the symbol has a dictionary comment
- **center:** the symbol has a child diagram
- **right:** (reserved for future use)

When a symbol expands to a child diagram, a three-dot expand indicator appears above it, with its center dot blackened.

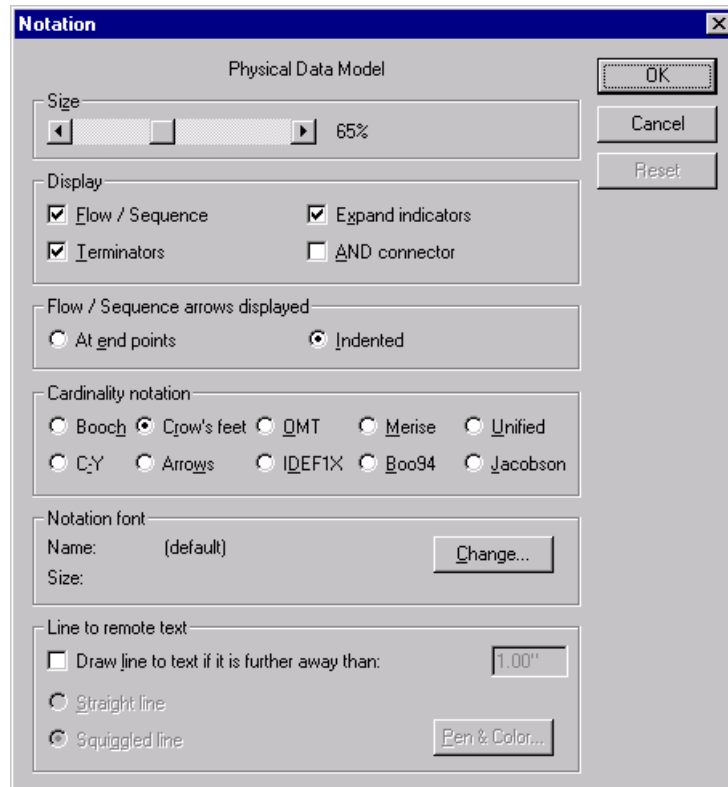
Figure 8-1. Example of Expand Indicators



A check-box on the **Notation** dialog under the **Format** menu, **Diagram Format** command, controls whether expand indicators are to be shown or hidden.



Figure 8-2. Notation Dialog - Display Expand Indicators



### Accessing Multi-Level Diagrams

You can use the **Child Diagram**, **Child Attach** and **Child Detach** commands, located in the **Edit** menu, to change the linkage between two diagrams. You can link together different diagram types.

The **Child Diagram**, **Child Create** command, also located in the **Edit** menu, opens a window asking you to name a new child diagram and select the diagram type. The new diagram is created and attached to its parent process.

The **Child Diagram** icon accesses an existing child diagram below a selected symbol.

**Note:** This icon is not active unless the diagram is related to a parent symbol.

The **Parent Diagram** icon accesses the diagram above the currently displayed diagram. The **Top Diagram** icon goes to the highest diagram in the hierarchical structure.

**Note:** This icon is not active unless the selected symbol is related to a child diagram.

---

## Data Flow Organization Techniques

### The Concept of the Child Diagram in Structured Analysis

A workable model that can be clearly understood is usually *leveled* or broken down into a series of linked parent-child diagrams. This chapter deals with the basic ideas and techniques involved in organizing model diagrams into multi-level units that retain referential integrity.

A single diagram rarely contains an entire data flow or behavioral model. For this reason, the process of creating child diagrams and the necessity of retaining referential integrity between the different levels of diagrams you create, becomes an important consideration. When creating a child data flow diagram, it is important to follow certain design rules. The two most important ones are listed below:

1. Never rename a data flow copied from a parent. If you want a different name, delete the data flow on the child and draw a brand new one. The problem here is to preserve the dictionary definition. If you rename the child flow, it takes the dictionary definition with it, leaving the parent flow undefined.
2. There is frequently a need to "break out" a data flow on a child diagram. For example, where the parent diagram has flow "A," the child needs to have subset flows, A1 and A2. First, as noted immediately above, start by deleting the unneeded "A" flow that was brought down from the parent. Then draw A1 and A2.

Now the question arises of how to ensure that parent and child balance properly. Here is Ed Yourdon's<sup>1</sup> recommendation: Return to the parent's "A" flow and give it this expression as its definition:

---

<sup>1</sup> Yourdon, E., *Modern Structured Analysis*, Prentice-Hall, 1989, Prentice-Hall, Englewood Cliffs, NJ

3. [ *A1* | *A2* ]

*A1* and *A2* should be defined as data structures. Then go to the child diagram and define dataflow *A1* as being made up of just data structure *A1*, and dataflow *A2* as being made up of just data structure *A2*. Finally, *A1* and *A2* can be populated with lists of data elements.

In their textbook, Gane & Sarson state that three levels should be enough to handle most situations. On the other hand, Yourdon and DeMarco tell you to keep creating child levels until it doesn't make sense to go any lower. (They give guidelines to help you recognize when you've reached a good stopping point.)

### Setup to Ensure Decimal Point Numbering

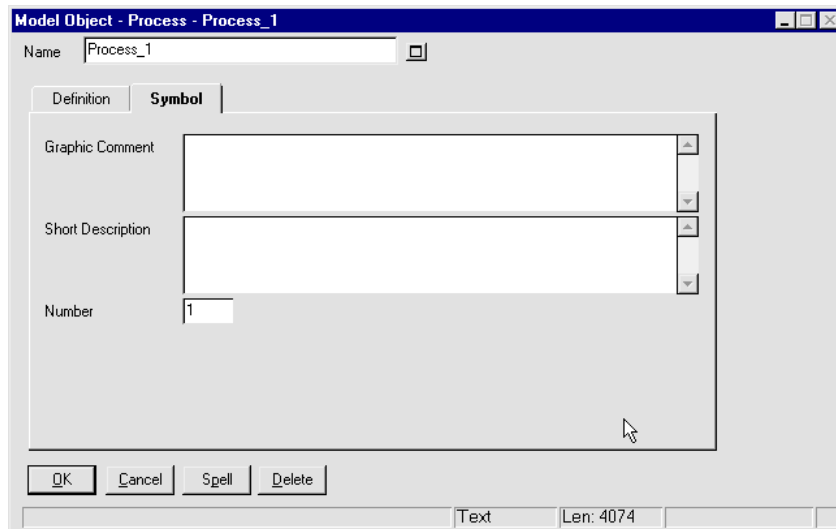
The authors of data flow diagram methodologies recommend that you number your child process symbols with decimal points, such as P1 . 2 . 3. Rational System Architect lets you enable this method if your project requires it. You can also enable it for symbols other than processes. To put decimal point numbers in your child symbols, follow these steps to set the required options:

1. Click the **Tools** menu and select **Preferences**. Ensure that in the Auto list, **Number** is toggled on.
2. Select the symbol you want to expand to a new level by clicking on it once.
3. Click the **Format** menu, select **Symbol Format** and **Text Display**. Ensure that there is a check in **Name Level Numbers**.
4. With the symbol still selected, click the **Format** menu, select **Symbol Format** and **Symbol Style**. Ensure that there is a check in **Allow Auto Number** and in **Allow Level Numbers**.
5. Click **OK**.

Also, before creating the child diagram, ensure that the symbol you are about to expand currently has a number of its own, such as *P1*. If not, right-click on the symbol, select **Edit**, then click the **Symbol** tab. If the value of the *Number* property is 0, you may enter a number manually. If you have set the dialogs correctly as explained

above, however, all symbols that are supposed to be numbered are numbered automatically as they are added to diagrams.

Figure 8-3. Symbol Properties Tab



### Balancing Leveled Data- flow Diagrams

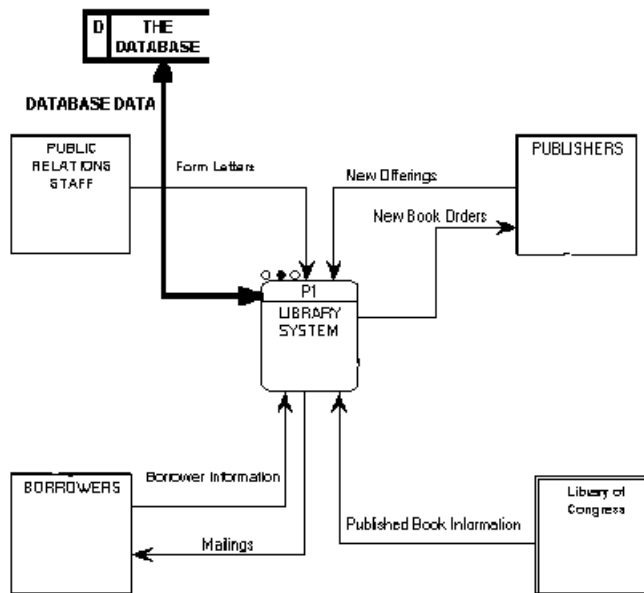
It is considered good practice to ensure that each child data-flow diagram can balance against its parent process. This means that any data entering the parent process should also be seen to enter the child diagram. Similarly, any data leaving the parent should also be seen leaving the child diagram. Refer to the on-line help for details. Search on *Balance Child(ren)* and *Balance Parent* commands in the on-line help.

## Data Store Leveling

*Rational System Architect provides support for a Gane & Sarson Data Flow Diagram that includes the Data Store symbol. If you are modeling the processing of your data warehouse, the following discussion of Data Store Leveling may be interesting and useful.*

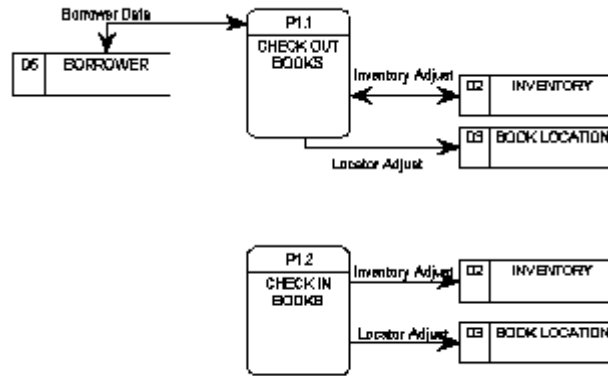
The two diagrams that follow are a simple example of two levels: the diagram in Figure 8-4 contains a data flow, *DATABASE DATA*, that carries data from and to the parent symbol, *LIBRARY SYSTEM*, and a data store, *THE DATABASE*. Both the data store and the data flow must be leveled on the child diagram, pictured in Figure 8-5.<sup>2</sup>

Figure 8-4. A Context Diagram Showing "The Database"



<sup>2</sup> Both diagrams are incomplete. Only the leveling required by the data store and the data flow are included.

Figure 8-5. A Portion of the Child Diagram with Three Data Stores



Clearly, *THE DATABASE* in Figure 8-4 must be a composite of the four data stores seen in Figure 8-5<sup>3</sup>. Similarly, *DATABASE DATA* in Figure 8-4 must be a composite of all the data flows into and out of the data stores on Figure 8-5.

---

<sup>3</sup> Some data stores appear twice.

Here is how to define the data stores and the data flows in order to ensure proper parent/child balancing:

Type	Name	Definition
Data Store	"THE DATABASE"	BORROWER + BOOK LOCATION + INVENTORY + STACKS
	BORROWER	BORROWER
	BOOK LOCATION	BOOK LOCATION
	INVENTORY	INVENTORY
Data Flow	"DATABASE DATA"	"Borrower Data" + "Books from Stacks" + "Inventory Adjust" + "Locator Adjust" + "Books to Stacks"
	"Borrower Data"	"Borrower Data"
<b>Type</b>	<b>Name</b>	<b>Definition</b>
	"Inventory Adjust"	"Inventory Adjust"
(Similarly for the other data flows into and out of the data stores).		

The key to this leveling technique is to define everything in terms of data structures. "Real" data stores and flows at the lower level are each defined with a single structure, while the composite store and flow in the context diagram are defined as composites of those same structures. This technique ensures proper balancing.

The final step is to define all of the data structures with their appropriate data elements or lower-level structures.

**Tip:** Run the reports "COMPARE DATA ELEMENTS ENTITIES AND PROCESSES" and "COMPARE DATA ELEMENTS IN PROCESSES AND ENTITIES" to make sure all the data elements stored in the logical data model are processed, and vice versa.





# 9

---

## ***Reporting and Documenting System***

### **Introduction**

This chapter introduces Rational System Architect's reporting and documenting facilities. Detailed information is located in the on-line help.

<b>Topics in this chapter</b>	<b>Page</b>
Internal Reporting System	9-2
Microsoft Word Reports	9-5
HTML Generator	9-6

---

## Internal Reporting System

Rational System Architect's internal reporting system provides over 130 prewritten reports that are contained within a number of .rpt files. These reports are accessed through the **Reports** menu choice within the tool, and can be easily selected and run without modification. The user can also modify the existing reports, or add their own through a built-in graphical user interface. The reports are written in a SQL-based language, used to traverse the information stored in an encyclopedia. For this reason, to write a complicated report, knowledge of the Rational System Architect encyclopedia metamodel and underlying semantic relationships is generally necessary.

### Prewritten Report Files

Rational System Architect comes complete with a set of standard reports that you can run against the information in your project encyclopedia to create printed output about the information. You can also copy a standard report and edit it to more closely meet your needs. Or, if you are a more experienced user, you can create your own Rational System Architect report from scratch.

There are a number of prewritten report files that each contain a litany of individual reports. These report files are located in a **Reports** subdirectory under the main **Rational System Architect** program folder. The following report files are provided:

**Address.rpt** – reports that track addressable definitions (ie, requirements)

**Entprise.rpt** – reports on Business Enterprise modeling diagrams/definitions

**Ermatrix.rpt** – matrix reports on ER diagrams

**Erreport.rpt** – reports on ER diagrams

**IDEF.rpt** – reports on diagrams/definitions of the IDEF method

**UML.rpt** – reports on UML diagrams/definitions

**Project.rpt** – general, project-level reports

**Reports.rpt** – general reports on diagrams/definitions

**Rules.rpt** – rules reports on structured diagrams

**XMLschema.rpt** – reports on entities, elements and attributes

## Running a Report

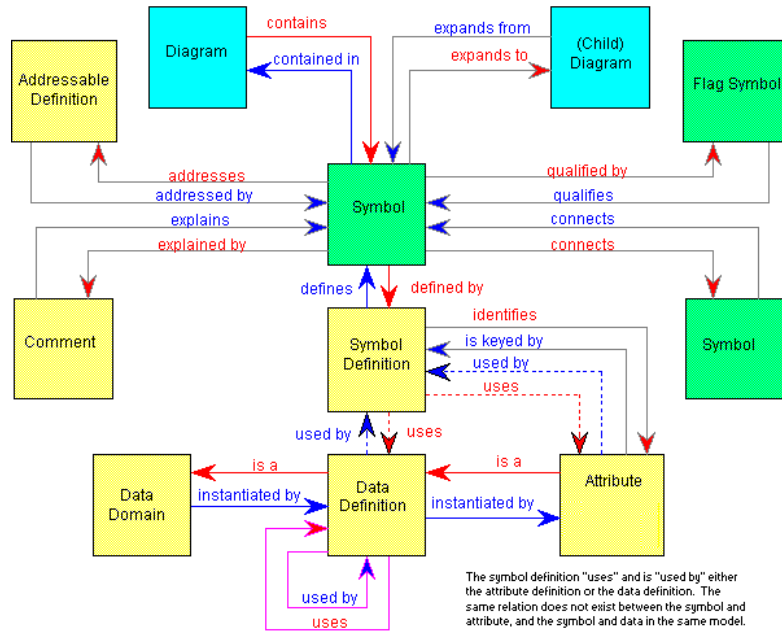
To run a report, perform the following steps:

1. Select **Reports, Report Generator**. The **Reports** dialog opens.
2. Select the report file that contains the report you wish to run. The default report file is **Reports.rpt**. To select a different report file, click **File** menu and select **Open Report File**.
3. Select a report file from the **Report File Selection** dialog and click **Open**.
4. In the **Reports** dialog, select the individual report you wish to run.
5. Click on **Print**, or **Draft**.

## Creating Your Own Report

The Reporting system offers users a choice of a graphical interface and a text editor to create their own reports. However, users should become familiar with the Rational System Architect metamodel before attempting to write their own report. It is imperative that when creating a report, the user 'tunnel' through the correct relationships between elements in the Rational System Architect metamodel. Please see the Rational System Architect on-line help and on-line tutorial for more information.

Figure 9-1. The Rational System Architect Metamodel.



The procedure for creating a new report is as follows:

1. Select **Reports, Report Generator** to open the **Reports** dialog.
2. Optionally create a new report file by selecting **File, New Report File**. You may create a new report within an existing report file as well.
3. Select **Add** in the **Reports** dialog. The **Add Report** GUI will open.
4. Give the report a name, and begin tunneling down through diagrams, symbols, definitions, and relationships using the GUI interface, and referencing the Rational System Architect metamodel.

**Note:** Extensive information on the Internal Reporting system, including an on-line tutorial, is provided in the on-line help.

---

## Microsoft Word Reports

The SA/Word reporting feature permits you to use Microsoft Word templates, written in Visual Basic for Applications (VBA) programming language, to query an open encyclopedia and produce formatted reports in Word. A number of prewritten templates (.DOT files) are provided with the product, and can be found in Rational System Architect's installed subdirectory under the Templates folder. You may use these templates as is, customize them or create new ones using VBA.

### Word Templates Provided

The following Word templates are provided with Rational System Architect. You may run a Word Report from the product by selecting **Reports, Word Reports** and selecting a report from the drop-down menu.

- Saaudit9.dot** – reports on the SA Audit ID trail.
- Sacat.dot** – reports on the Catalyst methodology.
- SADiags9.dot** – general reports on diagrams in the project.
- SAIDEF.dot** – reports on the IDEF methodology.
- SALogMod9.dot** – reports on logical ER data models.
- SAObjMod9.dot** – reports on UML object/component models.
- SAPhyMod9.dot** – reports on physical data models.
- SAStruct9.dot** – reports on structured analysis and design models.

Extensive information on the SA/Word reporting system, including an on-line tutorial, is provided in the on-line help.
--

---

## HTML Generator

Rational System Architect provides an HTML generator, SA/HTML, that publishes diagrams and definitions of a Rational System Architect encyclopedia to HTML format. This enables the contents of an encyclopedia to be published on an internet website or corporate intranet.

SA/HTML has been designed to provide an easy to use interface, while offering flexibility in the delivery of HTML files. A template-based system enables a great deal of control over the display of information, together with the ability to customize the look and feel to suite your needs.

### Running SA/HTML

To run the HTML Generator, select **Reports, HTML Reports** from the main menu in Rational System Architect. You will be presented with the **System Architect - HTML** dialog, which enables multiple tabs of selections on HTML generation.

You must select a path and file name to publish the report in the **Publish Home Page** option in the **General** tab, and select one or more diagrams to publish in the **Diagrams** tab for the **Publish** button to be enabled.

Extensive information on the SA/HTML generator, including an on-line tutorial, is provided in the on-line help.
---

# 10

---

## *IBM support*

### **Introduction**

There are a number of self-help information resources and tools to help you troubleshoot problems. If there is a problem with your product, you can:

Refer to the release information for your product for known issues, workarounds, and troubleshooting information.

Check if a download or fix is available to resolve your problem.

Search the available knowledge bases to see if the resolution to your problem is already documented.

If you still need help, contact IBM® Software Support and report your problem.

<b>Topics in this chapter</b>	<b>Page</b>
Contacting IBM Rational Software Support	10-2



---

## Contacting IBM Rational Software Support

If you cannot resolve a problem with the self-help resources, contact IBM® Rational® Software Support.

**Note:** If you are a heritage Telelogic customer, you can find a single reference site for all support resources at

<http://www.ibm.com/software/rational/support/telelogic/>

### Prerequisites

To submit a 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 at <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](http://www.ibm.com/software/lotus/passportadvantage/brochures_faqs_quickguides.html).
- For further assistance, contact your IBM representative.

To submit a problem online (from the IBM Web site) to IBM Rational Software Support:

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

Other information

For Rational software product news, events, and other information, visit the IBM Rational Software Web site:

<http://www.ibm.com/software/rational/>.

**Submitting problems**

To submit a problem to IBM Rational Software Support:

1. Determine the business impact of the 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.

To determine the severity level, use the following table.

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	The 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 the problem and gather background information. When you describe the 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?
3. Are you currently using a workaround for the problem? If so, be prepared to describe the workaround when you report the problem.
4. Submit the problem to IBM Rational Software Support in one of the following ways:
- Online: Go to the IBM Rational Software Support Web site at <https://www.ibm.com/software/rational/support/>. In the Rational support task navigator, click **Open Service Request**. Select the electronic problem

reporting tool, and open a Problem Management Record (PMR) to describe the problem.

- 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 by using the IBM Support Assistant. For more information, go to <http://www.ibm.com/software/support/isa/faq.html>.
- By phone: For the phone number to call in your country or region, visit 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. For complete contact information for each country, visit <http://www.ibm.com/planetwide/>.

# 11

---

## *Appendix:*

### **Introduction**

This chapter contains information about the legal uses and trademarks of IBM® Rational® System Architect®.

<b>Topics in this chapter</b>	<b>Page</b>
Notices	11-2
Trademarks	11-5
Copyright acknowledgements	11-6

---

## Notices

© Copyright IBM Corporation 1986, 2009.

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

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

IBM World Trade Asia Corporation  
Licensing  
2-31 Roppongi 3-chome, Minato-ku  
Tokyo 106-0032, 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, MA 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.

### **Copyright license**

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. 2000 2009.

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



---

## 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 “[Copyright and trademark information](#)” at [www.ibm.com/legal/copytrade.html](http://www.ibm.com/legal/copytrade.html)

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

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

---

## Copyright acknowledgements

International Proofreader™ English (US and UK) text proofing system © 2003 by Vantage Technology Holdings, Inc. All rights reserved. Reproduction or disassembly of embodied algorithms or database prohibited.

International Proofreader™ French text proofing system © 2003 by Vantage Technology Holdings, Inc. All rights reserved. Reproduction or disassembly of embodied algorithms or database prohibited.

International Proofreader™ German text proofing system © 2003 by Vantage Technology Holdings, Inc. All rights reserved. Reproduction or disassembly of embodied algorithms or database prohibited.

International Proofreader™ Afrikaans text proofing system © 2003 by Vantage Technology Holdings, Inc. All rights reserved. Reproduction or disassembly of embodied algorithms or database prohibited.

International Proofreader™ Catalan text proofing system © 2003 by Vantage Technology Holdings, Inc. All rights reserved. Reproduction or disassembly of embodied algorithms or database prohibited.

International Proofreader™ Czech text proofing system © 2003 by Vantage Technology Holdings, Inc. All rights reserved. Reproduction or disassembly of embodied algorithms or database prohibited.

International Proofreader™ Danish text proofing system © 2003 by Vantage Technology Holdings, Inc. All rights reserved. Reproduction or disassembly of embodied algorithms or database prohibited.

International Proofreader™ Dutch text proofing system © 2003 by Vantage Technology Holdings, Inc. All rights reserved. Reproduction or disassembly of embodied algorithms or database prohibited.

International Proofreader <sup>TM</sup> Finnish text proofing system © 2003 by Vantage Technology Holdings, Inc. All rights reserved.  
Reproduction or disassembly of embodied algorithms or database prohibited.

International Proofreader <sup>TM</sup> Greek text proofing system © 2003 by Vantage Technology Holdings, Inc. All rights reserved.  
Reproduction or disassembly of embodied algorithms or database prohibited.

International Proofreader <sup>TM</sup> Italian text proofing system © 2003 by Vantage Technology Holdings, Inc. All rights reserved.  
Reproduction or disassembly of embodied algorithms or database prohibited.

International Proofreader <sup>TM</sup> Norwegian, text proofing system © 2003 by Vantage Technology Holdings, Inc. All rights reserved.  
Reproduction or disassembly of embodied algorithms or database prohibited.

International Proofreader <sup>TM</sup> Portuguese, text proofing system © 2003 by Vantage Technology Holdings, Inc. All rights reserved.  
Reproduction or disassembly of embodied algorithms or database prohibited.

International Proofreader <sup>TM</sup> Russian, text proofing system © 2003 by Vantage Technology Holdings, Inc. All rights reserved.  
Reproduction or disassembly of embodied algorithms or database prohibited.

International Proofreader <sup>TM</sup> Spanish, text proofing system © 2003 by Vantage Technology Holdings, Inc. All rights reserved.  
Reproduction or disassembly of embodied algorithms or database prohibited.

International Proofreader <sup>TM</sup> Swedish, text proofing system © 2003 by Vantage Technology Holdings, Inc. All rights reserved.  
Reproduction or disassembly of embodied algorithms or database prohibited.

---

# Index

•

*.DGX file*, 4-32

*.FON file*, 4-21

*.FOT file*, 4-21

*.TTF file*, 4-21

## A

Actual size view

    Keyboard accelerator, 4-34

Add

    Addresses (Specifications), 7-8, 7-10

**Add Definitions**

**To encyclopedia**, 2-21

Adding new definition

    Immediate save, 2-21

Adding new dictionary entry

    Immediate save, 2-21

Addresses

    Attach specifications to symbol, 7-9, 7-10

    Define specifications, 7-8, 7-10

    Detach specifications from symbol, 7-11

AND connector

    Made invisible, 4-15

    Symbol

        Compared to XOR, 4-10

Arial font

    Default font, 4-30

    SA.INI default, 4-22

    Scaleable, 4-21

## Index

### Arrowheads

- Crow's feet, 4-8
- To select line, 4-17

### Attach

- Addresses (Specifications), 7-9, 7-10
- Attach button, 7-11

### Attribute

- Defined, 5-9
- Primary Key, 5-9

### AUTOEXEC.STY file

- Create master style sheet, 4-24
- Instructions, 4-31

## **B**

### Bend points on lines

- To select line, 4-17

### Block Move

- Keyboard accelerator, 4-35

## **C**

### C typedef

- SA Schema Generator, 1-8

### C++ header file

- SA Schema Generator, 1-8

### CASE

- Environment, 7-13

### Change Request, 7-12, 7-14

### Child Diagram, 8-2

### Clicking speed, 4-3

### Clipboard

- Copying diagram to, 4-32
- Copying symbols to
  - Instructions, 5-23
  - Keyboard accelerator, 4-33
- Copying text to, 5-23
- Cutting symbols to
  - Keyboard interface, 4-33
- Pasting symbols from**
  - Instructions**, 4-32
  - Keyboard interface, 4-33

- COBOL data definition
  - SA Schema Generator, 1-8
- Comment
  - graphic
    - symbol property, 5-17
    - symbol property, 5-17
- component-based modeling, 1-2
- Configuration Settings, 3-3
- Consistency checks, 4-12
- Copying
  - Symbol
    - Keyboard accelerator, 4-34
- Copying a diagram, 4-32
- Cross-Reference, 3-2, 7-12, 7-14
- Crow's feet
  - Notation dialog, 4-15
  - Source and target of line, 4-8
- Curved line
  - Drawing, 4-8
  - for Yourdon/DeMarco & Ward & Mellor DFD's, 4-6

## **D**

- Data domain
  - Definition, 5-12
- Data Domain
  - Defined, 5-12
  - Definition, 5-8
- Data Element
  - Defined, 5-10
- Data store
  - Leveling of, 8-11
- Data structure
  - Defined, 5-10
  - Definition, 5-8
- dBASE III PLUS, 5-7
- DDL statement
  - SA Schema Generator, 1-8
- Define
  - Addresses (Specifications), 7-8, 7-10

## Index

### **Defining symbol**

**Instructions**, 2-21

#### Definition

related to symbol, 5-17

#### Deleting

Dictionary entry

Parent symbol and child diagram data flow, 8-8

Symbol

Clipboard

Keyboard interface, 4-33

Keyboard accelerator

Edit Menu, Cut command, 4-34

Parent symbol and child diagram data flow, 8-8

DeMarco, Tom, 8-9

Description, 7-12

#### Detach

Addresses (Specifications), 7-11

Detach button, 7-11

#### Diagram

Generating, 4-1

Leveling, 8-11

Saving when exiting Rational System Architect, 2-27

#### Dictionary

Comments

Expand indicators, 4-15, 8-5

Interface, 7-13

Dictionary Comment, 5-17

Definition, 5-17

#### Display font

Instructions, 4-21

## **E**

#### Encyclopedia

definition, 9-2

Path, 4-31

#### Entity

Primary Key, 5-9

Subordinate definitions

Attribute, 5-9

Data Domain, 5-12

Data Element, 5-10

- Entity Relation diagram
  - Relationship line, 4-8
- Exiting from SA/Data Architect
  - Keyboard accelerator, 4-34
- Exiting from Rational System Architect, 2-27
- Expand indicators
  - Defined, 5-18
  - Dictionary
    - Comments:, 8-5
  - Notation Dialog, 4-15

## **F**

- Font
  - Instructions, 4-21
  - Reduce size
    - Truncation indicator, 4-19
  - Size
    - Reduction through SA.INI entry, 4-23
  - Style, 4-15
    - SA.INI default, 4-22
  - Style Sheet**, 4-29
- FontModern
  - SA.INI entry, 4-23
- Full page view
  - Keyboard accelerator, 4-35
- Functional Requirements, 7-12

## **G**

- Gane & Sarson, 1-2
- Gane, Chris & Trish Sarson, 8-9
- Generating Diagrams, 3-1, 4-1, 7-1
- Glossary, 4-32
- Graphic comment, 5-17
  - Definition, 5-17

## **H**

- Help
  - Keyboard accelerator, 4-34



## Index

### Helvetica font

- Default font, 4-30
- Not scaleable, 4-21
- SA.INI default, 4-22

## I

- IDEF methodology, 1-2
- Import and Export, 5-20
- Import Facility, 7-13
- Imposing style settings on symbols, 4-29, 4-31
- Interface data flows, 4-7

## K

### Keyboard Shortcuts (Accelerators)

- Alt F4, 2-27
- F10, 4-17
- Instructions**, 4-34

## L

- Level numbers with decimal points, 8-9
- Leveling
  - of data stores, 8-11
  - of diagrams
    - Creating a child diagram, 8-8
    - Data store role, 8-11
    - Instructions**, 8-2
    - Loose linkage, 8-2

## M

### Matrices

- without text, 6-5, 6-19

### **Matrix, Text Cross-Reference**, 6-3

### **Matrix“Multi-dimensional” Cross-Reference**, 6-3

### **Matrix“X in Cell” Cross-Reference**, 6-3

### Minispecs

- Copying to clipboard, 5-23

Modifying dictionary entry, 2-21

Moving symbols

Instructions, 4-18

Multi-dimensional Matrix, 6-12

Multiple symbol selection

Block move, 4-18

Techniques, 4-18

## **N**

Name style

Grid & Reduced View Settings dialog, 4-16

Level numbers, 8-9

Text Display dialog, 4-15

Naming symbols

Parent symbol and child diagram data flow, 8-8

New diagram

Starting a, 4-32

Notation style, 4-15

Numbering symbols

Relationship between parent symbol and child diagram, 8-4

## **O**

Overlapping symbols, 4-17

## **P**

Page-Jones, Meilir, 4-32

Pasting

Symbol

Keyboard accelerator, 4-34

Physical data model

SA Reverse Data Engineer, 1-9

Preference setting

Auto Number, 8-9

Display Truncation indicator, 4-19

Primary Key

Defined, 5-9

## Index

**Printer font**, 4-21

Printing Diagram

Keyboard accelerator, 4-34

Properties

Maximum characters allowed, 5-5

## R

Redraw Command

Forcing redraw of diagram

Keyboard accelerator, 4-34

Reduced diagram view

By percentage

Keyboard accelerator, 4-35

One full page, 4-22

Keyboard accelerator, 4-35

Relationship

Consistency check, 4-12

Relationships within the encyclopedia

Relationship file, 9-6

Renaming

Symbols, 8-4

Parent symbol and child diagram data flow, 8-8

Reporting System

Overview, 1-6

Reports

Printed

Copying to clipboard, 5-23

Requirement, 7-12

Requirements Traceability, 7-13

Resizing symbols

Manually

Using handles, 4-18

**Right mouse button**

**Elliptical line**, 4-9

## S

SA Reverse Data Engineer

Physical data model diagram, 1-9

- SA Schema Generator, 1-8
- SA.INI
  - Default font, 4-30
  - Font= setting, 4-22
  - Stylesheet= setting, 4-24
- SA/Data Architect
  - Import Facility, 7-13
  - Tracking Facility, 7-2
- Saving diagram
  - On exiting Rational System Architect, 2-27
  - With new symbols, 4-28
- Scaleable font, 4-23
  - Arial, 4-21
- Screen capture, 5-26
- Screen Painter** option
  - Overview, 1-7
- Select Mode**, 4-3
- Selecting
  - All symbols
    - From one diagram to another, 4-32
    - Keyboard accelerator, 4-34
  - Multiple symbols
    - Block move, 4-18
    - Techniques, 4-18
  - Symbol, 4-6
    - Keyboard accelerator, 4-34, 4-35
- Session
  - Save Style sheet, 4-28
- Shortcut keys, 4-34
- Simultaneous Select/Draw**, 4-3
- Specifications
  - Attach Addresses to symbol, 7-9, 7-10
  - Define Addresses, 7-8, 7-10
  - Detach Addresses from symbol, 7-11
- SSADM methodologies, 1-2
- Structure charts
  - Line style, 4-6
  - Transform from other diagram type, 4-36
- Style sheet
  - General discussion, 4-24
  - Opening, 4-31

## Index

Saving when exiting Rational System Architect, 2-27

### **Style Sheets**, 4-22

#### Symbol

Attach Addresses (Specifications), 7-9, 7-10  
Define Addresses (Specifications), 7-8, 7-10  
Detach Addresses (Specifications), 7-11  
relation to its definition, 5-17

### **Symbol definition**

**Overview**, 2-21

#### Symbol style

Allow Auto Number, 8-9  
Allow **Level** Numbers, 8-9  
Set as template, 4-28

## **T**

Test Plan, 7-12

Test Plans, 7-13

Tests performed on relation lines, 4-12

Tracking Facility, 3-2, 3-3, 7-3, 7-4, 7-6, 7-7, 7-12

#### Traffic light

symbol adornment, 5-18

Transform analysis, 4-32

#### Truncated text

Instructions, 4-19

#### Truncation indicator

Instructions, 4-19

## **U**

UML notation, 1-2

Undo command, 4-20

#### USRPROPS.TXT

Maximum characters for values, 5-5

## **W**

Ward & Mellor, 1-2

#### Ward & Mellor dataflow diagram

Elliptical arc line

Drawing, 4-8

Ward, Paul & Steven Mellor, 4-10

Work, 4-12, 4-13

## **X**

XOR (exclusive OR) connector symbol  
Compared to AND, 4-10

## **Y**

Yourdon, Edward, 8-9

Yourdon/DeMarco, 1-2

Yourdon/DeMarco dataflow diagrams  
Curved line, 4-6, 4-8