

# Analyzing Programs





IBM Asset Transformation Workbench  
v1.1



# Analyzing Programs

Note:

Before using this information and the product it supports, read the information in “Notices.”

**First Edition (February 2005)**

This edition applies to IBM Asset Transformation Workbench (product number 5724-L54) and to all subsequent releases and modifications until otherwise indicated in new editions.

For the latest information about this product, please refer to the Release Notes.

No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose, without the express written permission of IBM. Information in this document is subject to change without notice and is not guaranteed to be error-free.

You can order publications through your IBM representative or the IBM branch office serving your locality. When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Licensed Materials - Property of IBM.

Product Reference: IBM Asset Transformation Workbench v1.1

Document Reference: REL7.3.07.DOC04.A

**© 2005 Copyright International Business Machines Corporation. All rights reserved.**

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

**© 2004, 2005 Relativity Technologies, Inc. All rights reserved.**

RescueWare is a registered trademark of Relativity Technologies, Inc. All other brands mentioned in this document are trademarks or registered trademarks of their respective holders.

# Contents

## Preface

<i>Audience</i> . . . . .	xiii
<i>Organization</i> . . . . .	xiv
<i>Conventions</i> . . . . .	xiv
<i>Related Manuals</i> . . . . .	xv
<i>Online Help</i> . . . . .	xvi

## 1 Introducing HyperView

<i>Understanding HyperView Parse Trees</i> . . . . .	1-1
<i>Starting HyperView</i> . . . . .	1-3
<i>Using the HyperView Main Window</i> . . . . .	1-4
<i>Context Pane</i> . . . . .	1-5
<i>Source Pane</i> . . . . .	1-6
<i>Clipper Pane</i> . . . . .	1-6
<i>Impact Pane</i> . . . . .	1-7
<i>Callie Pane</i> . . . . .	1-7
<i>Execution Path Pane</i> . . . . .	1-8
<i>Flowchart Pane</i> . . . . .	1-8
<i>Rules Pane</i> . . . . .	1-9

<i>Components Pane</i> . . . . .	1-9
<i>Data Flow Pane</i> . . . . .	1-10
<i>Data View Pane</i> . . . . .	1-10
<i>PL/I Call Diagram Pane</i> . . . . .	1-11
<i>Bird's Eye Pane</i> . . . . .	1-12
<i>Model Reference Pane</i> . . . . .	1-12
<i>Using HyperView Auxiliary Windows</i> . . . . .	1-13
<i>Impact Report Window</i> . . . . .	1-13
<i>Animator Window</i> . . . . .	1-14
<i>Advanced Search Window</i> . . . . .	1-14
<i>Using Basic Navigation Controls</i> . . . . .	1-15
<i>Using the Properties Window</i> . . . . .	1-15
<i>Using the List Browser Window</i> . . . . .	1-18
<i>Opening a List</i> . . . . .	1-18
<i>Creating a List</i> . . . . .	1-20
<i>Exporting a List</i> . . . . .	1-20
<i>Deleting a List</i> . . . . .	1-20
<i>What's Next?</i> . . . . .	1-21

## **2 Viewing Program Source and Context**

<i>Using the Source Pane</i> . . . . .	2-1
<i>Selecting and Copying Code</i> . . . . .	2-2
<i>Navigating to Related Constructs</i> . . . . .	2-3
<i>Navigating to Multiple Occurrences of a Construct</i> . . . . .	2-3
<i>Searching for Code</i> . . . . .	2-4
<i>Specifying the Change Magnitude for a Source File</i> . . . . .	2-4
<i>Setting Source Pane Options</i> . . . . .	2-5
<i>Adjusting the Display</i> . . . . .	2-6
<i>Using the Context Pane</i> . . . . .	2-7
<i>What's Next?</i> . . . . .	2-8

## **3 Performing Advanced Searches**

<i>Understanding Advanced Searches</i> . . . . .	3-1
<i>Searching for Similarly Structured Code</i> . . . . .	3-6
<i>Executing Advanced Searches</i> . . . . .	3-8

	<i>Defining Advanced Search Criteria</i> . . . . .	3-9
	<i>What's Next?</i> . . . . .	3-10
<b>4</b>	<b>Staging Program Analysis with Clipper</b>	
	<i>Getting Started in Clipper</i> . . . . .	4-1
	<i>Using the Clipper Pane</i> . . . . .	4-5
	<i>Working with Categories</i> . . . . .	4-6
	<i>Working with Lists</i> . . . . .	4-6
	<i>Generating Business Rules</i> . . . . .	4-7
	<i>Showing Impacts</i> . . . . .	4-7
	<i>Creating Projects in Clipper</i> . . . . .	4-8
	<i>Generating Clipper Reports</i> . . . . .	4-9
	<i>Printing Reports</i> . . . . .	4-10
	<i>Exporting Reports</i> . . . . .	4-10
	<i>What's Next?</i> . . . . .	4-10
<b>5</b>	<b>Analyzing Impact Traces</b>	
	<i>Using the Impact Report Tool</i> . . . . .	5-2
	<i>Understanding Impact Report Depths</i> . . . . .	5-2
	<i>Generating an Impact Trace in the Impact Report Tool</i> . . . . .	5-3
	<i>Using the Impact Report Window</i> . . . . .	5-5
	<i>Setting Impact Report Options</i> . . . . .	5-6
	<i>Creating Projects in the Impact Report Tool</i> . . . . .	5-11
	<i>Exporting Impact Reports</i> . . . . .	5-12
	<i>Using the Impact Pane</i> . . . . .	5-13
	<i>Generating an Impact Trace in the Impact Pane</i> . . . . .	5-13
	<i>Setting Impact Pane Options</i> . . . . .	5-15
	<i>What's Next?</i> . . . . .	5-17
<b>6</b>	<b>Analyzing Program Control Flows</b>	
	<i>Using the Callie Pane</i> . . . . .	6-2
	<i>Choosing the Diagram View</i> . . . . .	6-3
	<i>Reconstructing Diagrams</i> . . . . .	6-4
	<i>Setting Callie Pane Options</i> . . . . .	6-4

<i>Using the Execution Path Pane</i> . . . . .	6-8
<i>Generating an Execution Path</i> . . . . .	6-8
<i>Using the Flowchart Pane</i> . . . . .	6-9
<i>Using the Animator</i> . . . . .	6-11
<i>What's Next?</i> . . . . .	6-12

## **7 Extracting Business Rules**

<i>Understanding Business Rules</i> . . . . .	7-1
<i>Guidelines for Assigning Segments</i> . . . . .	7-2
<i>Methods for Assigning Segments</i> . . . . .	7-2
<i>Understanding the Rules Pane</i> . . . . .	7-2
<i>Extracting Business Rules Manually</i> . . . . .	7-3
<i>Autodetecting Business Rules</i> . . . . .	7-6
<i>Flow to Field</i> . . . . .	7-7
<i>MOVES to Screen Message</i> . . . . .	7-7
<i>Handle Not Found</i> . . . . .	7-8
<i>I/O Rules</i> . . . . .	7-8
<i>Screen Validation</i> . . . . .	7-8
<i>Test on Field</i> . . . . .	7-9
<i>Extracting Business Rules with Clipper</i> . . . . .	7-10
<i>Validating Business Rules after Refreshing or Editing Code</i> . . . . .	7-11
<i>Using the Rules Pane</i> . . . . .	7-13
<i>Editing Rule Properties</i> . . . . .	7-14
<i>Assigning Property Values to Multiple Rules</i> . . . . .	7-22
<i>Filtering the Business Rules Display</i> . . . . .	7-23
<i>Generating Business Rule Reports</i> . . . . .	7-25
<i>Printing Reports</i> . . . . .	7-26
<i>Exporting Reports</i> . . . . .	7-26
<i>What's Next?</i> . . . . .	7-26

## **Glossary**

## **Bibliography**



**Notices**

**Index**



## ***Preface***

**T**he IBM Asset Transformation Workbench (ATW) is a suite of PC-based software products for analyzing, re-architecting, and transforming legacy applications. The products are deployed in an integrated environment with access to a common repository of program objects. Repository models serve as the basis for a rich set of diagrams, reports, and other documentation.

The ATW suite consists of customizable modules that together address the needs of organizations at every stage of legacy application evolution — maintenance/enhancement, renovation, and modernization.

### ***Audience***

This guide assumes that you are a corporate Information Technology (IT) professional with a working knowledge of the legacy platforms you are using the product to analyze. If you are transforming a legacy application, you should also have a working knowledge of the target platform.

## Organization

This guide contains the following chapters:

- Chapter 1, “Introducing HyperView,” provides an overview of the HyperView tool.
- Chapter 2, “Viewing Program Source and Context,” describes how to use the Source and Context panes to examine legacy programs interactively.
- Chapter 3, “Performing Advanced Searches,” describes how to use the HyperView advanced search facility to create sophisticated filters for construct searches.
- Chapter 4, “Staging Program Analysis with Clipper,” describes how to use the Clipper tool to create lists of candidates for business rule extraction, event injection, impact analysis, and other tasks.
- Chapter 5, “Analyzing Impact Traces,” describes how to use the Impact pane and Impact Report tool to perform impact analysis.
- Chapter 6, “Analyzing Program Control Flows,” describes how to use the Callie, Execution Path, Flowchart, and Animator panes to examine the procedure flow for a legacy program.
- Chapter 7, “Extracting Business Rules,” describes how to use the Rules pane to extract business rules.
- The Glossary defines the names, acronyms, and special terminology used in this guide.

## Conventions

This guide uses the following typographic conventions:

- **Bold type** — Indicates a specific area within the graphical user interface, such as a button on a screen, a window name, or a command or function.
- *Italic type* — Indicates a new term. Also indicates a document title. Occasionally, italic type is used for emphasis.

- `Monospace type` — Indicates computer programming code.
- **bold monospace type** — Indicates input you type on the computer keyboard.
- **1A/1B, 2A/2B** — In task descriptions, indicates mutually exclusive steps: perform step A or step B, but not both.

## Related Manuals

This document is part of a complete set of ATW manuals. Together they provide all the information you need to get the most out of the system.

- *Getting Started* introduces ATW. This guide provides an overview of the workbench tools and discusses basic concepts. It describes how to install the product and how to manage licenses. It also describes how to use common product features.
- *Preparing Projects* describes how to set up ATW projects. This guide describes how to load applications in the repository and how to use reports and other tools to ensure that the entire application is available for analysis.
- *Analyzing Projects* describes how to analyze applications at the project level. This guide describes how to create diagrams of applications and how to perform impact analysis across applications. It also describes how to estimate project complexity and effort, and how to create a project dictionary.
- *Profiling Projects* describes how to create and browse Web-generated views of the repositories in your organization.
- *Creating Components* describes how to extract program components from a legacy application.
- *Parser Reference Manual* describes legacy constructions supported by Application Analyzer in reference format.
- *Architecture Reference Manual* describes legacy constructions supported by Application Architect in reference format.

## Online Help

In addition to the manuals provided with the system, you can learn about the product using the integrated online help. All GUI-based tools include a standard Windows **Help** menu.

You can display:

- The entire help system, with table of contents, index, and search tool, by selecting **Help: Help Topics**.
- Help about a particular ATW window by clicking the window and pressing the **F1** key.

Many ATW tools have *guides* that you can use to get started quickly in the tool. The guides are help-like systems with hyperlinks that you can use to access functions otherwise available only in menus and other program controls.

To open the guide for a tool, choose **Guide** from the **View** menu. Use the table of contents in the **Page** drop-down to navigate quickly to a topic.

## *Introducing HyperView*



**M**uch of the power of ATW resides in a set of program analysis tools collectively called *HyperView*. HyperView lets you analyze legacy programs interactively, by examining synchronized, complementary views of the same information — source, context, impacts, and so forth. You can use HyperView to analyze procedure and data flows, stage program analyses, and extract business rules.

HyperView is designed primarily for programs, but you can also use it to analyze JCL or ECL files, DDLs or Natural Data Definition Modules, PSB and DBD files, and IDMS schemas.

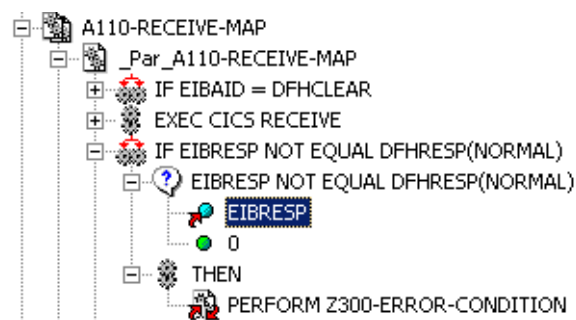
### *Understanding HyperView Parse Trees*

When you verify a legacy source file, ATW creates a *parse tree* that defines the relationships among the constructs that comprise the file — its sections, paragraphs, statements, conditions, variables, and so forth. A variable, for example, can be related in the parse tree to any of three other constructs — a declaration, a dataport, or a condition. You view the parse

tree for a source file in the HyperView Context pane (see [“Context Pane” on page 1-5](#)).

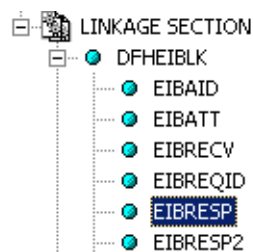
Figure 1-1 shows a portion of the parse tree for the GSS5.CBL program. The parse tree shows that the program executes a PERFORM statement if the value of the variable EIBRESP satisfies the condition EIBRESP NOT EQUAL DFHRESP(NORMAL).

Figure 1-1 Use of EIBRESP Variable



If you are interested in investigating other uses of EIBRESP in the program, you can navigate to the declaration of the variable in the parse tree (Figure 1-2). Select EIBRESP in the Context pane and choose **Declaration** in the **Edit** menu. The **Edit** menu displays choices for each type of construct to which the selection is related.

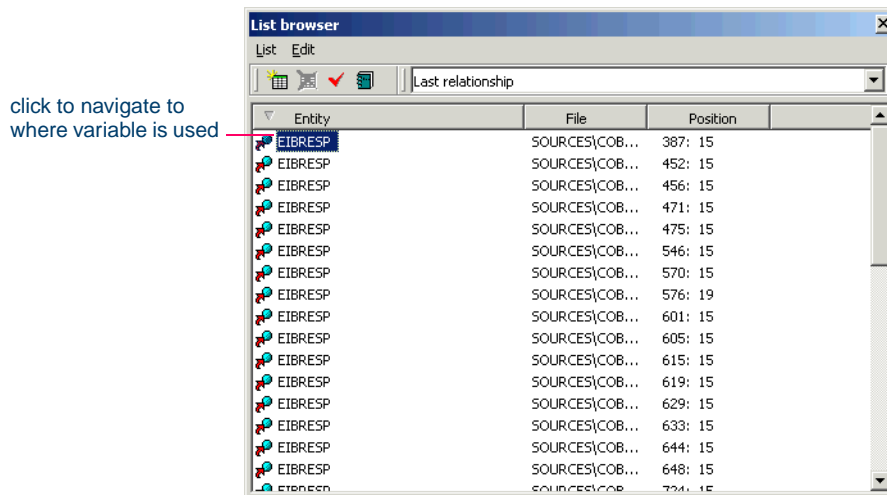
Figure 1-2 Declaration of EIBRESP Variable



From the declaration, you can generate a list of instances in which the variable is used (Figure 1-3 on page 1-3). Select the declaration in the Context pane and choose **Instances** in the **Edit** menu.



Figure 1-3 List of Instances of EIBRESP Variable



Select an instance in the list to navigate to that instance in the parse tree. You can also use the Properties window to navigate to related constructs, as described in [“Using the Properties Window” on page 1-15](#).

## Starting HyperView

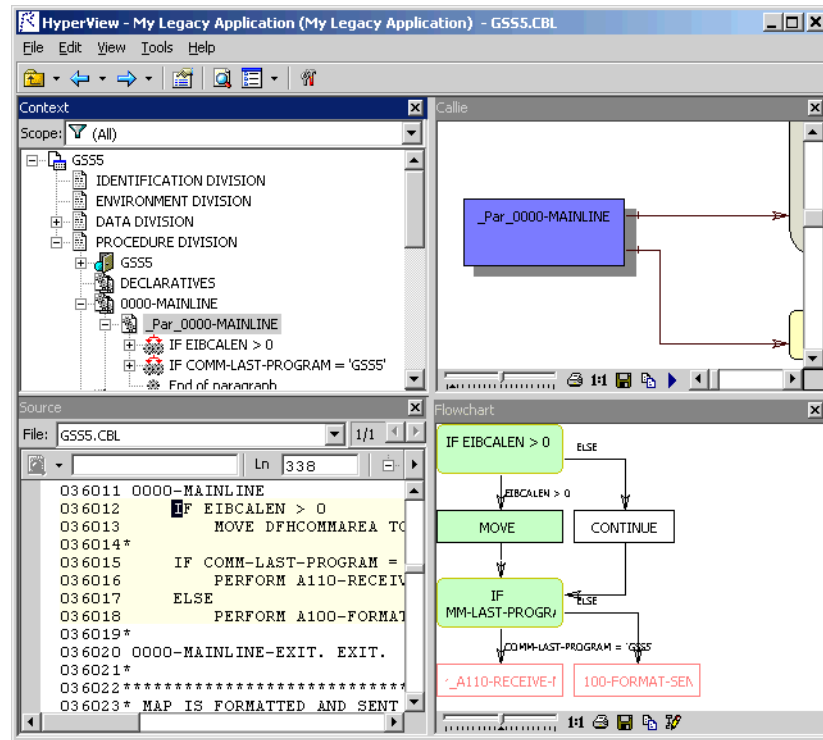
You can invoke HyperView directly by selecting a source file in the ATW Repository Browser and choosing **Interactive Analysis** in the **Analyze** menu. HyperView is also available within the project analysis tools — Diagrammer, Change Analyzer, and so forth — but “silently,” that is, without your ever actually invoking it as such. Tool usage is identical in either case.

### To invoke HyperView directly:

- 1 In the Repository Browser, select the source file you want to analyze and choose **Interactive Analysis** in the **Analyze** menu. The HyperView window opens. Figure 1-4 on page 1-4 shows a typical HyperView configuration.

1-4 Introducing HyperView  
Using the HyperView Main Window

Figure 1-4 HyperView Window — Typical Configuration



## Using the HyperView Main Window

The HyperView main window consists of a series of panes that offer complementary views of code constructs in the selected source file. The display in each pane is synchronized with the others — selecting a construct in one pane automatically moves the cursor to the construct in the other panes.

The first time you open HyperView it displays the Source and Context panes. Select the appropriate choice in the **View** menu to show the other panes. Use the choices at the top of the **View** menu to configure the panes in logical groupings. Choose **Control Flow**, for example, to view the Source, Flowchart, Callie, and Animator panes. You can hide a pane by clicking the close box in the upper righthand corner.

A pane is available only if it has been configured for the workbench and if the view it offers is relevant for the selected source file — you will not see a Data Flow pane for a copybook, for example. **Edit** menu choices and pane-specific menu choices have equivalent choices in the right-click menu.

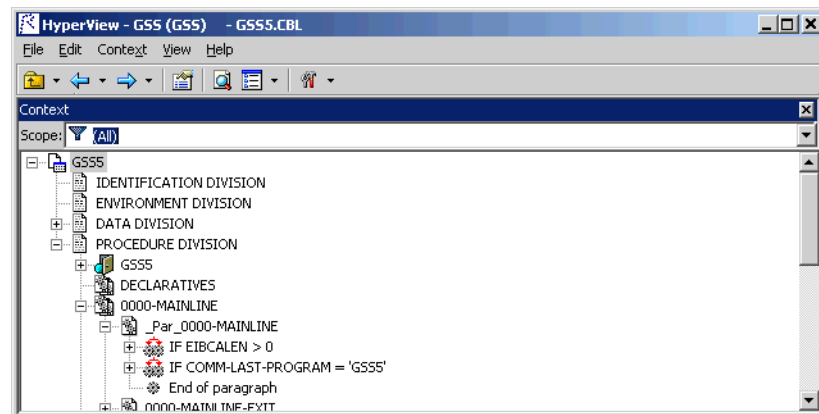
The HyperView window's state — size, location, configuration, and option settings — is saved across sessions. For guidance on moving and resizing HyperView panes, and on configuring the workbench, see *Getting Started* in the ATW document set

In addition to the panes described below, HyperView provides access to the standard ATW Activity Log, User Preferences window, and Project Options window. To view the standard tools, select the appropriate choice in the **View** menu. For tool usage, see *Getting Started* in the ATW document set.

## Context Pane

The Context pane (Figure 1-5) displays the *parse tree* for the selected source file. The parse tree displays source code constructs — sections, paragraphs, statements, conditions, variables, and so forth — in hierarchical form, making it easy to locate code constructs quickly. Constructs are displayed in the order they appear in your code.

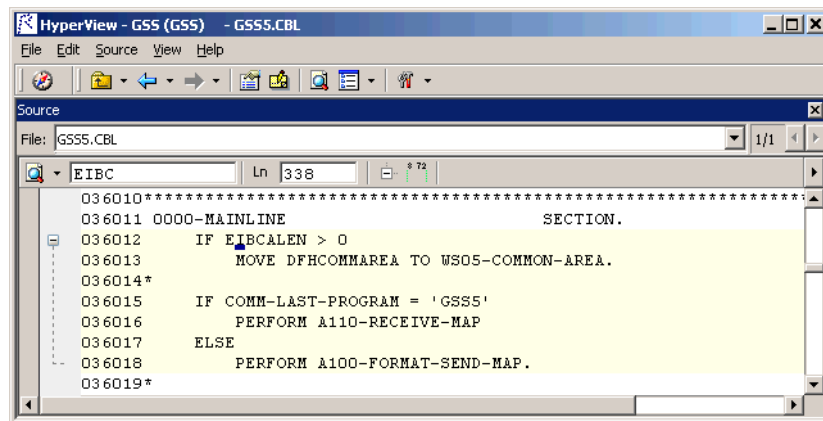
Figure 1-5 *Context Pane*



## Source Pane

The Source pane (Figure 1-6) displays view-only source code for the selected file and included files. Sophisticated search facilities let you navigate to code constructs quickly.

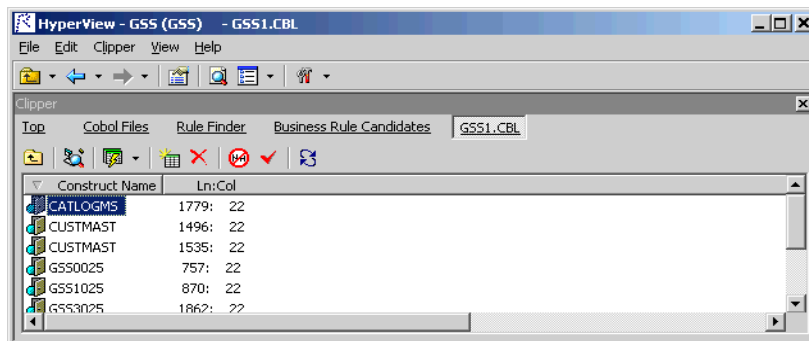
Figure 1-6 Source Pane



## Clipper Pane

The Clipper pane (Figure 1-7) lets you create lists of candidates for business rule extraction, event injection, impact analysis, and other tasks. Each list captures the results of a different stage of your analysis and serves as input for subsequent tasks.

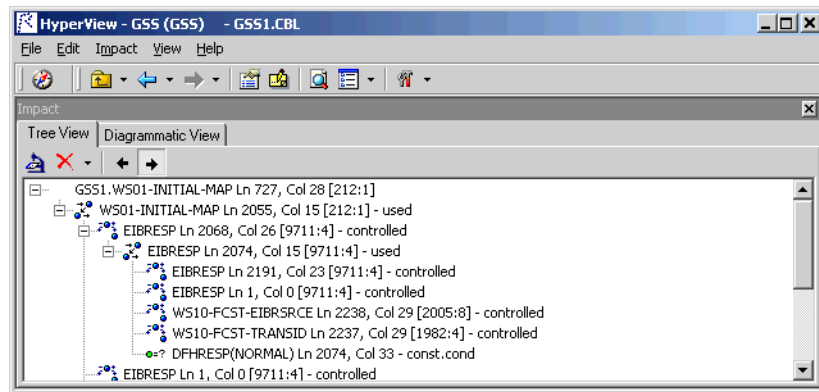
Figure 1-7 Clipper Pane



## Impact Pane

The Impact pane (Figure 1-8) displays a hierarchical view and diagram of the *impact trace* for a program variable. An impact trace describes how data items interact with each other in a program — exchange values, use each other in computations, and so forth.

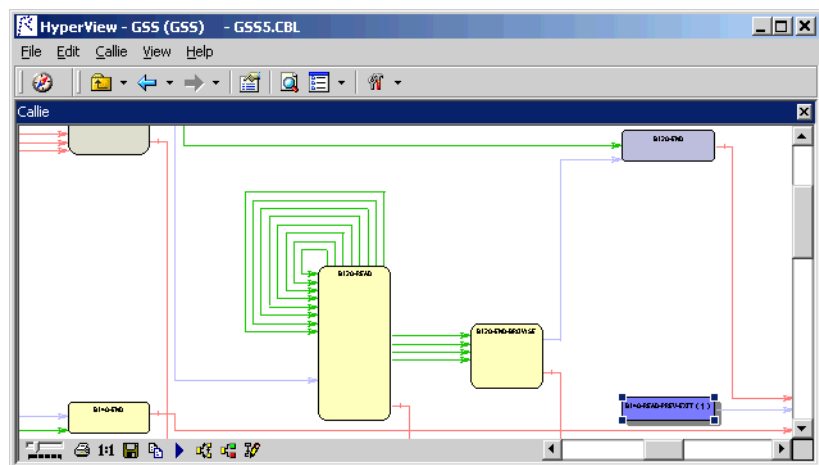
Figure 1-8 *Impact Pane (Hierarchy View)*



## Callie Pane

The Callie pane (Figure 1-9) displays a diagram that shows the flow of control between paragraphs or procedures in a program.

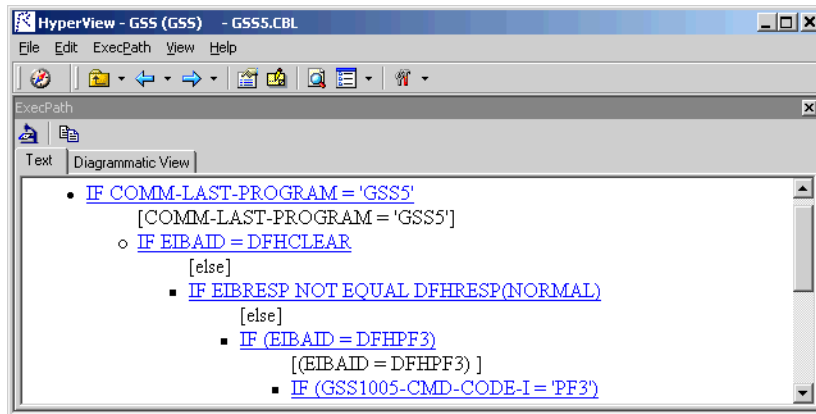
Figure 1-9 *Callie Pane*



### Execution Path Pane

The Execution Path pane (Figure 1-10) displays a hierarchical view and diagram of the conditions that determine the flow of control in a program.

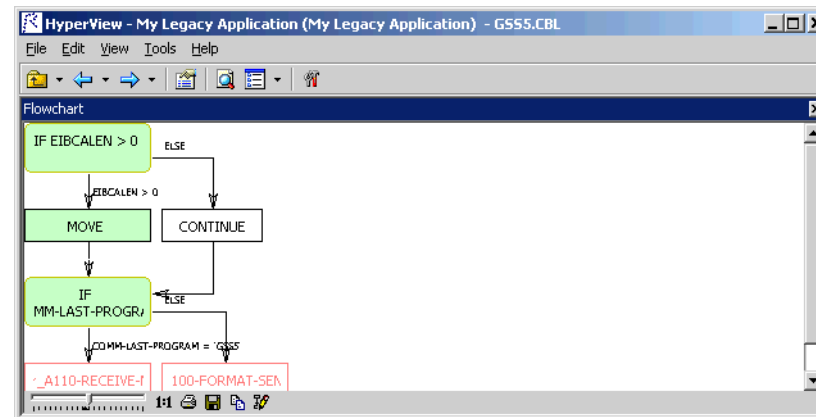
Figure 1-10 Execution Path Pane (Hierarchy View)



### Flowchart Pane

The Flowchart pane (Figure 1-11) displays a diagram of the flow of control between statements in a paragraph or procedure.

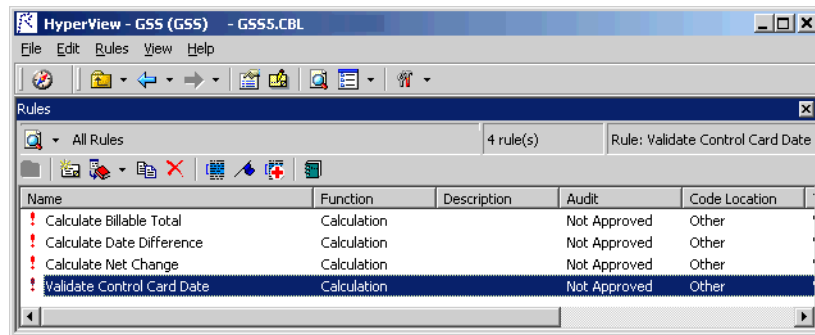
Figure 1-11 Flowchart Pane



## Rules Pane

The Rules pane (Figure 1-12) lets you create *business rules* from code segments extracted manually from source or autodetected. You can also create business rules from candidates listed in Clipper. A business rule identifies a code segment according to its business function.

Figure 1-12 Rules Pane

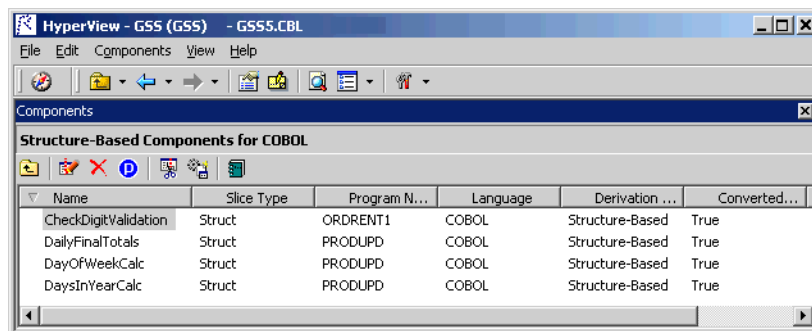


## Components Pane

The Components pane (Figure 1-13) lets you extract *components* from legacy source. A component is a self-contained program that can be re-used with other programs in modular fashion.

**Note:** For Components pane usage, see *Creating Components* in the ATW document set.

Figure 1-13 Components Pane

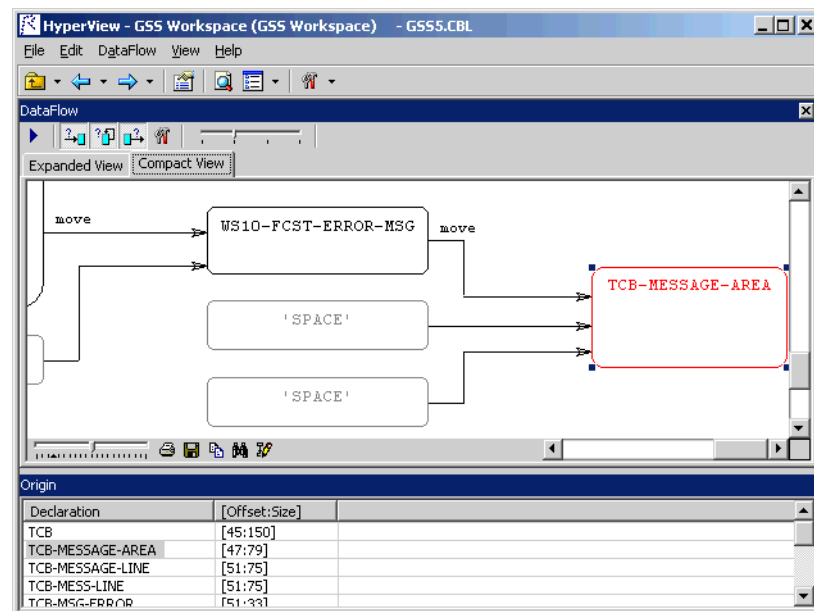


### Data Flow Pane

The Data Flow pane (Figure 1-14) displays a diagram of the incoming and outgoing data flows for a program variable up to a *dataport* — an I/O statement or a call to or from another program. It also displays a list of variable offsets and memory allocations.

**Note:** Usage for the Data Flow and Data View panes (see below) is identical to that for the corresponding panes in the Global Data Flow tool. For more information, see *Analyzing Projects* in the ATW document set.

Figure 1-14 Data Flow Pane

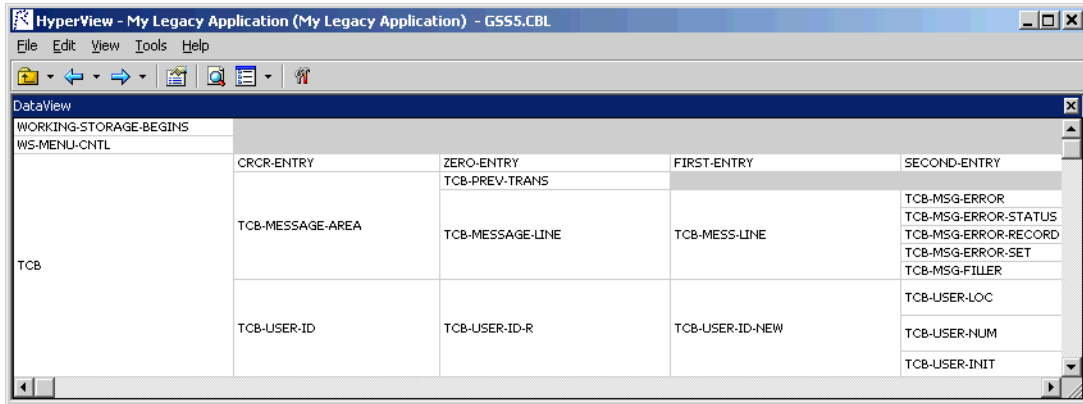


### Data View Pane

The Data View pane (Figure 1-15 on page 1-11) displays program variable structures, substructures, and fields in a hierarchical grid that visually represents memory usage.



Figure 1-15 Data View Pane

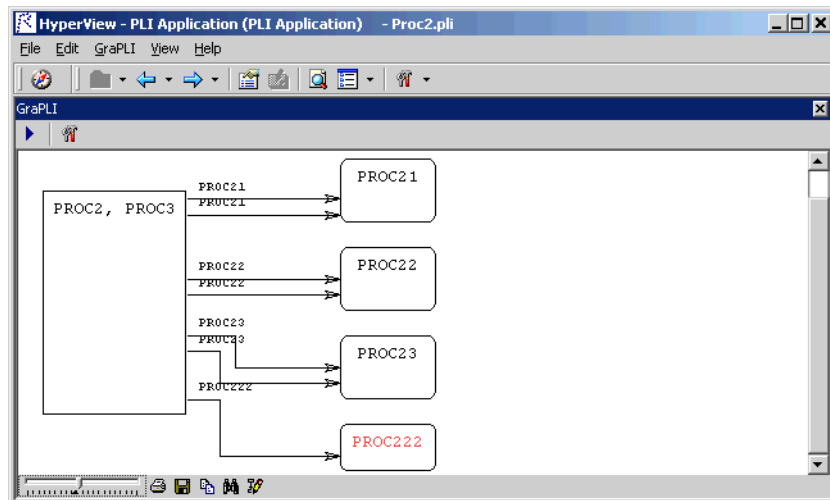


### PL/I Call Diagram Pane

The PL/I Call Diagram pane (Figure 1-16) displays the call relationships between PL/I external and internal procedures. Use it to perform low-level analysis of PL/I programs.

**Note:** PL/I Call Diagram usage is described in *Analyzing Projects* in the ATW document set.

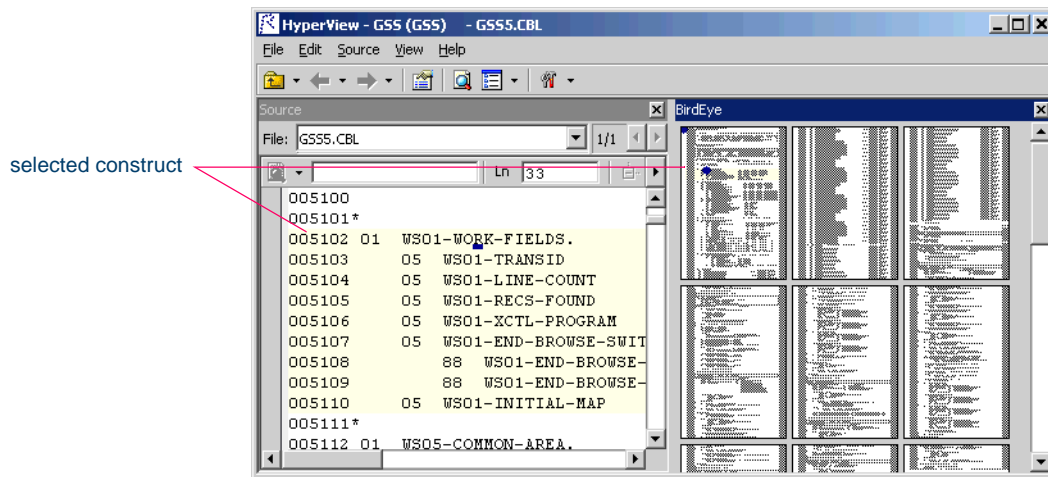
Figure 1-16 PL/I Call Diagram Pane



### Bird's Eye Pane

The Bird's Eye pane (Figure 1-17) works with the Source pane to let you quickly identify the location of a code construct relative to the entire program and any included files.

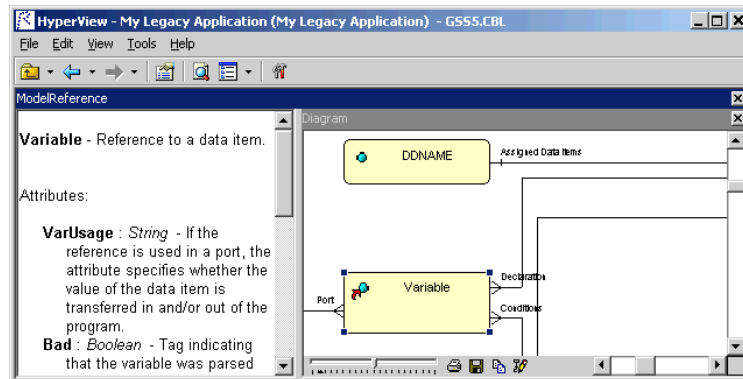
Figure 1-17 Bird's Eye Pane



### Model Reference Pane

The Model Reference pane (Figure 1-18) displays the parse tree meta-model in text and diagram form.

Figure 1-18 Model Reference Pane



## Using HyperView Auxiliary Windows

HyperView provides auxiliary windows for impact analysis, code animation, and construct searches. Displays are synchronized with the HyperView panes.

To open the impact analysis and code animation windows, select a construct in the Context or Source panes, then select **Impact Report** or **Animator** in the HyperView **View** menu. To open the search window, select **Find** in the **Edit** menu.

### Impact Report Window

Like the Impact pane (see [“Impact Pane” on page 1-7](#)), the Impact Report window (Figure 1-19) displays an impact trace for a program variable. But where the Impact pane is better-suited for data flow analysis, the Impact Report window is a more traditional impact analysis tool — it lets you hone in on the data items in a project that may be impacted by changing another data item’s definition or usage.

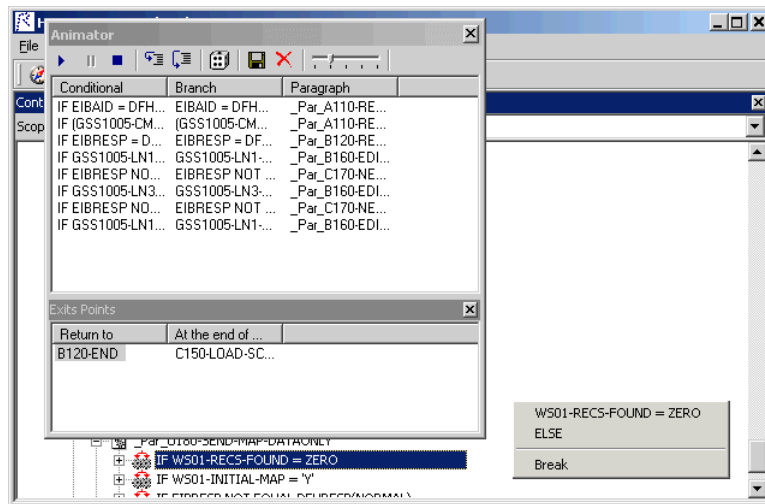
Figure 1-19 *Impact Report Window with Impact Trace*

Depth	Item	Statement	Relationship	Object Name	File
0	W501-LINE-C...	IF W501-LINE-...	start	G555	SOURCES\COB
1	FA01-KEY	FCSTADJ	controlled	G555	SOURCES\COB
2	FA01-KEY	MOVE FA01-KE...	used	G555	SOURCES\COB
3	W510-SCR-KEY	MOVE FA01-KE...	comp@	G555	SOURCES\COB
4	W510-SCR-YY	MOVE W510-S...	used	G555	SOURCES\COB
5	G551005-LN1-...	MOVE W510-S...	comp@	G555	SOURCES\COB
6	G551005O	G551005	used	G555	SOURCES\COB
7	G551005I	G551005	used	G555	SOURCES\COB
8	G551005-LN1-...	MOVE G55100...	used	G555	SOURCES\COB
9	COMM-SELC-Y...	MOVE G55100...	move	G555	SOURCES\COB
10	W505-COMMO...	83406288	used	G555	SOURCES\COB
11	W505-COMMO...	MOVE DFHCO...	used	G555	SOURCES\COB
12	W505-COMMO...	G554	used	G555	SOURCES\COB
13	DFHCOMMAREA	G554	calls	G554	SOURCES\COB
14	DFHCOMMAREA	G55	used	G554	SOURCES\COB
15	DFHCOMMAREA	G55	calls	G55	SOURCES\COB
16	DFHCOMMAREA	MOVE DFHCO...	used	G55	SOURCES\COB
17	W505-COMMO...	MOVE DFHCO...	move	G55	SOURCES\COB
18	EIBCALEN	IF EIBCALEN = 0	controlled	G55	SOURCES\COB
19	EIBRESP	ASSIGN	controlled	G55	SOURCES\COB
20	EIBRESP	IF EIBRESP NO...	used	G55	SOURCES\COB

### Animator Window

The Animator (Figure 1-20) lets you step through the code displayed in a HyperView pane. You can choose program branches yourself, or have the Animator choose them randomly.

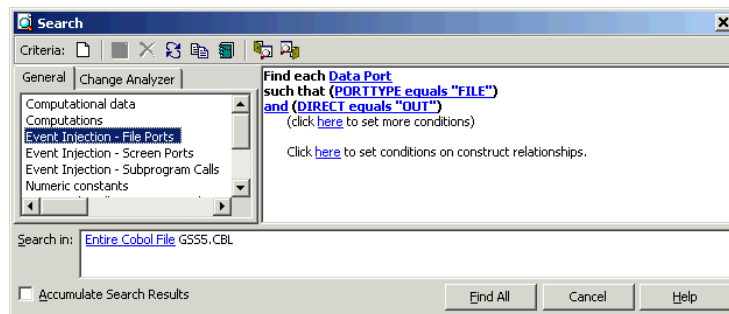
Figure 1-20 Code Animator Window with Context Pane



### Advanced Search Window







HyperView provides simple and advanced search facilities. Use the Advanced Search window (Figure 1-21) to create sophisticated filters for construct searches.

Figure 1-21 Advanced Search Window



## Using Basic Navigation Controls

HyperView offers Explorer-style navigation controls on the tool bar, with corresponding choices in the **Edit** menu:

- Click the  button on the tool bar to navigate backward in the history of your selections in the HyperView window (regardless of the pane in which they were made). Click the adjacent  button to display the selection history in a drop-down menu. Choose a selection by clicking it in the menu.
- Click the  button on the tool bar to navigate forward in the history of your selections in the HyperView window (regardless of the pane in which they were made). Click the adjacent  button to display the selection history in a drop-down menu. Choose a selection by clicking it in the menu.
- Click the  button on the tool bar to navigate to the parent of the selected construct in the parse tree. Click the adjacent  button to display all of the ancestors of the selected construct in a drop-down menu. Choose a selection by clicking it in the menu.

## Using the Properties Window

HyperView Properties windows offer convenient ways to navigate program code and analyze program constructs. The Properties window for a construct identifies its:

- *Attributes* — the construct's type, name, location, and characteristics.
- *Relationships* — how the construct interacts with other constructs.
- *Structure* — the context of the construct in the parse tree.

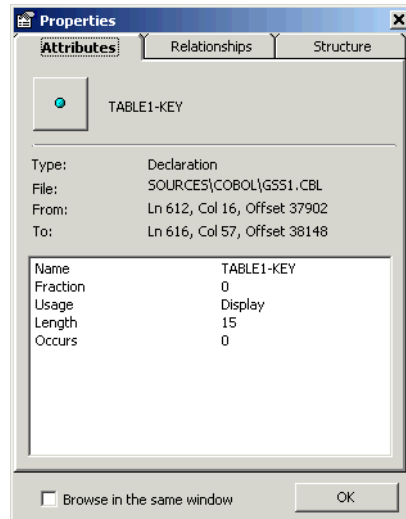
Choose a construct in a relationship list or structure tree to navigate to the construct in a HyperView pane. Properties windows are not modal, so you can leave the windows up throughout your HyperView session, and open them for as many constructs as necessary.

### **To view construct properties:**

- 1 In any HyperView pane except the Model Reference pane, select the construct whose properties you want to view and choose **Properties**

in the **Edit** menu. The Properties window opens. Click the Attributes tab (Figure 1-22).

Figure 1-22 Properties Window — Attributes Tab



- 2 The Attributes tab displays the construct's type, name, location, and characteristics. Click the Relationships tab (Figure 1-23 on page 1-17).

**Tip:** Choose **Browse in the same window** only if you do *not* want to open Properties windows for other constructs concurrently.

- 3 The Relationships tab lists the types of relationships available for the selected construct. If relationships of that type exist, they are listed below the relationship type. Click the plus sign (+) next to a relationship type to expand the list. Click the minus sign (-) to collapse the list. Click a construct in the list to navigate to it in an open HyperView pane.


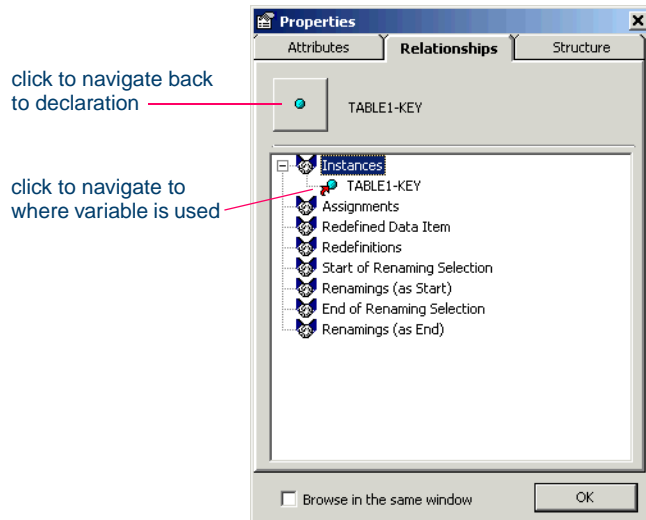
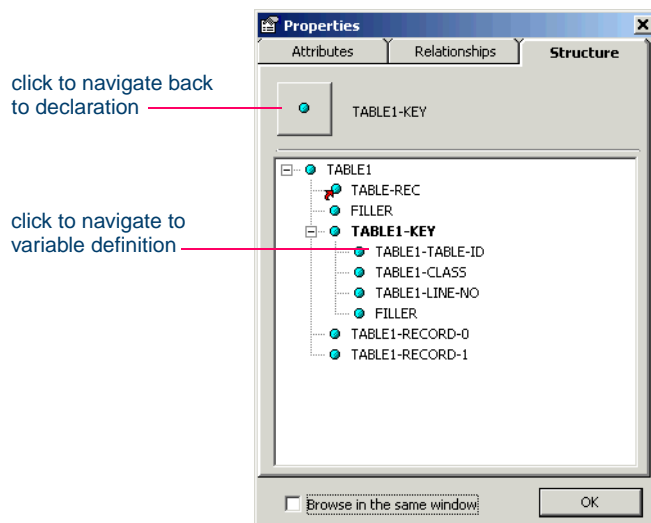
**Tip:** Click the  button below the Attributes label to navigate back to the construct described in the Properties window.

Figure 1-23 Properties Window — Relationships Tab



- 4 Click the Structure tab. The Structure tab opens (Figure 1-24).

Figure 1-24 Properties Window — Structure Tab



- 5 The Structure tab displays the context of the selected construct in the parse tree, from its parent to its children. Click a construct in the structure tree to navigate to it in an open HyperView pane.
- 6 Click **OK** to dismiss the window.



## **Using the List Browser Window**

HyperView lists offer convenient ways to navigate program code and record the results of program analyses. HyperView generates lists whenever you perform advanced searches, navigate to a related construct, or identify candidates in Clipper. You can export HyperView lists to a variety of standard formats, and create customized lists.

**Note:** Some HyperView lists are generated by external tools, such as Component Maker.

### **Opening a List**

Each time you perform an advanced search or navigate to a related construct, HyperView generates a list of target constructs. The last search or relationship results overwrite previous results. These lists are predefined and cannot be modified or deleted.

- If more than one target construct is listed, HyperView automatically opens the list in the List Browser (Figure 1-3 on page 1-3) and moves the cursor to the first item in the list in every open HyperView pane.
- If only one target construct is listed, HyperView simply moves the cursor to that construct in every open HyperView pane. To view the list, click the  button next to the  button on the tool bar and choose the list from the drop-down menu of available lists.

Similarly, each time you select a program that contains candidates for business rule extraction, event injection, or impact analysis in Clipper, HyperView generates a list of target candidates. Because you can view the list directly in the Clipper window, HyperView does not automatically open it in the List Browser. These lists can be deleted but not modified.



**To open a list:**



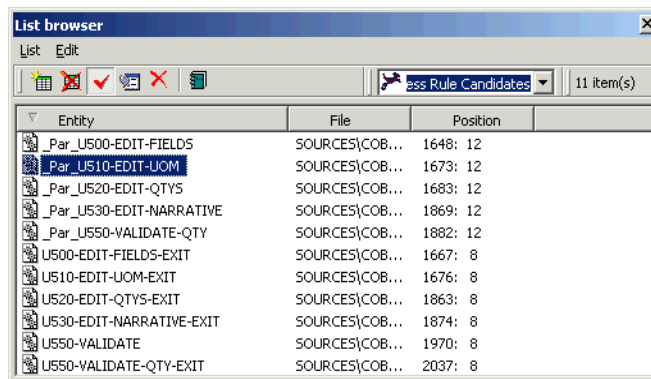
- 1 Click the  button next to the  button on the HyperView tool bar and choose the list you want to open from the drop-down menu of available lists. The list opens in the List Browser (Figure 1-25).

Figure 1-25 List Browser




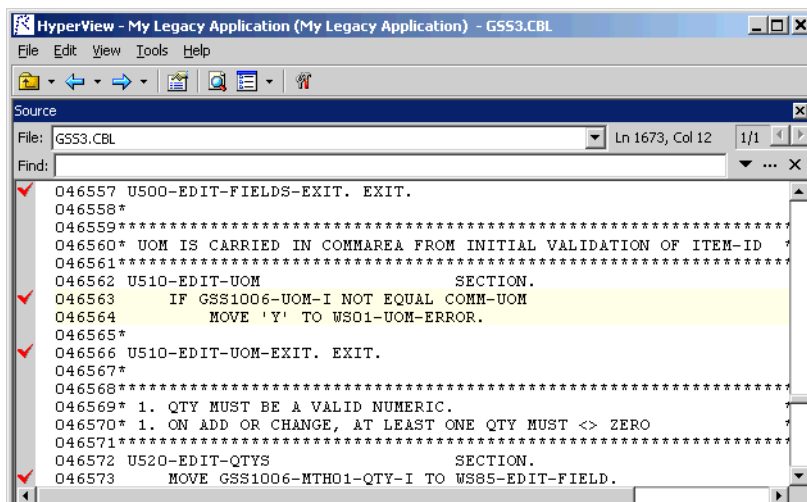
**Tip:** To place a check mark in the HyperView Source pane display for each item in a list (Figure 1-26), click the  button on the List Browser tool bar.



Figure 1-26 Source Pane with Checked List Items



## Creating a List

When you create a list, you can copy constructs to it from another list or directly from the HyperView main window.

### To create a list:

- 1 Click the  button next to the  button on the HyperView tool bar and choose **New list** from the drop-down menu of available lists. The New List dialog opens. Enter the name of the new list in the text field and click **OK**. The new list opens in the List Browser (Figure 1-25 on page 1-19).
- 2A To add constructs from another list, choose the list in the drop-down on the List Browser tool bar. When the list opens, select the constructs you want to add and choose **Copy to:New List** in the **Edit** menu. Use the drop-down on the tool bar to return to the new list.
- 2B To add a construct from the HyperView main window, select the construct and choose **Add Selected Construct** in the **Edit** menu.

**Tip:** Click a column heading in the List Browser to sort the entries by that column. Grab-and-drag the border of a column heading to increase or decrease the width of the column.

To delete an entry in a user-defined list, select it and choose **Delete** in the **Edit** menu. To move an entry from one user-defined list to another user-defined list, select the entry and choose **Move to:New List** in the **Edit** menu.

## Exporting a List

You can export lists to a variety of standard formats. Choose **Save** in the **File** menu to export a list to HTML, Excel, RTF, Word, or formatted text.

## Deleting a List

You can delete a list by opening it in the List Browser and choosing **Delete** in the **List** menu. You cannot delete the predefined search or relationship lists.

## **What's Next?**

That completes your introduction to HyperView! Now let's take a closer look at how you use HyperView to:

- Analyze program source and context.
- Perform advanced searches.
- Stage program analysis.
- Analyze impacts.
- Analyze program control flows.
- Extract business rules.

### ***Where to Look for Documentation on Other HyperView Tasks***

Two other tasks you can perform with HyperView — analyzing program data flows and PL/I call flows — are described in *Analyzing Projects* in the ATW document set. A third task — creating logical components — is described in *Creating Components*.

**1-22** Introducing HyperView  
*What's Next?*

## Viewing Program Source and Context



The HyperView Source and Context panes offer complementary views of legacy source code. The Source pane displays view-only source for the selected file. The Context pane displays the same code in hierarchical form, in a *parse tree* that defines the relationships among the code constructs that comprise the source.

### Using the Source Pane

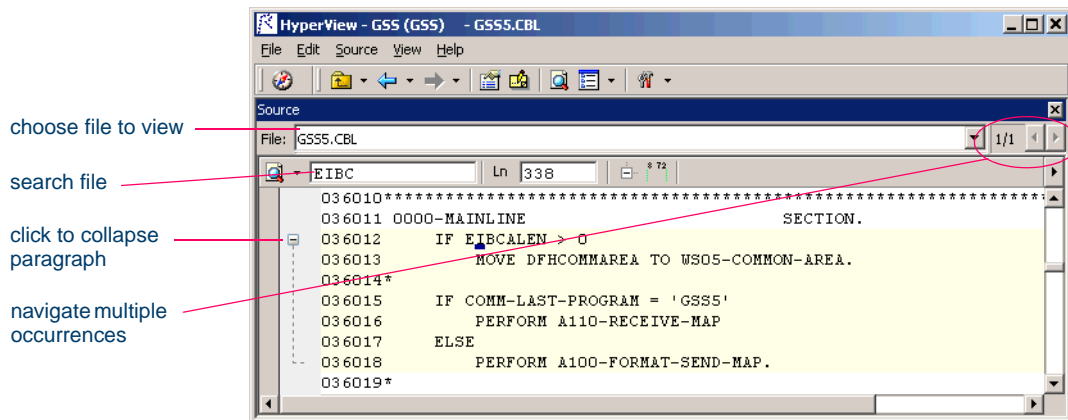
The Source pane displays view-only source for the selected legacy file (Figure 2-1 on page 2-2). The name of the file appears in the **File** drop-down. You can display the source code for an included file by choosing the file in the **File** drop-down.

Click in the Source pane to navigate to a construct. Enter a line number in the **Ln** field above the source code and press Enter to navigate to the line. Tool tips show line numbers in the source file when you scroll the Source pane vertically.

**Note:** Use the Editor in the ATW main window to modify program source code. For Editor usage, see *Getting Started* in the workbench documentation set.

## 2-2 Viewing Program Source and Context Using the Source Pane

Figure 2-1 Source Pane



### Selecting and Copying Code

There are two ways to select and copy code in the Source pane — as *construct* or as *text*. Which you choose depends on the task you want to perform.

#### Selecting and Copying Constructs

Click inside a code construct in the Source pane to select it. The selected construct is highlighted in yellow. The number of the first line of the selection is displayed in the **Ln** field above the source code. To copy the construct to the clipboard, choose **Copy Selected Construct** in the **Source** menu.

#### Selecting and Copying Text

Copy code as text when you are assigning code segments to business rules manually and want to select either more or less code than a construct contains. To select a code segment as text, click-and-drag from the first line of the segment to the last line of the segment. The selected text is highlighted in blue. To copy the selected segment to the clipboard, choose **Copy Selected Text** in the **Source** menu.

**Tip:** Click the minus sign (-) next to a paragraph, procedure, or data definition to collapse its contents. Click the plus sign (+) to expand its contents.

### ***Navigating to Related Constructs***

The **Edit** menu lists every relationship type for the construct selected in the Source pane. If you select a variable construct in the Source pane, for example, the **Edit** menu shows **Conditions**, **Port**, and **Declaration** choices. The choices are grayed-out if no relationships of the given type exist for the selected construct.

To view all the constructs in the source file that have a given relationship with a construct, select the construct in the Source pane and choose the appropriate relationship in the **Edit** menu.

- If only one construct has the specified relationship, HyperView simply moves the cursor to that construct in every open HyperView pane.
- If more than one construct has the specified relationship, HyperView automatically opens a list of related constructs in the List Browser and moves the cursor to the first item in the list in every open HyperView pane. To navigate to another item in the list, choose it in the List Browser. For List Browser usage, see [“Using the List Browser Window” on page 1-18.](#)

**Tip:** For a practical example of how you can use relationships to locate constructs of interest, see [“Understanding HyperView Parse Trees” on page 1-1.](#)



### ***Navigating to Multiple Occurrences of a Construct***


If a Cobol copybook, PL/I Include file, or Natural Include file is referenced multiple times in a program — in different structures, for example — you can use the arrows in the upper righthand corner of the Source pane (Figure 2-1 on page 2-2) to navigate between each occurrence of an included construct in the Context pane.

In the Source pane, click on the construct in the included file. The numbers in the upper righthand corner show the sequence number of the current construct versus the total number of constructs. The notation “2/3,” for example, identifies the second occurrence of a construct that occurs three times. Use the arrows to navigate between each occurrence. For Context pane usage, see [“Using the Context Pane” on page 2-7.](#)

## Searching for Code

HyperView provides simple and advanced search facilities for source code. For advanced search usage, see [Chapter 3, “Performing Advanced Searches.”](#)

To use the simple search facility, enter the text for the search in the field next to the  button on the tool bar. HyperView locates text matches as you type. Click the  button or press Enter to navigate to the next matching construct.

To view matching constructs in a list, click the adjacent  button. From the drop-down menu, choose:

- **Find All** to display pattern matches in a list.
- **Wildcard Find All** to display wildcard pattern matches in a list. You can use wildcard patterns allowed in LIKE statements by Visual Basic for Applications (VBA).
- **Recent Search List** to display the results of the last simple search.

Double-click an item in a list to navigate to it in the Source pane.

## Specifying the Change Magnitude for a Source File

Suppose you are planning to implement a change request and want to know how long it will take to complete the change. In this situation, you will typically run an *effort estimation* for your project based on weighted values for selected complexity metrics.

But what if your own analysis of the project shows that a given program will actually take much less time to change than the weighted calculation would suggest. The program might have thousands of source lines, for example, increasing its calculated complexity, while actually being very easy to modify.

A *change magnitude* is a way of overriding the calculated value for a source file. Your “subjective” estimate of the effort involved — Small, Medium, Large, Extra Large — becomes an input to the effort calculation, along with the weighted values.

To specify a change magnitude for a source file, right-click the file in the Source or Context window. In the right-click menu, set the change mag-



nitude to S for Small, M for Medium, L for Large, or XL for Extra Large. For more information on effort estimation, see *Analyzing Projects* in the ATW document set.

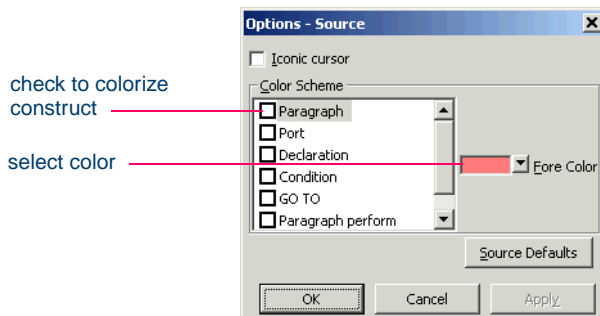
### Setting Source Pane Options



In very complex programs, it may be helpful to color-code source constructs. And it's almost always a good idea to specify that the icon for a selected construct replace the standard cursor. Use the Options window for the Source pane to color-code source constructs and "iconize" the cursor.

#### To change display options:

- 1 In the **Source** menu, choose **Options**. The Options window opens (Figure 2-2).

Figure 2-2 *Options Window*



- 2 In the Color Scheme pane, place a check mark next to the construct type you want to display in color. The current color of the type (if any) is displayed in the **Fore Color** drop-down.
- 3 Click the adjacent  button to edit the color of the construct type. A standard Windows color control is displayed. Use the Palette tab to select the color from the Windows palette. Use the System tab to match the color with the color of standard Windows elements.
- 4 Select **Iconic cursor** to specify that the icon for a selected construct ( for a data port, for example) replace the standard cursor.

**Tip:** You can restore the default options settings by clicking **Source Defaults**, then choosing **Restore Source Defaults** in the drop-down menu. Choose **Save To** in the drop-down menu to save the option settings to a file. Choose **Load From** in the menu to restore the option settings from a file.

- 5A Click **Apply** if you want to save your settings without dismissing the Options window.
- 5B Click **OK** if you want to save your settings and dismiss the Options window.

### ***Adjusting the Display***

Use the choices in the **Source** menu to collapse and expand code, and demarcate code from preceding and trailing enumeration characters.

#### ***Collapsing and Expanding Paragraphs or Subprograms***

In very long programs, it may be helpful to display the names of paragraphs or subprograms only and hide their source code. To collapse paragraphs or subprograms, choose **Collapse All** in the **Edit** menu. Click the plus sign (+) next to a collapsed paragraph or subprogram to show its source code again. To show the source code for all collapsed paragraphs or subprograms, choose **Expand All** in the **Edit** menu.

#### ***Showing Source Code Boundaries***

To demarcate source code from leading and trailing enumeration characters in source lines, choose **Show Boundaries** in the **Edit** menu.

#### ***Using the Bird's Eye and Model Reference Panes***

Use the Bird's Eye pane (Figure 1-17 on page 1-12) with the Source pane to quickly identify the location of a code construct relative to the entire program and any included files.

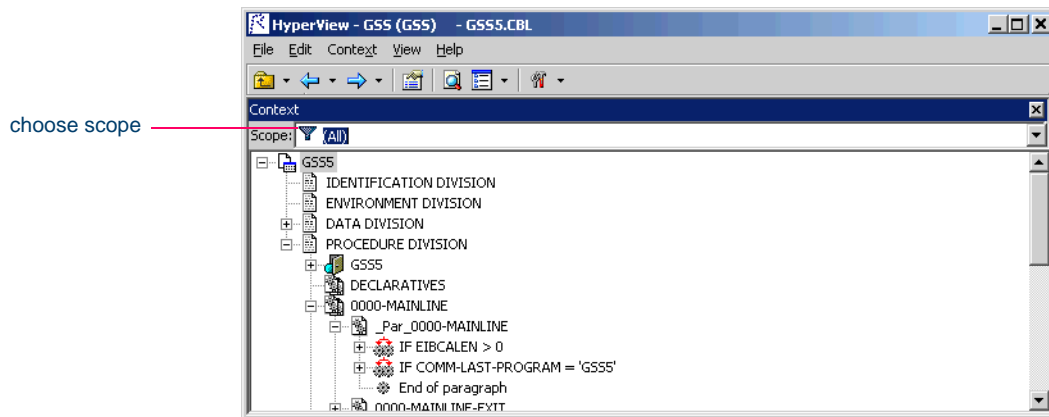
Use the Model Reference pane (Figure 1-18 on page 1-12) to view the parse tree metamodel in text and diagram form.

## Using the Context Pane

The Context pane displays the *parse tree* for the selected source file (Figure 2-3). The parse tree displays source code constructs — sections, paragraphs, statements, conditions, variables, and so forth — in hierarchical form, making it easy to locate code constructs quickly. Constructs are displayed in the order they appear in your code.

**Note:** HyperView adds artificial “owning” sections or paragraphs to the parse tree as necessary. An added, or *faked*, section or paragraph is denoted in the parse tree with a leading underscore: `_S1`, for example.

Figure 2-3 Context Pane



The *scope* of the parse tree determines the constructs and relationships it displays — all constructs, only control statements, only declarations, and so on. Scopes include the lists HyperView generates when you perform advanced searches, navigate to a related construct, or identify candidates in Clipper. Choose the scope of the parse tree in the **Scopes** drop-down on the Context pane tool bar.

Click the plus sign (+) next to a construct to expand its hierarchy. Click the minus sign (-) to collapse the hierarchy. The **Edit** menu lists every relationship type for the construct selected in the Context pane.

## 2-8 Viewing Program Source and Context *What's Next?*

Perform other tasks as you do in the Source pane:

- Navigate to related constructs as described in [“Navigating to Related Constructs” on page 2-3](#).
- Navigate to multiple occurrences of a construct as described in [“Navigating to Multiple Occurrences of a Construct” on page 2-3](#).
- Set change magnitudes as described in [“Specifying the Change Magnitude for a Source File” on page 2-4](#).

### ***What's Next?***

Now that you know how to view program source and context in HyperView, let's look at how use the HyperView advanced search facility to create sophisticated filters for construct searches. That's the subject of the next chapter.

## Performing Advanced Searches



The HyperView advanced search facility distills the parse tree metamodel into a series of prompts that help you build complex filters for construct searches. It also offers predefined search filters for business rule extraction, event injection, duplicate identification, and other tasks.

### Understanding Advanced Searches

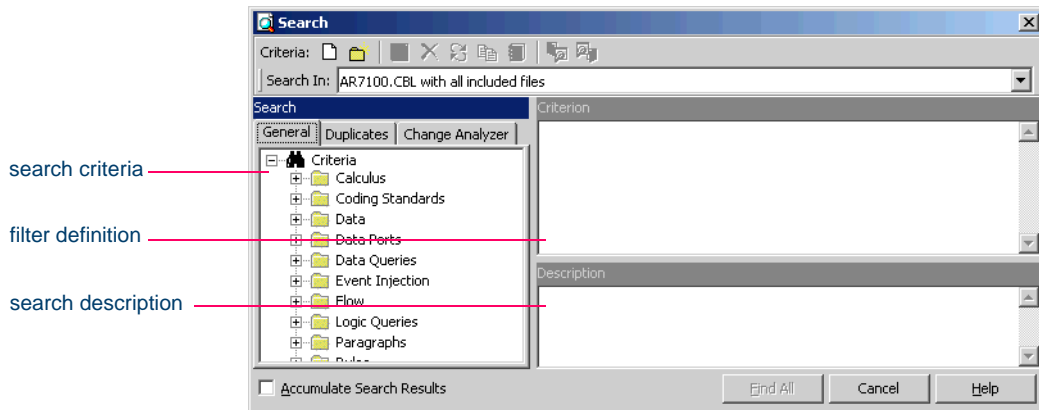
The advanced search facility uses the *attributes* and *relationships* of a construct to define the conditions you set in your search for the construct. As an example of how the facility works, let's look at how you might set up a search filter for the conditions a program uses to validate the variable EIBRESP.

#### *To set up a search filter:*

- 1 In the HyperView **Edit** menu, choose **Find**. The Search window opens. Click the **General** tab (Figure 3-1 on page 3-2).

3-2 Performing Advanced Searches  
*Understanding Advanced Searches*

Figure 3-1 Search Window — General Tab




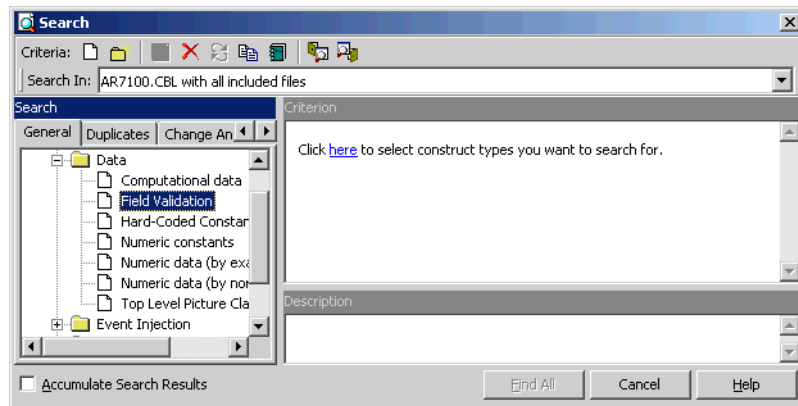
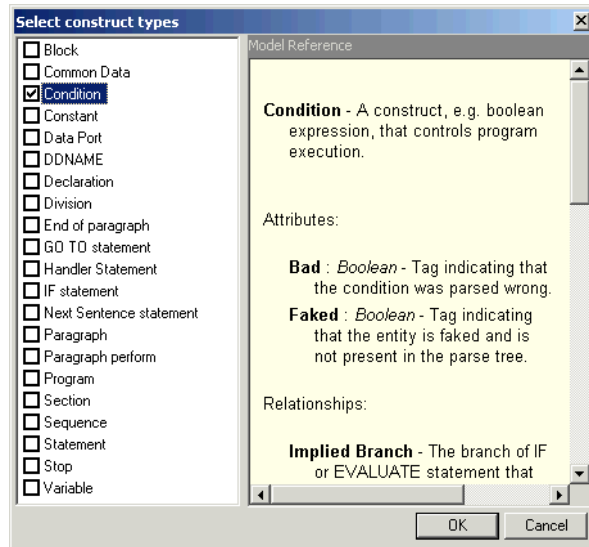
- 2 Select a folder for the new criterion in the General tab, then click the  button on the tool bar. The New Criterion dialog opens. Enter Field Validation in the text field and click **OK**. HyperView creates the Field Validation criterion in the selected folder (Figure 3-2).

Figure 3-2 Search Window — After Step 2



- 3 Click the [here](#) link in the righthand pane of the Search window. The Select Construct Types window opens (Figure 3-3 on page 3-3).

Figure 3-3 *Select Construct Types Window*

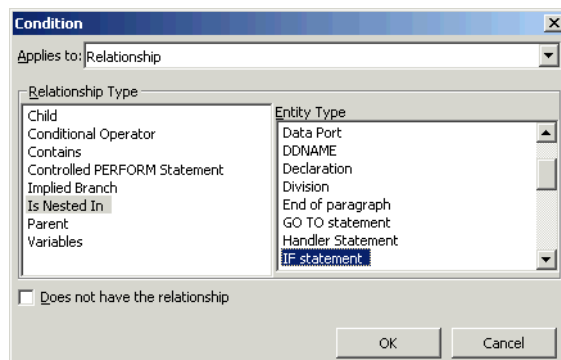


- 4 Select **Condition** in the list of constructs in the lefthand pane. In the righthand pane, review the definition of a condition in the parse tree metamodel, then click **OK**. HyperView adds the condition construct to the filter definition in the righthand pane of the Search window:

**Find All Condition**

- 5 Click the **All** link. The Condition window opens (Figure 3-4).

Figure 3-4 *Condition Window (After Editing)*



3-4 Performing Advanced Searches  
*Understanding Advanced Searches*

6 In the Condition window, choose:

- Relationship in the **Applies to** drop-down
- Is Nested In in the **Relationship Type** list box
- If Statement in the **Entity Type** list box

Click **OK**. HyperView adds the relationship to the filter definition in the righthand pane of the Search window:

Find **Condition**  
which **is nested in any** IF statement

7 Click the **any** link. The Condition window opens. In the Condition window, choose:

- Relationship in the **Applies to** drop-down
- Contains in the **Relationship Type** list box
- Paragraph perform in the **Entity Type** list box

Click **OK**. HyperView adds the relationship to the filter definition in the righthand pane of the Search window:

Find **Condition**  
which **is nested in** IF statement which **contains any**  
Paragraph perform

8 Click the **any** link. The Condition window opens. In the Condition window, choose:

- Attribute in the **Applies to** drop-down
- Caption in the **Name** list box
- Like in the **Operations** drop-down

Enter \*ERROR\* in the **Values** field and click **OK**. HyperView adds the attribute to the filter definition in the righthand pane of the Search window:

Find **Condition**  
which **is nested in** IF statement which **contains** Paragraph  
perform such that **Caption** Like "\*"ERROR\*"

**Note:** Consult the Model Reference pane for attribute values.

9 Click the **is nested in** link. In the pop-up menu, choose **And**. The Condition window opens. In the Condition window, choose:



- Relationship in the **Applies to** drop-down
- Contains in the **Relationship Type** list box
- Variable in the **Entity Type** list box

Click **OK**. HyperView adds the relationship to the filter definition in the righthand pane of the Search window:

Find [Condition](#)  
which [is nested in](#) IF statement which [contains](#) Paragraph  
perform such that [Caption](#) Like "\*\*ERROR\*\*"  
[and](#)  
which [contains any](#) Variable


- 10 Click the **any** link. The Condition window opens. In the Condition window, choose:

- Attribute in the **Applies to** drop-down
- Caption in the **Name** list box
- Like in the **Operations** drop-down

Enter \*EIB\* in the **Values** field and click **OK**. HyperView adds the attribute to the filter definition in the righthand pane of the Search window:

Find [Condition](#)  
which [is nested in](#) IF statement which [contains](#) Paragraph  
perform such that [Caption](#) Like "\*\*ERROR\*\*"  
[and](#)  
which [contains](#) Variable such that [Caption](#) Like "\*\*EIB\*\*"

- Tip:** To edit an element in the filter definition, select its link and choose **Edit** in the pop-up menu. To delete an element from the filter definition, select its link and choose **Delete** in the pop-up menu.

- 11 Enter a description of the filter definition in the Description pane.
- 12 Click the  button on the tool bar to save the filter definition. To execute the search, follow the instructions in [“Executing Advanced Searches” on page 3-8](#).

## Searching for Similarly Structured Code

Paragraphs that perform business logic often have identically or similarly structured code. The advanced search facility lets you use the *signature* of a paragraph to search for instances of similarly structured code. A signature identifies the structure of a paragraph, as in the following example:

```
BLOCK
IF
    THEN
        Perform_PROC
```

**Note:** This feature is not available for PL/I and Natural programs.

### To search for similarly structured code:



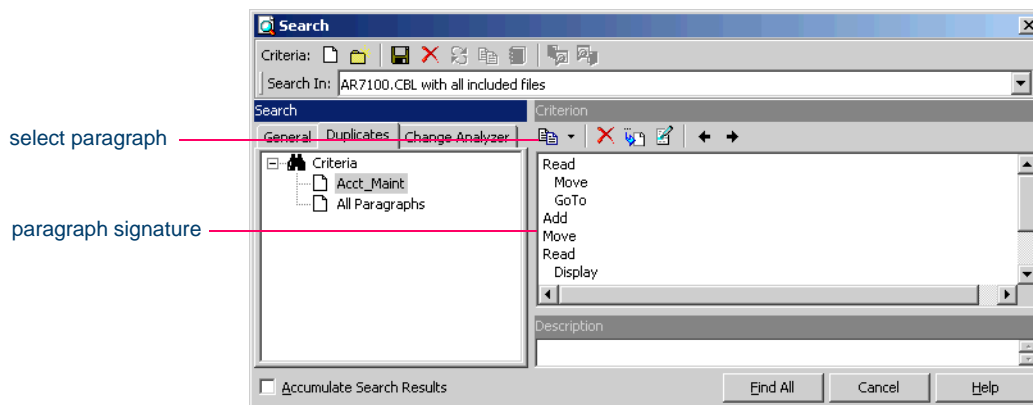



- 1 In the HyperView **Edit** menu, choose **Find**. The Search window opens. Click the **Duplicates** tab (Figure 3-5).
- 2 Define a new search criterion, as described in [“Understanding Advanced Searches”](#) on page 3-1.
- 3 Click the  button next to the  button on the tool bar and choose the paragraph whose signature you want to use from the drop-down menu of available paragraphs. HyperView displays the signature of the selected paragraph in the Search window (Figure 3-5).

Figure 3-5 Search Window — Duplicates Tab



**Tip:** Use the  and  buttons on the tool bar to adjust the nesting level of a line in the signature. Select a line in the signature and click the  button on the tool bar to delete the line.



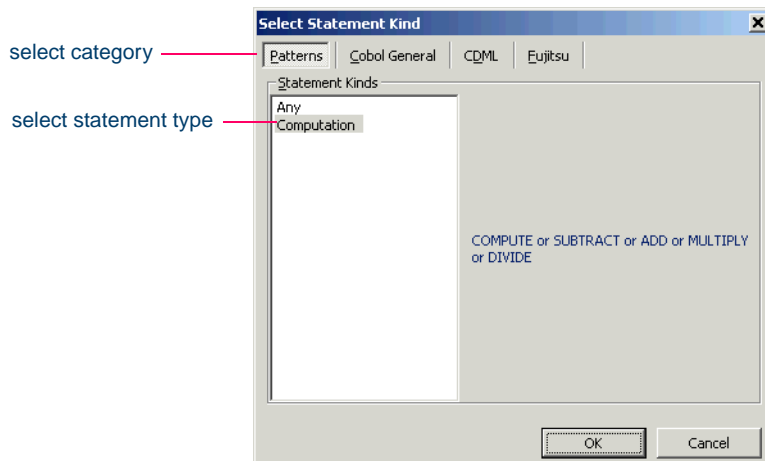

- 4 Insert statements in a signature by selecting a line in the signature and clicking the  button on the tool bar. HyperView inserts an ellipsis (...) above the selected line. Select the ellipsis and click the  button on the tool bar. The Select Statement Kind window opens (Figure 3-6).

Figure 3-6 *Select Statement Kind Window — Duplicates Tab*



- 5 Click the button for the category of statement you want to insert. In the Statement Kinds pane, choose the statement you want to insert and click **OK**. HyperView replaces the ellipsis with the selected statement.

**Tip:** You can also replace a line in the signature with a statement.

- 6 Click the  button on the tool bar to save the search criterion. To execute the search, follow the instructions in [“Executing Advanced Searches” on page 3-8](#).

## **Executing Advanced Searches**

You can execute advanced searches in standalone mode — against the program or construct selected in the Source or Context panes. You can also execute advanced searches in Clipper — against the current project. For Clipper usage, see [Chapter 4, “Staging Program Analysis with Clipper.”](#)

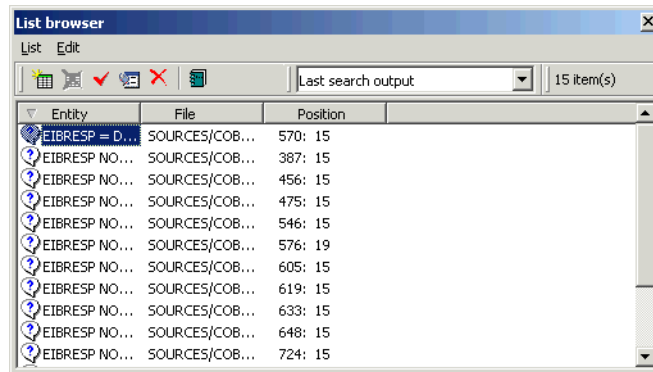
**Note:** The Change Analyzer tab opens a scoped version of the advanced search facility for Change Analyzer. For Change Analyzer usage, see *Analyzing Projects* in the ATW document set.

### **To execute an advanced search in standalone mode:**

- 1 In the HyperView **Edit** menu, choose **Find**. The Search window opens. Click the **General** tab (Figure 3-1 on page 3-2).
- 2 In the **Search In** drop-down on the tool bar, choose:
  - **Source file name with All Included Files** if you want to execute the search against the selected source file and any included files.
  - **Source file name Only** if you want to execute the search against the selected source file only.
  - **Selected Construct - Construct** if you want to execute the search against the construct selected in the Source or Context panes.
- 3 Choose the criterion for the search you want to execute from the list of criteria in the lefthand pane.
- 4 Select **Accumulate Search Results** if you want the search results to be added to the results of the last search. Otherwise, the new results overwrite the previous results.
- 5 Click **Find All**. HyperView displays a dialog box with the results of the search operation. Click **OK**. The dialog and Search window are dismissed and the search results are displayed in the List Browser window (Figure 3-7 on page 3-9).

**Note:** For List Browser usage, see [“Using the List Browser Window”](#) on page 1-18.


Figure 3-7 *List Browser Window with Search Results*





## Defining Advanced Search Criteria


Define advanced search criteria as described in [“Understanding Advanced Searches” on page 3-1](#) and [“Searching for Similarly Structured Code” on page 3-6](#). You can use wildcard patterns allowed in LIKE statements by Visual Basic for Applications (VBA).

**Editing a Search Criterion** To edit a search criterion, select the criterion, then select the link for the element you want to change in the filter definition. Choose **Edit** in the pop-up menu to edit the element, or **Delete** to delete the element.

**Tip:** To restore the original definition of a predefined search criterion, select the criterion and click the  button on the tool bar.


**Copying a Search Criterion** To copy a search criterion, select the criterion and click the  button on the tool bar. The New Criterion dialog opens. Enter the name of the new criterion in the text field and click **OK**.


**Saving a Search Criterion** To save a search criterion, select the criterion and click the  button on the tool bar.


**Creating Folders for Search Criteria** To create a folder for search criteria, click the  button on the tool bar. The New Folder dialog opens.



### 3-10 Performing Advanced Searches *What's Next?*


Enter the name of the new folder in the text field and click **OK**. The new folder appears in alphabetical order in the tree in the lefthand pane of the window. Drag-and-drop search criteria to move them to the folder. You can create folders within folders.

To copy a folder and all its contents, select the folder and click the  button on the tool bar. The New Folder dialog opens. In the text field, enter text to be prepended to the folder name and to the names of each of its subfolders and search criteria, and click **OK**.

To modify a folder name, click in the name area for the folder to make the name editable, enter the new name, and press Enter. To delete a folder, select the folder and click the  button on the tool bar. You are prompted to confirm the deletion. Click **OK**.

**Saving a Search Criterion as HTML** To save a search criterion in an HTML file, select the criterion and click the  button on the tool bar. The Save Criteria dialog opens, where you can specify the name and folder for the HTML file.

**Exporting and Importing a Search Criterion** To export a search criterion to an XML file, select the criterion and click the  button on the tool bar. The Export Criteria dialog opens, where you can specify the name and folder for the XML file. To import a search criterion, click the  button on the tool bar. The Import Criteria dialog opens, where you can select the criterion you want to import.

**Deleting a Search Criterion** To delete a search criterion, select the criterion and click the  button on the tool bar. You are prompted to confirm the deletion. Click **OK**.

**Note:** You cannot delete a predefined search criterion.

## **What's Next?**

That's all you need to know to perform an advanced search in HyperView. Now let's look at how you might use the search results to generate lists of candidates for business rule extraction, event injection, and impact analysis in Clipper.

## *Staging Program Analysis with Clipper*



**C**lipper lets you create lists of candidates for business rule extraction, event injection, impact analysis, and other tasks. Each list captures the results of a different stage of your analysis and serves as input for subsequent tasks. A system analyst, for example, might use Clipper to create a list of programs impacted by a change request, then create a project that contains only the source files for those programs.

### *Getting Started in Clipper*

Let's look at a sample use of Clipper that shows you how to create a set of business rules. The sample should help you get oriented in the tool. Usage for other tasks is similar.

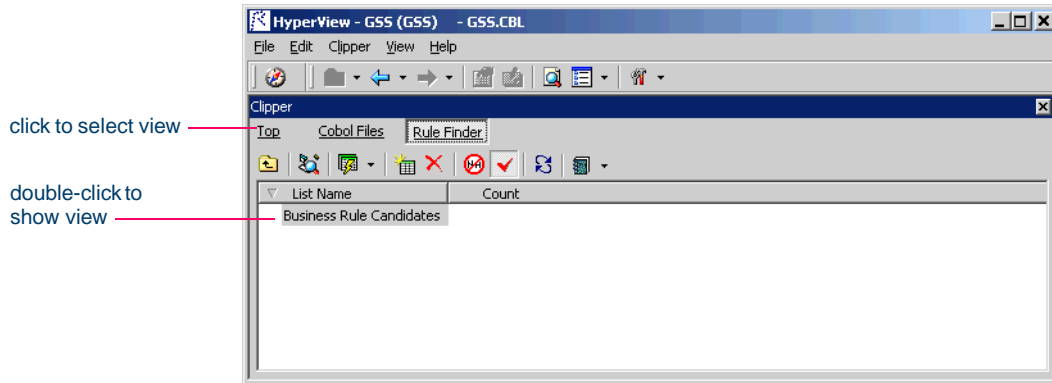
#### *To create business rules with Clipper:*


- 1 In the HyperView **View** menu, choose **Clipper**. The Clipper window opens. Click the **Top** view link.
- 2 The Top view opens. In the Model Name column, double-click Cobol Files.

4-2 Staging Program Analysis with Clipper  
*Getting Started in Clipper*

- 3 The Cobol Files view opens. In the Category Name column, double-click Rule Finder. The Rule Finder view opens (Figure 4-1).

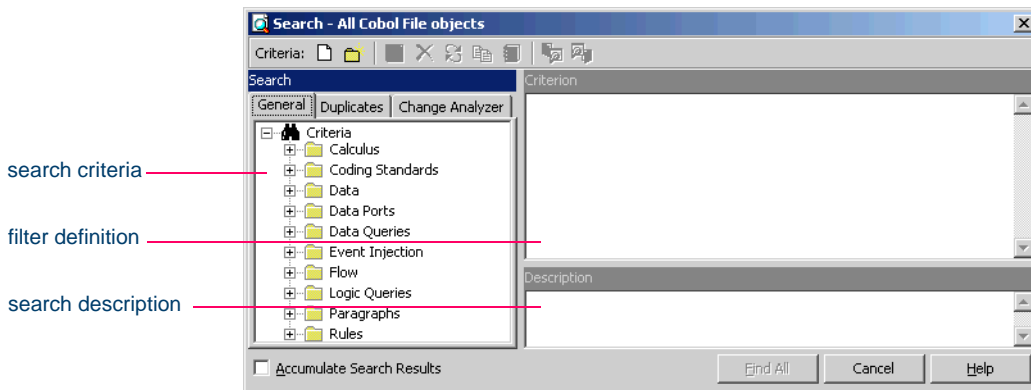
Figure 4-1 *Clipper Pane — Rule Finder View*



- 4 Click the  button on the tool bar. The Search window opens (Figure 4-2).

**Tip:** Notice that the Search window for Clipper does not have the **Search In** drop-down on the tool bar. That's because all Clipper searches are against the entire current project.

Figure 4-2 *Search Window for Clipper*

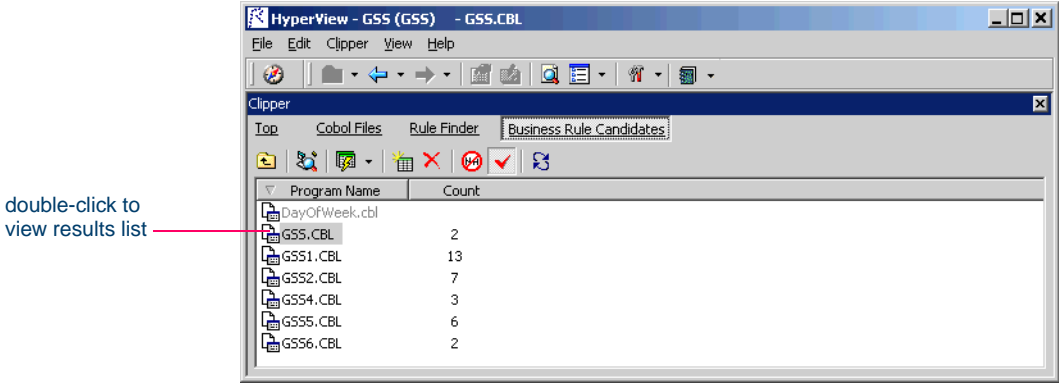





- 5 Click the predefined Rule Finder - Data Ports criterion in the Rules folder in the lefthand pane of the Search window and click **Find All**. The Search window is dismissed and the results of the search are displayed in the Clipper Business Rule Candidates view (Figure 4-3).

**Tip:** You can modify the predefined criterion or create your own criterion as described in [Chapter 3, “Performing Advanced Searches.”](#)

Figure 4-3 *Clipper Pane — Business Rule Candidates View*



**Tip:** Select a construct in the Source or Context panes and click the  button on the tool bar to add the construct to the list of candidates manually.



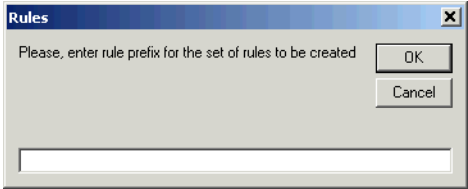
- 6 Click the  button next to the  button on the tool bar and choose **Create Rules** in the drop-down menu. The Rules dialog opens (Figure 4-4).

Figure 4-4 *Rules Dialog*

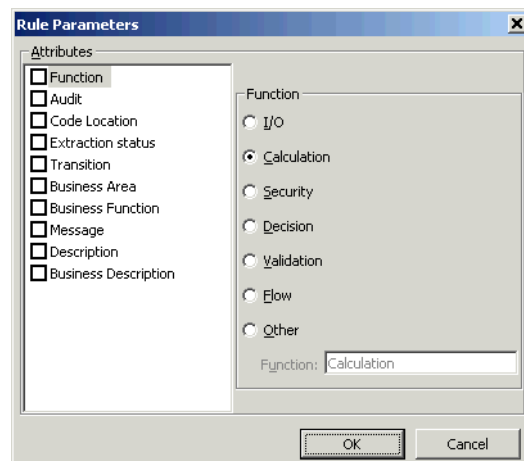


- 7 In the text field, enter a descriptive prefix for the set of rules Clipper will generate. If you enter Data Port, for example, the business rules in the set will be named Data Port1, Data Port2, Data Port3, and so on. You can change the names in the HyperView Rules pane, as described in [“Renaming Rules” on page 7-13](#). Click **OK**.

**Tip:** You can create a rule for a single construct returned by the search as described in [“Generating Business Rules” on page 4-7](#).

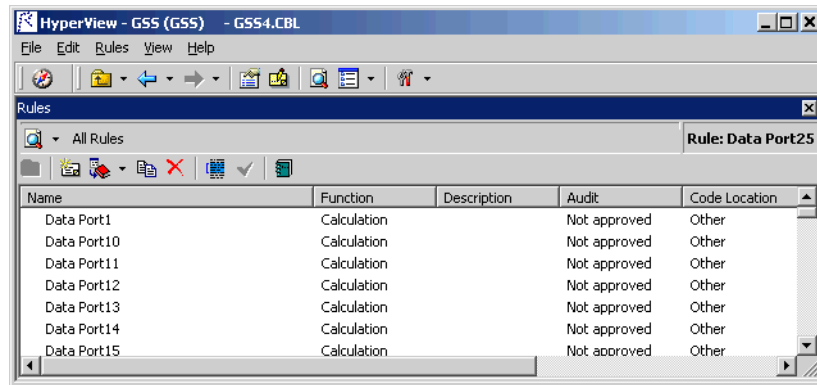
- 8 The Rule Parameters window opens (Figure 4-5). Use this window to specify the attributes for every rule in the set of rules Clipper will generate. Select an attribute in the lefthand pane, then specify its value in the righthand pane. Place a check mark next to each attribute you want to set. For descriptions of the attributes, see [“Editing Rule Properties” on page 7-14](#).

Figure 4-5 *Rule Parameters Window*



- 9 HyperView displays an informational message that tells you how many business rules it created. Click **OK**. You can view and edit the new business rules in the HyperView Rules pane (Figure 4-6 on page 4-5).


Figure 4-6 *Rules Pane*



## Using the Clipper Pane

The Clipper pane consists of a hierarchy of views that let you specify the candidate lists you want to display:

- The *Top* view displays the types of source files you can create lists for.
- The *Category* view displays the categories of lists you can create for the selected source file type.
- The *List* view displays the lists in the selected category.
- The *File* view displays the source files of the selected type in the current project.
- The *Construct* view displays each construct in the list for the selected source file.


Each view gives you access to the next-level view in the hierarchy. See Figure 4-1 on page 4-2 for basic navigation details. Click the  button on the tool bar to navigate to the parent of the current view.


**Sorting Entries** Click a column heading in a view to sort the view entries by that column.

**Sizing Columns** Grab-and-drag the border of a column heading to increase or decrease the width of the column.

### Working with Categories


The Category view displays the categories of lists you can generate for a source file type. Generally speaking, categories correspond to tasks you perform with a list — event injection or business rule extraction for a Cobol source file, for example. HyperView supplies default categories for each source type. You can create your own categories as necessary.


**Creating Categories** In the Category view, click the  button on the tool bar to create a new category for the selected source type. The New Category dialog opens, where you can specify the name of the new category.


**Deleting Categories** In the Category view, select a category and click the  button on the tool bar to delete the category. You cannot delete a predefined category.

### Working with Lists


The List view displays the lists in a category. HyperView supplies default lists for each category — calls and ports, for example, for the event injection category. You can create your own lists as necessary.


**Creating Lists** In the List view, click the  button on the tool bar to create a new list for the selected category. The New List dialog opens, where you can specify the name of the new list.


**Deleting Lists** In the List view, select a list and click the  button on the tool bar to delete the list.


**Generating Lists Automatically** In the List view, select a list and click the  button on the tool bar to open the Search window, where you can use the advanced search facility to generate list constructs automatically. See steps [4-5](#) on pages 4-2-4-3.


**Tip:** Select **Accumulate Search Results** in the Search window if you want the search results to be added to the results of the last search. Otherwise, the new results overwrite the previous results.

**Adding Constructs to a List Manually** In the File view, select a source file, then choose a construct in the Source or Context panes. Click the  button on the tool bar to add the construct to the list of candidates for the selected file.



**Deleting Constructs from a List** In the Construct view, select a construct and click the  button on the tool bar to delete the construct from the selected list.

**Hiding Files with Empty Lists** In the File view, click the  button on the tool bar to hide source files with empty lists. Displayed source files with empty lists are grayed out.

**Highlighting List Constructs in Source Code** In the File view, select a source file and click the  button on the tool bar to place a check mark in the Source pane display for each item in the list of candidates for the selected file.



**Refreshing Lists** In the List view, select a list and click the  button on the tool bar to refresh the list.

### Generating Business Rules

To generate business rules for all the constructs in a list, select the list in the List view for the business rule category, click the  button next to the  button on the tool bar, and choose **Create Rules** in the drop-down menu. Follow the instructions in steps [7-9](#) on page 4-4 to generate business rules.

**Tip:** As long as the Rules pane is open, you can generate a business rule for a single construct by selecting the construct in the Construct view and choosing **Create Rule** in the **Rules** menu.

### Showing Impacts

To show impacts for all the constructs in a list, select the list in the List view for the impact report category, click the  button next to the  button on the tool bar, and choose **Show Impacts** in the drop-down menu. HyperView displays the impacts for the constructs in the list in the

Impact Report window. For usage, see [Chapter 5, “Analyzing Impact Traces.”](#)

## Creating Projects in Clipper

You can create a project directly in Clipper from the results of your analysis. The project contains only source files with non-empty lists of the selected type. You can also add source files to an existing project.

### To create or add to a project:





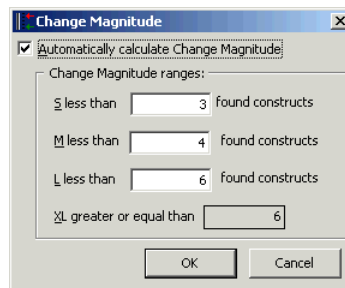
- 1A To create a project that contains all source files with non-empty lists, double-click the list in the List view, click the  button next to the  button on the tool bar, and choose **Create Project** in the drop-down menu. The New Project dialog opens. Enter the name of the new project in the text field and click **OK**. The Change Magnitude window opens (Figure 4-7).
- 1B To add to an existing project all source files with non-empty lists, double-click the list in the List view, click the  button next to the  button on the tool bar, and choose **Add to Project Name project** in the drop-down menu. The Change Magnitude window opens (Figure 4-7).

Figure 4-7 *Change Magnitude Window*



- 2 In the Change Magnitude window, select the **Automatically calculate change magnitude** check box if you want Clipper to set change magnitudes for the listed source files based on the ranges specified

in the fields below the check box. The Clipper settings will override any existing change magnitudes for the files.



The fields below the check box contain default ranges for the available values: Small, Medium, Large, and Extra Large. Compare the number of constructs in each source file listed in the File view with the default ranges, then modify the ranges as necessary.

If you want Cobol source files with 6 to 10 constructs to have a Large change magnitude, for example, set the range for Medium to less than 6 and the range for Large to less than 11. When you are satisfied with your choices, click **OK**.

For more information on change magnitudes, see [“Specifying the Change Magnitude for a Source File” on page 2-4](#).

## ***Generating Clipper Reports***

Use Clipper reports to show the distribution of list instances across a project and view the details for each instance. The following reports are available:

- For each list in a category, the *Metrics Report* shows the number of list items in each program in the project. The report for the default event injection category, for example, shows the number of calls and ports. To generate a Metrics Report, select a category, then click the  button on the tool bar and choose the report type from the pull-down menu.
- For each instance in a list, the *Details Report* shows the program in which it occurs, its location in the program, and other attributes. You can customize the report to include any HyperView attribute related to the instance. To generate a Details Report, select a list, then click the  button on the tool bar and choose the report type from the pull-down menu. In the Select Attributes dialog, choose the attributes you want to view in the report.

#### 4-10 Staging Program Analysis with Clipper *What's Next?*

##### ***Printing Reports***

Click **Page Setup** in a report window to set up the page for a printed report. Click **Print** to print the report. The Print dialog opens, where you can set options for the print job. Click **OK**.

##### ***Exporting Reports***

You can export Clipper reports to a variety of standard formats. Click **Save** in a report window to export a report to HTML, Excel, RTF, Word, or formatted text. A Save As dialog opens, where you can specify the name, location, and file type of the report.

**Note:** Excel is usually the best choice if you need to manipulate the report or perform computations.

##### ***What's Next?***

Now that you know how to search programs in HyperView and stage program analysis in Clipper, you're ready to drill down deeper into programs. The next chapter looks at how you analyze impacts in HyperView.



## Analyzing Impact Traces



An *impact trace* describes how data items interact with each other in a program. HyperView offers two tools for analyzing impact traces. Which one you use depends on whether you are interested in data flow analysis or traditional impact analysis:

- The Impact pane displays a hierarchical view and diagram of an impact trace. Use these views to analyze the data flows in a project.
- The Impact Report window displays a hierarchy of views that let you narrow an impact trace based on the depth of the affected item or the program that contains the affected item. Use these views to perform traditional impact analysis — to hone in on the data items in a project that may be impacted by changing another data item’s definition or usage.

**Note:** Projects must have been verified with the **Enable Data Element Flow** and **Enable Impact Report** options set in the Verification tab of the Project Options window. JCL files must have been verified according to the instructions in *Preparing Projects* in the ATW document set.

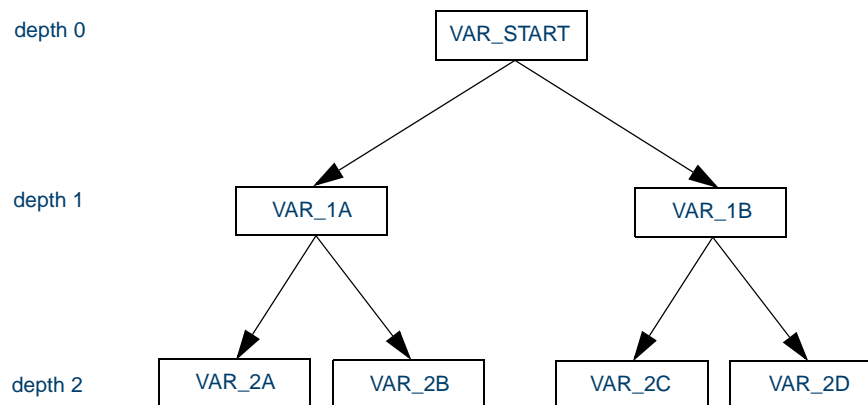
## Using the Impact Report Tool

The Impact Report tool lets you answer the kinds of “What if?” questions posed in the recent past by the industry-wide changes for Y2K, Zip+4, and the Euro dollar: “What if I change the type of this variable, or the length of this field?”

### Understanding Impact Report Depths

An impact report typically shows the flow of data from a *startup item* to every data item that would be affected by its modification. The report is organized in hierarchical form according to the *depth* of the affected item — items that would be directly affected by a change in the startup item are at depth 1, items that would be affected by a change in depth 1 items are at depth 2, and so on (Figure 5-1).

Figure 5-1 *Impact Report for Affected Data Items*



*An impact report is organized in hierarchical form according to the depth of affected items.*

An impact trace can also show the flow of data *to* a startup item from the data items that affect it — the *causes* of the startup item rather than its *consequences*. For complex tasks, such as changing the precision of a floating point variable, you can combine the two modes and show causes and consequences in the same trace.

### Generating an Impact Trace in the Impact Report Tool

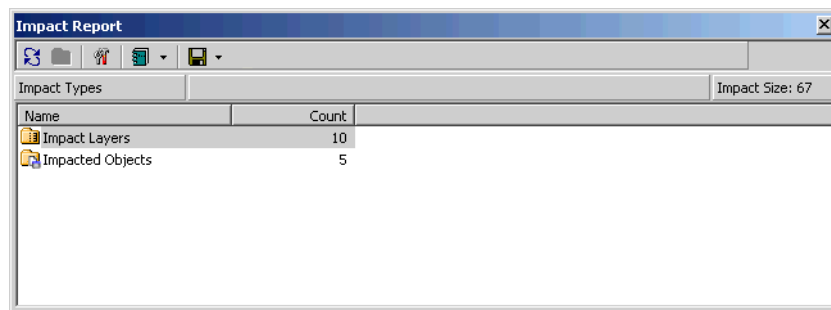
You can select the startup item for an impact trace manually in the Source or Context panes, or create a list of startup items in Clipper.

**Note:** For Clipper usage, see [Chapter 4, “Staging Program Analysis with Clipper.”](#)

#### To generate an impact trace in the Impact Report tool:

- 1 In the Source or Context pane, select the construct that contains the startup item or items and choose **Impact Report** in the **View** menu. The Impact Report window opens (Figure 5-2).

Figure 5-2 *Impact Report Window*



- 2A To navigate to an impact trace based on the depth of the affected item (see [“Using the Impact Report Tool”](#) on page 5-2), double-click the Impact Layers folder. The Impact Layers view opens. Double-click the appropriate depth. The Impacted Data Items view opens (Figure 5-3). Continue to [step 3](#).
- 2B To navigate to an impact trace based on the object that contains the affected item, double-click the Impacted Objects folder. The Impacted Objects view opens. Double-click the appropriate object. The Impacted Data Items view opens (Figure 5-3 on page 5-4). Continue to [step 3](#).

5-4 Analyzing Impact Traces  
Using the Impact Report Tool

Figure 5-3 Impacted Data Items View

Data Item Decl	Program	Impact Factor	File	Ln:Col
COMM-NEXT-BROWSE-KEY	GSS5	4	SOURCES\COBOL\GSS5.CBL	56: 12
DFHCOMMAREA	GSS4	1	SOURCES\COBOL\GSS4.CBL	197: 8
DFHCOMMAREA	GSS5	1	SOURCES\COBOL\GSS5.CBL	329: 8
GSS10050	GSS5	2	SOURCES\COBOL\GSS5.CBL	212: 8
TCB-USER-ID	GSS5	1	SOURCES\COPYBOOK\ON...	25: 12
WS05-COMMON-AREA	GSS	1	SOURCES\COBOL\GSS.CBL	37: 8
WS05-COMMON-AREA	GSS4	1	SOURCES\COBOL\GSS4.CBL	50: 8
WS10-SCR-ADJ-NO	GSS5	3	SOURCES\COBOL\GSS5.CBL	106: 20
WS10-SCR-ATTRB	GSS5	8	SOURCES\COBOL\GSS5.CBL	101: 16
WS10-SCREEN-LINES	GSS5	25	SOURCES\COBOL\GSS5.CBL	100: 12
WS10-SCREEN-LINES-TABLE	GSS5	2	SOURCES\COBOL\GSS5.CBL	99: 8

- 3 The Impacted Data Items view lists the declaration of each affected data item and the number of impacts on the data item (its *impact factor*). Double-click the declaration for the item whose trace you want to display. The Impacted Instances view opens (Figure 5-4).

Figure 5-4 Impacted Instances View

Data Item	Offset:Size	Program	File	Ln:Col
COMM-NEXT-BROWSE-KEY	397:10	GSS5	SOURCES\COBOL\GSS5.CBL	793: 47
COMM-NEXT-BROWSE-KEY	397:10	GSS5	SOURCES\COBOL\GSS5.CBL	803: 47
COMM-NEXT-BROWSE-KEY	397:10	GSS5	SOURCES\COBOL\GSS5.CBL	813: 47
COMM-NEXT-BROWSE-KEY	397:10	GSS5	SOURCES\COBOL\GSS5.CBL	823: 47

- 4 The Impacted Instances view lists each impact on the affected data item. Double-click the impact whose trace you want to display. The Impact Sequence view opens (Figure 5-5 on page 5-5).

**Tip:** Although a given use of a data item may be impacted multiple times, only one trace is shown.

Figure 5-5 *Impact Sequence View*

The screenshot shows a window titled "Impact Report" with a menu bar and a toolbar. Below the toolbar is a breadcrumb-style path: "Impact Sequence > Impact Layers - Depth > 5 - Dataitem COMM-NEXT-BROWSE-KEY at [SOURCES\COD] 6 item(s)". The main area contains a table with the following data:

Depth	Item	Statement	Relationship	Offset:Size	Object Name	File	Ln:Col
0	COMM-NEXT-BROW...	MOVE CO...	start	224:512	GSS5	SOURCES\C...	435: 17
➔ 1	FA01-KEY	MOVE CO...	move	4248:10	GSS5	SOURCES\C...	435: 43
← 2	COMM-NEXT-BROW...	MOVE CO...	move	397:10	GSS5	SOURCES\C...	435: 17
← 3	WS05-COMMON-AR...	MOVE DF...	used	397:10	GSS5	SOURCES\C...	339: 36
➔ 4	WS05-COMMON-AR...	GSS5.1.8...	used	397:10	GSS5	SOURCES\C...	861: 27
← 5	COMM-NEXT-BROW...	MOVE CO...	used	397:10	GSS5	SOURCES\C...	793: 47

- 5 The Impact Sequence view lists the data items at each depth in the impact trace, the statement in which each data item is impacted, and the relationship of the data item to the previous item.


**Note:** For a description of the relationships, see Table 5-1 on page 5-8.


- A ➔ symbol next to an item means that it is a consequence of the previous item. A ← symbol means that it is a cause of the previous item.
- A ↔ symbol means that the item is both a cause and a consequence.

### **Using the Impact Report Window**

The Impact Report window lets you navigate to an impact trace based on the depth of the affected item or the object that contains the affected item (see steps [2A-2B](#) on page 5-3). Each navigation path consists of a hierarchy of views that let you specify the data impacts you want to trace:

- The *Impact Layers* view lists the depths of affected items. The *Impacted Objects* view lists the objects that contain affected items and the number of impacts on the object.
- The *Impacted Data Items* view (Figure 5-3 on page 5-4) lists the declaration of each affected data item and the number of impacts on the data item.

- The *Impacted Instances* view (Figure 5-4 on page 5-4) lists each impact on the affected data item. Click the  button on the tool bar to save the instances to a list you can view in the List Browser.
- The *Impact Sequence* view (Figure 5-5 on page 5-5) lists the data items at each depth in the impact trace, the statement in which each data item is impacted, and the relationship of the data item to the previous item. For more information, see [step 5 on page 5-5](#).


Double-click an item in a view to navigate to the next-level view in the hierarchy. Click the  button on the tool bar to navigate to the parent of the current view.

**Sorting Entries** Click a column heading in a view to sort the view entries by that column.


**Sizing Columns** Grab-and-drag the border of a column heading to increase or decrease the width of the column.

### Setting Impact Report Options

Use the Impact Report Options window to hide causes or consequences, control the depth of the impact trace, filter out unwanted intra- or inter-program relationships, and limit the number of startup items. As a rule, limiting the scope of the report improves performance.

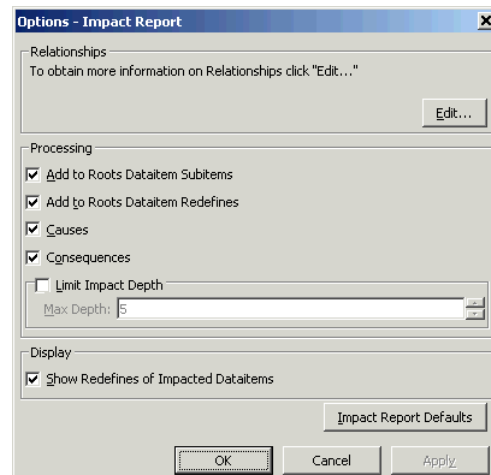
**Tip:** After setting impact report options, click the  button on the tool bar to refresh the impact report.

#### To set Impact Report options:

- 1 Click the  button on the tool bar. The Impact Report Options window opens (Figure 5-6 on page 5-7).
- 2 In the Processing pane, choose any combination of:
  - **Add to Roots Data Item Subitems** if you want the report to include as startup items, data items in nested declarations of selected startup items.
  - **Add to Roots Data Item Redefines** if you want the report to include as startup items, data items that redefine selected startup items.

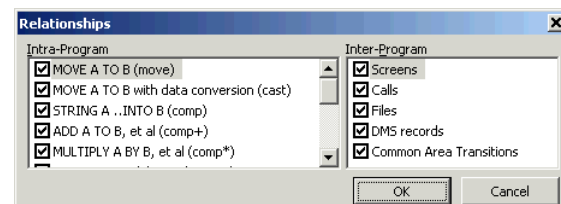
- **Causes** if you want the report to include data items that cause a previous item in the trace.
- **Consequences** if you want the report to include data items that are a consequence of a previous item in the trace.
- **Limit Impact Depth** if you want the report to include data items only up to a specified depth. Specify the depth in the **Max Depth** combo box.

Figure 5-6 *Impact Report Options Window*



- 3 In the Display pane, choose **Show Redefines of Impacted Data Items** if you want the report to include as affected items data items that redefine affected items.
- 4 In the Relationships pane, click the **Edit** button. The Relationships window opens (Figure 5-7).

Figure 5-7 *Impact Report Options Relationships Window*



**5-8** Analyzing Impact Traces  
*Using the Impact Report Tool*

- 5** In the Intraprogram pane, place a check mark next to each intraprogram impact relationship you want the report to include.
- 6** In the Interprogram pane, place a check mark next to each interprogram impact relationship you want the report to include.

**Note:** For a description of intraprogram and interprogram relationships, see Table 5-1 on page 5-8

- 7** Click **OK** to dismiss the Relationships window and return to the main Impact Report Options window.

**Tip:** You can restore the default option settings by clicking **Impact Report Defaults**, then choosing **Restore Impact Defaults** in the drop-down menu. Choose **Save To** in the drop-down menu to save the option settings to a file. Choose **Load From** in the menu to restore the option settings from a file.

- 8A** Click **Apply** if you want to save your settings without dismissing the Options window.
- 8B** Click **OK** if you want to save your settings and dismiss the Options window.

Table 5-1 *Impact Report Relationships*









Relationship	Definition	Type	Icon	Description
calls	N/A	interprogram	N/A	Denotes that a parameter is passed in a call to another program.
cast	MOVE A TO B with data conversion	intraprogram		Denotes that a data item is moved to a data item of a different type.



Table 5-1 *Impact Report Relationships* (continued)

Relationship	Definition	Type	Icon	Description
common area transitions	N/A	interprogram	N/A	For Unisys Cobol, denotes that a common-storage data area item is passed in a call to another program. The Perform Unisys Common-Storage Area Analysis verification option must be set.
comp	STRING A ... INTO B	intraprogram		Denotes arbitrary computation. The result is produced by applying complex rules to the argument, such as STRING.
comp+	ADD A TO B	intraprogram		Denotes addition-like operations: ADD, SUBTRACT, and corresponding parts of COMPUTE.
comp*	MULTIPLY A BY B	intraprogram		Denotes multiplication-like operations: MULTIPLY, DIVIDE, and corresponding parts of COMPUTE.
comp@	MOVE ARRAY(IDX) TO A	intraprogram		Denotes operations with array elements.
controlled	IF ... A THEN MOVE... TO B	intraprogram		Denotes control dependence: B depends on A, since B is assigned with a value under the control of a condition that depends on A.
cond	IF A = B ...	intraprogram		Denotes comparison of data items with a symmetric relationship.
cond*	IF A * X = B ...	intraprogram		Denotes comparison of a multiple of a data item with another data item.

**5-10** Analyzing Impact Traces  
*Using the Impact Report Tool*

Table 5-1 *Impact Report Relationships* (continued)

Relationship	Definition	Type	Icon	Description
const.cond	IF A = 1 ...	intraprogram		Denotes comparison of a data item with a constant.
const.move	MOVE 1 TO B	intraprogram		Denotes that a constant is moved into a data item.
const.comp	ADD 1 TO B	intraprogram		Denotes arithmetic operations with constants.
const.init	03 A ... VALUE 1	intraprogram		Denotes a data item initialized by a constant.
DMS records	N/A	interprogram	N/A	For Unisys Cobol, denotes communication via Unisys DMS database records.
files	N/A	interprogram	N/A	Denotes communication via files. Traced only when corresponding JCL, ECL, FCT, or CSD files are verified.
move	MOVE A TO B	intraprogram		Denotes that a data item is moved to a data item of the same type.
screens	N/A	interprogram	N/A	Denotes that data is sent to a screen by one program and received in a screen by another.
start	N/A	intraprogram	N/A	Denotes a startup item. (Not shown in Options window.)
used	MOVE ... TO A ... MOVE A TO ...	intraprogram		Denotes that a value assigned in a statement is used as an argument in another statement. (Not shown in Options window.)

**Note:** Icons appear in the Impact pane only.

## Creating Projects in the Impact Report Tool

You can create a project directly in the Impact Report tool from the results of your analysis. The project contains only source files that contain affected data items. You can also add source files to an existing project.

### To create or add to a project:





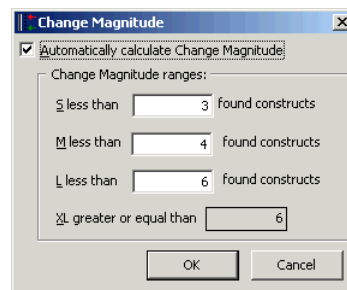
- 1A To create a project that contains all source files with affected data items, double-click the Impacted Objects folder, click the  button next to the  button on the tool bar, and choose **Create Project** in the drop-down menu. The New Project dialog opens. Enter the name of the new project in the text field and click **OK**. The Change Magnitude window opens (Figure 5-8).
- 1B To add to an existing project all source files with affected data items, double-click the Impacted Objects folder, click the  button next to the  button on the tool bar, and choose **Add to Project Name project** in the drop-down menu. The Change Magnitude window opens (Figure 5-8).

Figure 5-8 Change Magnitude Window



- 2 In the Change Magnitude window, select the **Automatically calculate change magnitude** check box if you want Impact Report to set change magnitudes for the listed objects based on the ranges specified in the fields below the check box. The Impact Report settings will override any existing change magnitudes for the objects.

The fields below the check box contain default ranges for the available values: Small, Medium, Large, and Extra Large. Compare

**5-12** Analyzing Impact Traces  
*Using the Impact Report Tool*

the number of affected items in each object listed in the Impacted Objects view with the default ranges, then modify the ranges as necessary.

If you want Cobol source files with 6 to 10 constructs to have a Large change magnitude, for example, set the range for Medium to less than 6 and the range for Large to less than 11. When you are satisfied with your choices, click **OK**.

For more information on change magnitudes, see [“Specifying the Change Magnitude for a Source File” on page 2-4](#).

### **Exporting Impact Reports**

Use an HTML report to control the level of detail in the impact reports you export. You can export a simple *impact layers* report to HTML, Excel, RTF, Word, or formatted text.

#### **To export an impact report:**


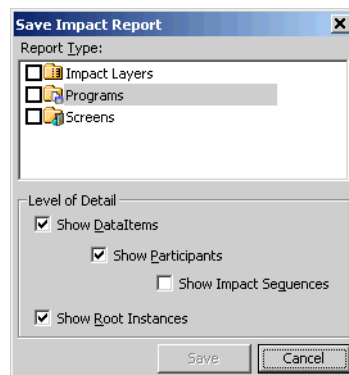


- 1 Click the  button on the tool bar. The Save Impact Report window opens (Figure 5-9).

Figure 5-9 *Save Impact Report Window*



- 2 In the Report Type pane, choose any combination of **Impact Layers** and the available check boxes for affected objects.
- 3 In the Level of Detail pane, choose any combination of:

- **Show Data Items** if you want the report to include affected data items. If you choose this option, choose **Show Participants** if you want the report to include each impact instance for the affected data items. If you choose **Show Participants**, choose **Show Impact Sequences** if you want the report to include the relationships between impact instances.
  - **Show Root Instances** if you want the report to include startup items.
- 4 Click **OK** to dismiss the Save Impact Report window. A Save As dialog opens, where you can specify the name and folder for the report.

**Tip:** Click the  button next to the  button on the tool bar and choose **Save Page As** in the drop-down menu to export an impact layers report to HTML, Excel, RTF, Word, or formatted text.




## Using the Impact Pane

The Impact pane lets you analyze the data flows in a project. If you are extracting business rules, for example, you might want to view an impact trace to make sure that a segment encapsulates all of the business logic for a candidate rule. Or you might want to use a scoped trace to help locate candidates that exhibit a particular type of relationship.

### Generating an Impact Trace in the Impact Pane

Follow the steps below to generate an impact trace in the impact pane.

#### **To generate an impact trace in the Impact pane:**

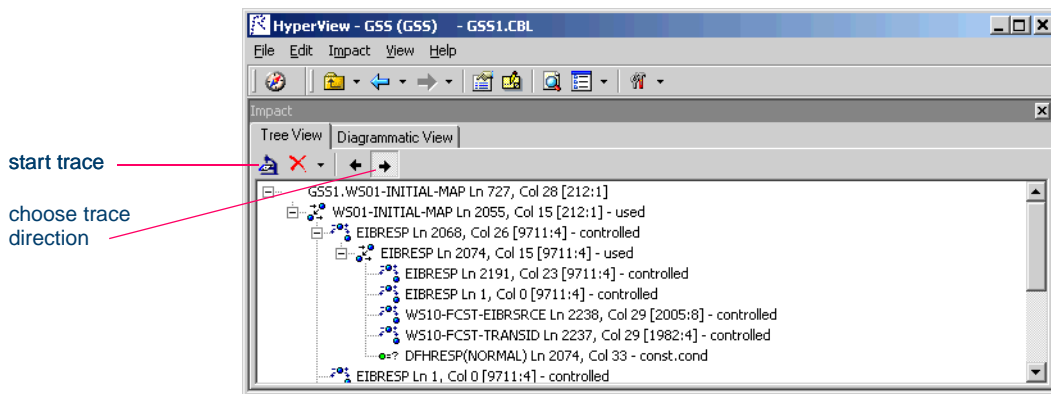
- 1 In the Source or Context pane, select the construct that contains the startup item or items. In the Impact pane, click the  button on the tool bar to view the flow of data into the startup items, click the  button to view the flow of data out of the startup items.
- 2 Click the  button on the tool bar to start the trace. HyperView displays the trace results in the Impact pane (Figure 5-10 on page 5-14). Click the plus sign (+) next to a data item to expand its hierarchy. Click the minus sign (-) to collapse the hierarchy.

**5-14** Analyzing Impact Traces  
*Using the Impact Pane*

Items that have not been visited are displayed in **bold** type. Repeated items are displayed in gray type. Repeated items at a different location are displayed in blue type. If a repeated item at a different location has not been visited, it is displayed in **bold blue** type.



**Note:** Figure 5-10 shows the impact trace with the full detail for each variable — the variable’s location in source, relationships, and memory offset and size. Use the options for the Impact pane to hide or show these details, as described in [“Setting Impact Pane Options” on page 5-15](#). For a description of the relationships, see Table 5-1 on page 5-8.



Figure 5-10 *Impact Pane with Trace Hierarchy*



**3** Click the Diagrammatic View tab to display the impact trace in a diagram (Figure 5-11 on page 5-15). The startup item is displayed in red. Items that have not been visited are displayed in blue. Usage is similar to that for the Diagrammer tool described in *Analyzing Projects*.

**Tip:** If you choose to hide source and memory location data, you can place your cursor over a variable for a moment to display the data in a tool tip.

**4** Select an item in the trace and click the  button to start a trace for that item. The new trace is displayed below the previous trace. To delete a trace, select its startup item and click the  button on the

tool bar. To clear the pane, click the  button next to the  button and choose **Remove All** in the drop-down menu.



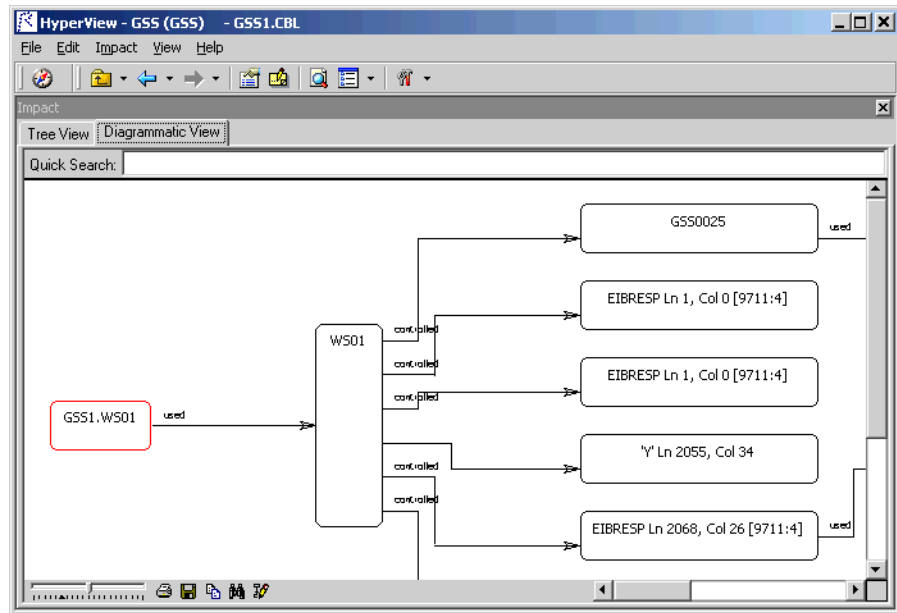
- 5 Click the  or  buttons on the tool bar to reverse the direction of the trace.

Figure 5-11 *Impact Pane with Trace Diagram*



### Setting Impact Pane Options

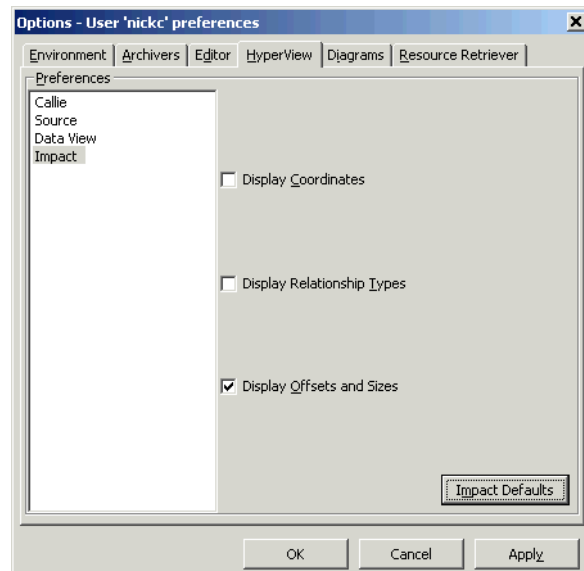
Use the Options window for the Impact pane to control the level of detail in the impact trace — whether to show or hide the locations of variables in source, their relationships, and their memory offsets and sizes.

**Note:** Use the Impact Report Options described in steps [4-6](#) on pages 5-7 and 5-8 to create scoped traces in the Impact pane. A scoped trace filters out unwanted relationships.

**To set Impact Pane options:**

- 1 In the **View** menu, choose **User Preferences**. The User Preferences window opens. Click the **HyperView** tab. In the Preferences pane, click **Impact**. The User Preferences window for the Impact pane opens (Figure 5-12).

Figure 5-12 *Impact Pane User Preferences Window*



- 2 Choose any combination of:
  - **Display Coordinates** if you want the trace to show line and column numbers for variables.
  - **Display Relationship Types** if you want the trace to show the relationships of variables.
  - **Display Offsets and Sizes** if you want the trace to show the memory offsets and allocations for variables.

**Tip:** You can restore the default option settings by clicking **Impact Defaults**, then choosing **Restore Impact Defaults** in the drop-down menu. Choose **Save To** in the drop-down



menu to save the option settings to a file. Choose **Load From** in the menu to restore the option settings from a file.

- 3A** Click **Apply** if you want to save your settings without dismissing the Options window.
- 3B** Click **OK** if you want to save your settings and dismiss the Options window.

### ***What's Next?***

That's all you need to know to analyze impacts in HyperView. The next chapter looks at how you analyze program control flows.

**5-18** Analyzing Impact Traces  
*What's Next?*

# Analyzing Program Control Flows



**C**ontrol flows describe the processing paths in a program. Call flows, decision flows, and statement flows each offer a different way of understanding the procedures in a program. HyperView offers four related resources for analyzing program control flows:

- The Callie pane displays a diagram that shows the flow of control between paragraphs or procedures in a program.
- The Execution Path pane displays a hierarchical view and diagram that show the conditions that determine the flow of control in a program.
- The Flowchart pane displays a diagram that shows the flow of control between statements in a paragraph or procedure.
- The Animator lets you step through the code displayed in a HyperView pane.

**Note:** Flowchart, Execution Path, and Animator are not available for Natural programs. Callie and Animator are not available for PL/I programs. In this chapter, the term “paragraph” also refers to Natural subroutines and PL/I procedures.

## ***Using the Callie Pane***

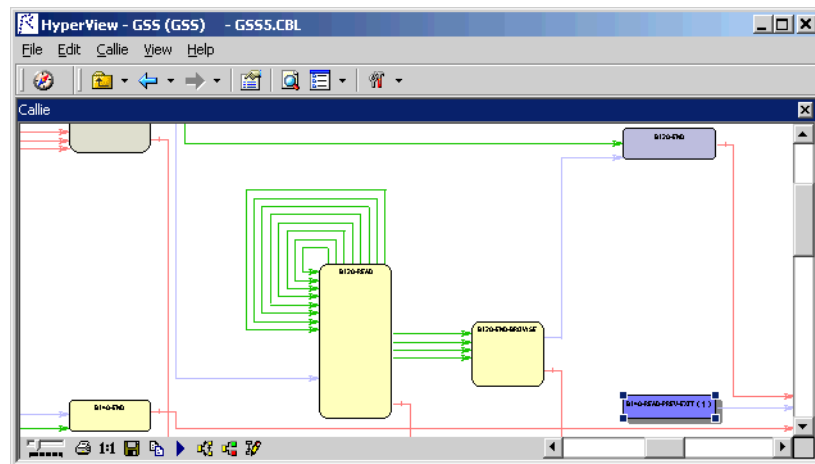
The Callie pane (Figure 6-1) displays a diagram that shows the flow of control between paragraphs in a program. Calls are traced to and from a *seed* paragraph — either the paragraph selected in the Source or Context panes when Callie is opened or the paragraph selected when Callie is re-drawn. Usage is similar to that for the Diagrammer tool described in *Analyzing Projects*.

The label of the box representing a paragraph contains the name of the paragraph. The selected paragraph is displayed with a background shadow. Call relationship lines are color-coded as follows:

- Black = PERFORM
- Green = GO TO
- Red = Fall Thru

Options control the depth of the flow trace and the colors of display objects. For more information, see [“Setting Callie Pane Options”](#) on page 6-4.

Figure 6-1 *Callie Pane*



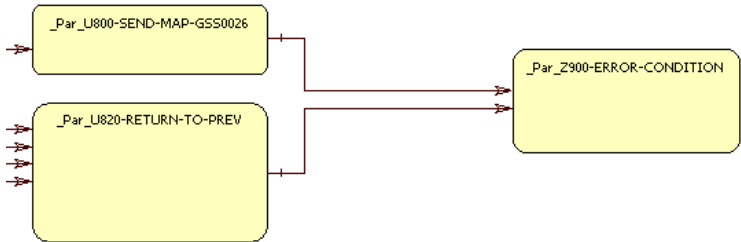
### Choosing the Diagram View

The Callie pane offers two views of program control flows, a *subgraph view* and a *subtree view*. Figure 6-2 and Figure 6-3 show the same paragraphs in both views.

#### Subgraph View

The subgraph view offers a cyclic representation of the information in the diagram. Paragraphs are drawn once. Relationship lines cross. Subgraph views are often easier to understand than subtree views. Choose **SubGraph mode** in the **Callie** menu to select the subgraph view.

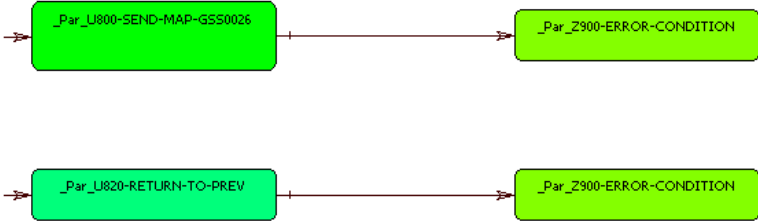
Figure 6-2 *Subgraph View*



#### Subtree View

The subtree view offers a linear representation of the information in the diagram. Paragraphs are drawn as many times as necessary. Relationship lines do not cross. Use this view if too many intersecting lines make a subgraph view hard to read. Choose **SubTree mode** in the **Callie** menu to select the subtree view.

Figure 6-3 *Subtree View*



### Reconstructing Diagrams

Select a paragraph in the diagram and choose **Reconstruct diagram** in the **Callie** menu to redraw the diagram with the control flow to and from the selected paragraph.

### Setting Callie Pane Options


Use the Callie User Preferences and Project Options windows to control the depth of the flow trace, the constructs displayed, and the colors of display objects. Callie User Preferences control the appearance and behavior of Callie across workspaces. Callie Project Options control the appearance and behavior of Callie in the selected project.

#### Setting Callie User Preferences (Subgraph View Only)

Callie makes it easy to track execution paths in subgraph views by displaying each node you visit in a different color. The colors range from the color specified for the selected paragraph to the color specified for unselected, or “default,” paragraphs — from violet to yellow in the preset colors, for example. The *track length* determines the number of visited nodes that Callie displays in different colors.

#### To set Callie User Preferences:

- 1 In the **View** menu, choose **User Preferences**. The User Preferences window opens. Click the **HyperView** tab. In the Preferences pane, click **Callie**. The Callie User Preferences window opens (Figure 6-4 on page 6-5).
- 2 The current default background color of the box representing a paragraph is displayed in the **Default Box Color** drop-down. The current background color of the box representing the selected paragraph is displayed in the **Selected Box Color** drop-down.

Click the adjacent  buttons to edit the color of the paragraphs. A standard Windows color control is displayed. Use the **Palette** tab to select the color from the Windows palette. Use the **System** tab to match the color with the color of standard Windows elements.

- 3 In the **Track Length** combo box, enter the number of visited nodes that Callie displays in different colors.

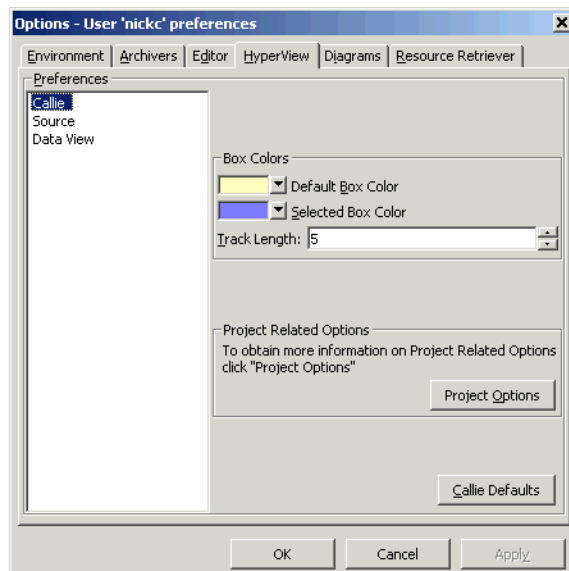
**Tip:** You can restore the default option settings by clicking **Callie Defaults**, then choosing **Restore Defaults** in the drop-down menu. Choose **Save To** in the drop-down menu to save the option settings to a file. Choose **Load From** in the menu to restore the option settings from a file.

**4A** Click **Apply** if you want to save your settings without dismissing the User Preferences window.

**4B** Click **OK** if you want to save your settings and dismiss the User Preferences window.

**Tip:** Click **Project Options** to open the Project Options window.

Figure 6-4 *Callie User Preferences Window*



### Setting Callie Project Options

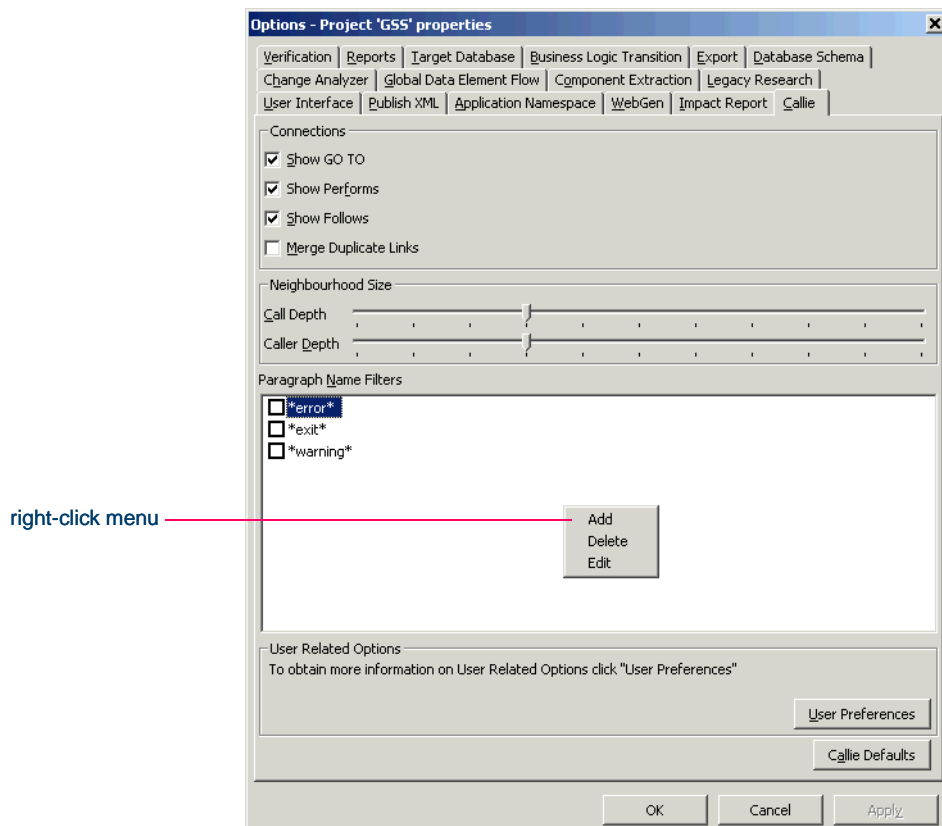
Callie Project Options control the *depth* of the flow trace — how many calling and called nodes in a sequence the diagram displays to and from the seed paragraph. You can also set Project Options that limit the rela-

relationship types the diagram displays (PERFORMs, GO TOs, or Fall Thrus) and filter out paragraphs by name.

**To set Callie Project Options:**

- 1 In the **View** menu, choose **Project Options**. The Project Options window opens. Click the **Callie** tab. The Callie Project Options window opens (Figure 6-5).

Figure 6-5 Callie Project Options Window



- 2 In the Connections pane, deselect relationship types you do not want to view in the diagram. Deselect:
  - **Show GO TO** to hide GO TO relationships.



- **Show Performs** to hide PERFORMS relationships.
  - **Show Follows** to hide Fall Thru relationships.
- 3** In the Connections pane, select **Merge Duplicate Links** to display a single relationship line rather than multiple lines when constructs are connected by identical call relationships.
  - 4** In the Neighborhood Size pane, use the **Call Depth** and **Caller Depth** sliders to specify the depth of the flow trace — how many calling and called nodes in a sequence, respectively, the diagram displays to and from the seed paragraph.
  - 5** In the Paragraph Name Filters pane, select the pattern that matches the names of paragraphs you want to exclude from the diagram. The recognized patterns are listed in the Paragraph Name Filters pane.

Right-click in the Paragraph Name Filters pane and choose **Add** in the pop-up menu to add a pattern to the list. The system displays an empty text field next to a selected check box. Enter the name of the new pattern in the field and click outside the field. You can use wildcard patterns allowed in LIKE statements by Visual Basic for Applications (VBA).

You can edit a pattern by right-clicking it and choosing **Edit** in the pop-up menu. You can delete a pattern by right-clicking it and choosing **Delete** in the pop-up menu.

**Tip:** You can restore the default options settings by clicking **Callie Defaults**, then choosing **Restore Callie Defaults** in the drop-down menu. Choose **Save To** in the drop-down menu to save the option settings to a file. Choose **Load From** in the menu to restore the option settings from a file.

- 6A** Click **Apply** if you want to save your settings without dismissing the Options window.
- 6B** Click **OK** if you want to save your settings and dismiss the Options window.

**Tip:** Click **User Preferences** to open the User Preferences window.

## Using the Execution Path Pane

The Execution Path pane (Figure 6-6) displays a hierarchical view and diagram of the conditions that determine the control flow in a program. Each view traces the control flow from the first involved condition to the paragraph or statement selected in the Source or Context panes — the *seed* construct. Use an execution path to ensure that a code segment encapsulates all of the business logic for a candidate business rule.

**Note:** The Execution Path tool may give incorrect results if the program contains a paragraph that is used in multiple PERFORM statements or in some combination of PERFORM and GOTO statements.

### Generating an Execution Path

Follow the steps below to generate an execution path.

#### To generate an execution path:


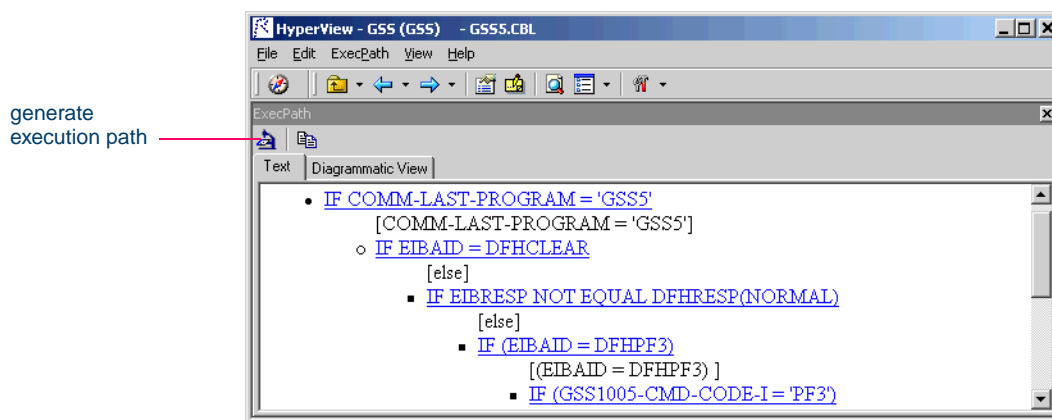

- 1 In the Source or Context pane, select the seed construct. In the Execution Path pane, click the  button on the tool bar. HyperView displays the execution path for the selected construct (Figure 6-6).

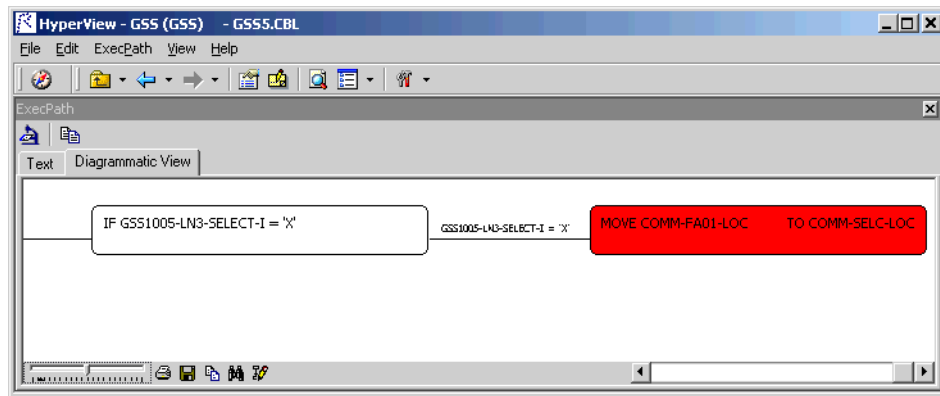
Figure 6-6 Execution Path Pane with Hierarchy




**Tip:** Click the  button on the tool bar to copy the hierarchy to the clipboard with all formatting preserved.

- 2 Click the Diagrammatic View tab to display the execution path in a diagram (Figure 6-7). The seed item is displayed in red. Usage is similar to that for the Diagrammer tool described in *Analyzing Projects*.

Figure 6-7 *Execution Path Pane with Diagram*



- 3 Select a statement in the execution path and click the  button to generate the execution path for that statement. The new execution path replaces the previous path.

## Using the Flowchart Pane

The Flowchart pane (Figure 6-8 on page 6-10) displays a statement flow diagram for the paragraph selected in the Source or Context panes. Use the right-click menu to navigate to related constructs in the parse tree. Otherwise, usage is similar to that for the Diagrammer tool described in *Analyzing Projects*.

The label of the box representing a statement contains the abbreviated text of the statement. Place your cursor over a label for a moment to display a tool tip with the full text of the statement. Captions along relationship lines show the conditions or blocks that determine the flow.

For Cobol programs, ordinary statements are displayed in boxes with a green background. Paragraph PERFORM statements are displayed in

**6-10** Analyzing Program Control Flows  
*Using the Flowchart Pane*

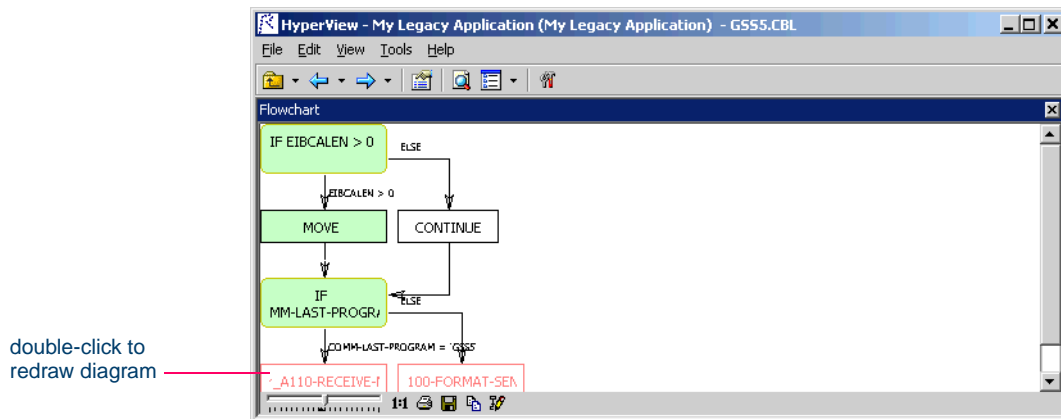
boxes with a pink border and white background. The selected statement is displayed with a background shadow.

Two artificial constructs — the CONTINUE used to represent an empty IF, THEN, or ELSE branch and the PARAGRAPH that represents the object of a GO TO or PERFORM statement — are displayed in boxes with a black border and white background and cannot be selected.

Double-click a Paragraph PERFORM statement to redraw the diagram with the paragraph flow for the called paragraph. Double-click the paragraph for a GO TO statement to redraw the diagram with the paragraph flow for that paragraph.

For PL/I programs, the right-click menu prompts you to switch to the targets of CALL, RETURN, and GO TO statements. If no targets are available, the statement is displayed with a black border.

Figure 6-8 *Flowchart Pane*



## Using the Animator

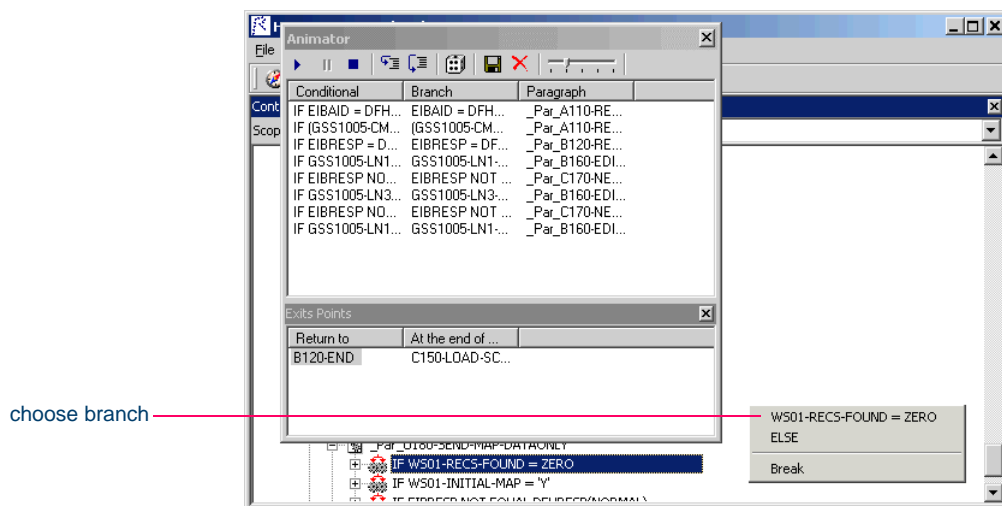
The Animator (Figure 6-9) lets you step through the code displayed in a HyperView pane. You can choose program branches yourself, or have the animator choose them randomly.


**Tip:** Animator is especially useful in tracking execution paths in Callie subgraph views.

### To step through code with the Code Animator:

- 1 In the Source or Context pane, select the construct you want to start the animation from, then choose **Animator** in the **View** menu. The Animator window opens (Figure 6-9).


Figure 6-9 *Animator Window with Context Pane*





- 2 Click the  button on the tool bar if you want Animator to choose logical branches randomly.






**Tip:** Use the slider at the right of the tool bar to set the speed of the animation.

## 6-12 Analyzing Program Control Flows *What's Next?*

- 3 Click the  button on the tool bar to start the animation. If you are choosing program branches yourself, Animator stops at each condition it encounters and displays a pop-up menu that prompts you to choose a logical branch. Click the branch you want to go down, or click **Break** to pause the animation.

HyperView populates the Animator window with each condition it encounters, the logical branch that you or the Animator chose, and the paragraph that contains the condition. The Exit Points pane at the bottom of the window displays the external paragraph currently being executed and the calling paragraph. You can hide the Exit Pane by deselecting the **Show Exit Points** choice in the **Animator** menu.

**Tip:** Click the  button on the tool bar to step through the code manually. Each time you click the button the Animator steps into the next piece of program code. Click the  button if you want to step through the code manually but step over Paragraph PERFORM statements.

- 4 Click the  button on the tool bar to pause the animation. Click the  button to restart the animation. Click the  button to stop the animation.
- 5 Click the  button on the tool bar to save the animation results to a list named Trace. You can view the list in the HyperView List Browser, as described in [“Using the List Browser Window” on page 1-18](#). Click the  to delete the results from the Animator window.

### ***What's Next?***

Now that you know how to analyze procedure flows in HyperView, let's look at how you use the Rules pane to extract business rules. That's the subject of the final chapter.

# Extracting Business Rules



**M**uch of the code in a legacy application — by some estimates as much as 80% — is devoted to tasks with only a marginal relationship to the business logic of the application. *Business rule extraction* lets you separate the application's business logic from these other tasks — data transfer, in particular.

That makes the application easier to understand, document, maintain, and modernize. It also makes it easier to determine the overlap between legacy applications on the one hand, and any gaps in their functionality on the other.

## Understanding Business Rules

A *business rule* is a named container that identifies and documents code segments according to their business function. A business rule named Calculate Date Difference, for example, might consist of this segment:

```
COMPUTE WS-DATE-VARIANCE =  
WS-C-CARD-DATE-CCYYMM - WS-TODAYS-DATE-CCYYMM.
```

### **Guidelines for Assigning Segments**

There are just a few guidelines for assigning code segments to business rules:


- A business rule can contain more than one code segment. The segments can derive from different programs, as long as the programs are part of the same project.
- If the segment you are assigning is contiguous with or overlaps an existing segment, HyperView joins the two segments to form a single segment. If the segment you are assigning is contiguous with or overlaps *two* existing segments, HyperView joins all three segments to form a single segment. Segments for *different* rules can overlap.
- A segment must be continuous.


### **Methods for Assigning Segments**

The HyperView Rules pane lets you create business rules from code segments extracted manually from source or autodetected. You can also create business rules from candidates listed in Clipper — in fact, the autodetection methods in the Rules pane are simply more elaborate versions of the rule detection filters you can define in Clipper.

### **Understanding the Rules Pane**

The Rules pane consists of a top-level view that lists business rules in the project (Figure 7-1 on page 7-3) and subviews for rule properties and segments. You can edit a rule's properties as described in [“Editing Rule Properties” on page 7-14](#).

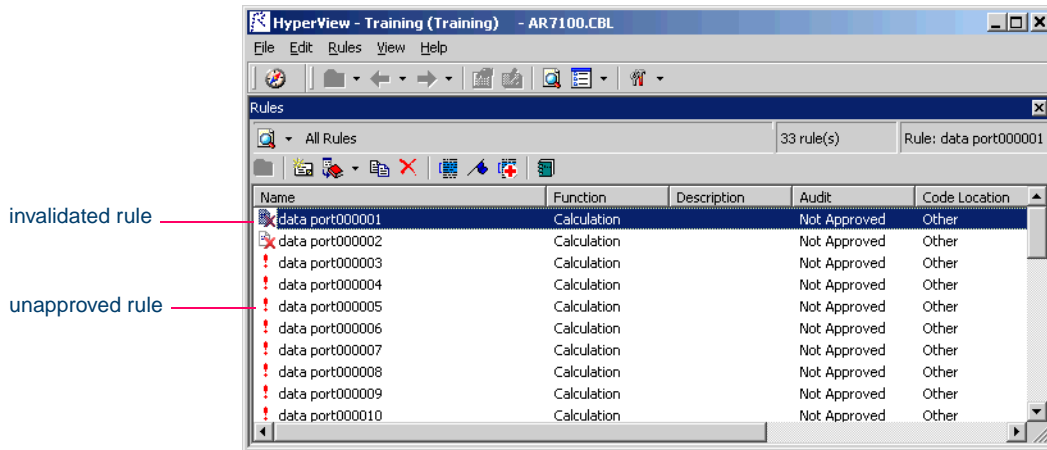
**Unapproved Rules** Newly created rules are considered unapproved, pending a decision that the rule represents valid business logic. The top-level view of the Rules pane lists unapproved rules with a  symbol. Approve a rule as described in [“Editing the Audit Property” on page 7-15](#).

**Invalid Rules** Refreshing or editing source code may result in a mismatch between rules and segments. Lines of code may have been added or deleted during the refresh or edit, causing a rule no longer to be synchronized with its segments after re-verification. The top-level view of the Rules pane lists these *invalidated rules* with a  symbol. Validate



rules as described in [“Validating Business Rules after Refreshing or Editing Code”](#) on page 7-11.

Figure 7-1 Rules Pane — Top-Level View



## Extracting Business Rules Manually

Even if you plan to autodetect business rules, it’s important to know how to assign segments to rules manually, because in some cases you will have to edit or supplement the segment that has been auto-assigned to the rule — an impact analysis or an execution path may show that you need to add another segment to the rule, for example.

### To extract business rules manually:

- 1 In the Source or Context pane, select the segment you want to assign to a business rule. In the Rules pane, choose **Create Rule** in the **Rules** menu. The New Rule dialog opens (Figure 7-2 on page 7-4).

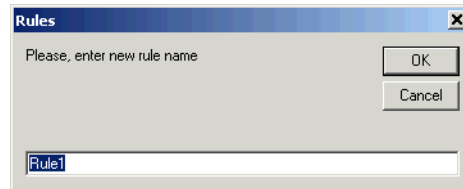
**Tip:** You can select a code segment as *construct* or *text*. Select a segment as text when you want to select either more or less code than a construct contains.

To select a segment as text, click-and-drag from the first line of the segment to the last line of the segment. The segment is

**7-4** Extracting Business Rules  
*Extracting Business Rules Manually*

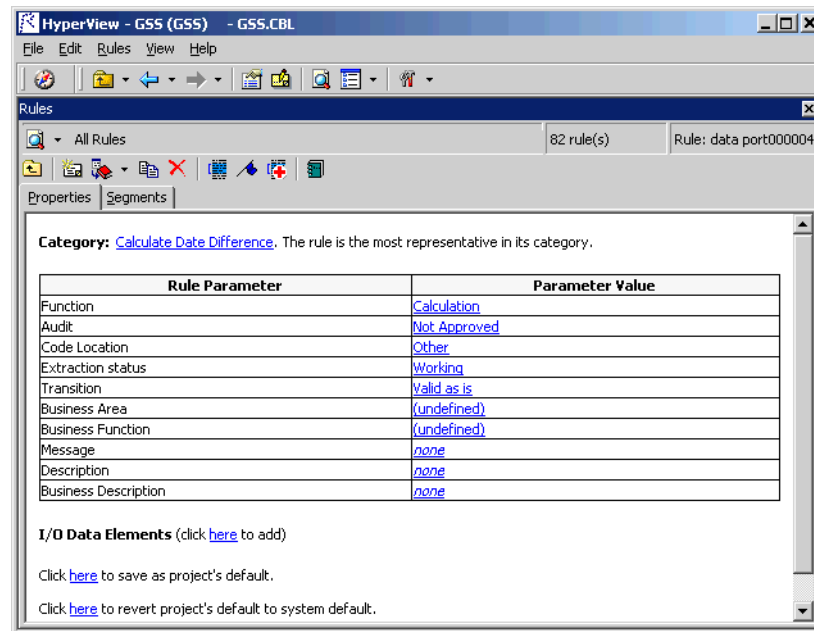
highlighted in blue. To select a segment as construct, click inside the construct in the Source pane. The segment is highlighted in yellow.

Figure 7-2 *New Rule Dialog*



- 2 Enter the name of the business rule in the text field and click **OK**. The name must be unique in the workspace that contains the project. HyperView displays the properties of the new rule in the Properties tab (Figure 7-3). Edit the new rule's properties as described in [“Editing Rule Properties”](#) on page 7-14.

Figure 7-3 *Rules Pane — Properties Tab*




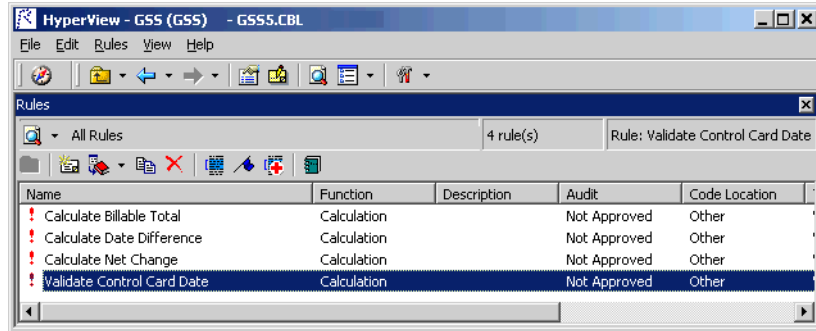
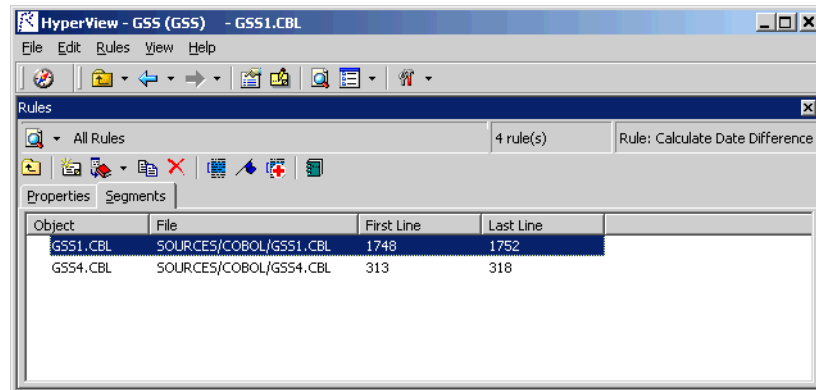
- 3 Click the  button on the tool bar to navigate to the top level of the Rules pane, where you can view the new rule in the list of rules in the project (Figure 7-4 on page 7-5).

Figure 7-4 *Rules Pane — Top-Level View*



- 4 To add a segment to the business rule, select the rule in the top-level view, then select the segment in the Source or Context pane. Choose **Assign Segment** in the **Rules** menu. HyperView displays the location of the segment in the Segments tab (Figure 7-5).

Figure 7-5 *Rules Pane — Segments Tab*





## **Autodetecting Business Rules**

The autodetect methods in the Rules pane are predefined filters similar to the rule detection filters you can set in Clipper. The methods identify code segments that perform common tasks in legacy applications:

- *Flow to Field* identifies fields to which values are moved.
- *MOVES to Screen Message* identifies message literals moved to screen fields.
- *Handle Not Found* identifies exception-handling code for NOT FOUND conditions.
- *I/O Rules* identifies input/output dataports.
- *Screen Validation* identifies code that validates screen fields.
- *Test on Field* identifies code that validates *any* field in a structure.

**Note:** Autodetect is not available for PL/I and Natural programs.

To invoke a method, click the  button next to the  button on the tool bar and choose the method from the drop-down menu. Available options and results are described below. Masks can contain wildcard characters allowed in LIKE statements by Visual Basic for Applications (VBA).

**Tip:** You can cancel autodetection at any time during the process by clicking the **Cancel** button in the upper righthand corner of the HyperView window.

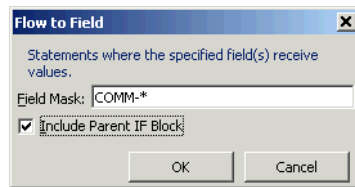
HyperView automatically assigns each segment extracted by the filter to a new business rule. Rule names consist of a prefix with the name of the method and a numeral that identifies the order in which *all rules* in the project were created. So, if a Calculate Date Difference rule already exists, and you choose the Flow to Field method, HyperView will generate rules named FlowToField2, FlowToField3, FlowToField4, and so forth.

The Audit property for an autodetected rule is initially set to Not Approved, pending a decision that the detected rule represents valid business logic. You can change rule names and properties as described in [“Using the Rules Pane” on page 7-13](#).

### **Flow to Field**

The Flow to Field method identifies segments containing fields to which values are moved (Figure 7-6). Select the **Include Parent IF Block** option to return a segment that includes the parent IF block for the MOVE statement(s). You must specify a field mask or nothing will be returned.

Figure 7-6 *Flow to Field Method*



The mask in Figure 7-6 identifies segments containing code like this:

```
IF GSS0026-SELECT1-I = 'X'
  MOVE GSS0026-CUSTNUM1-I TO COMM-CUST-NUM
  PERFORM U820-RETURN-TO-PREV.
```

### **MOVES to Screen Message**

The MOVES to Screen Message method identifies segments containing message literals moved to screen fields (Figure 7-7). The segment includes the IF block that contains the field. You must specify a field mask or nothing will be returned.

**Note:** If a message literal occurs in more than one segment, the method returns only the first occurrence.

Figure 7-7 *MOVES to Screen Message Method*



The mask in Figure 7-7 identifies segments containing code like this:

```
IF GSS1003-FUNCTION-CODE-I = 'AA'
OR GSS1003-FUNCTION-CODE-I = 'TM'
```

## 7-8 Extracting Business Rules *Autodetecting Business Rules*

```
MOVE SPACES TO GSS1003-MSG-LINE-O
MOVE '** FUNCTION NOT OPERATIONAL AT THIS TIME' TO
  GSS1003-MSG-LINE-O
MOVE -1 TO GSS1003-FUNCTION-CODE-L
PERFORM U150-SEND-MAP-DATAONLY.
```

### **Handle Not Found**

The Handle Not Found method identifies segments containing exception handling code for NOT FOUND conditions. Segments are returned for the handle condition statement, the affected statement, and the handler itself, as in:

```
EXEC CICS HANDLE      CONDITION
                      ERROR (BB-020-FILE-ERROR)
                      NOTFND (BB-030-NOT-FOUND)
...
EXEC CICS READ        DATASET ('EMERMSG')
                      RIDFLD (WW-MESSAGE-KEY)
                      INTO (WW-PEGS-MSG-RECORD)
...
BB-030-NOT-FOUND.
  MOVE WW-ERROR-MESS2 TO WW-MSG-TEXT (WS-SUB).
  MOVE 'Y' TO WI-ERROR-IND.
  GO TO BB-040-INCREMENT.
```

### **I/O Rules**

The I/O Rules method identifies segments containing input/output dataports, as in:

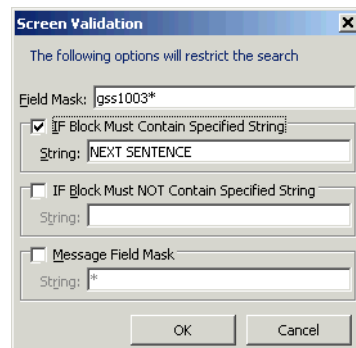
```
EXEC CICS READ
  DATASET ('FCSTNARR')
  INTO (FN01-GSS-NARRATIVE-REC)
  RIDFLD (FN01-KEY)
  RESP (EIBRESP)
END-EXEC.
```

### **Screen Validation**

The Screen Validation method identifies segments containing code that performs validation of screen fields or their synonyms (Figure 7-8 on page 7-9). A synonym is a data field whose value is related to the value of the matched field — a field whose value is assigned by a MOVE or REDEFINE statement, for example.

The segment includes the IF block that contains the field. Use the masks to specify the fields being validated and the fields to which message literals are moved. You can narrow the search by specifying a *complete* string that the IF block must or must not contain— wildcards are not allowed in the IF block string specification.

Figure 7-8 Screen Validation Method



The mask in Figure 7-8 identifies segments containing code like this:

```
IF (GSS1003-CMD-CODE-I = 'ENTER')
OR (GSS1003-CMD-CODE-I = 'PF3')
OR (GSS1003-CMD-CODE-I = SPACES OR LOW-VALUES)
  NEXT SENTENCE
ELSE
  MOVE -1 TO GSS1003-CMD-CODE-L
  MOVE WS10-FCST-MSG-4 TO GSS1003-MSG-LINE-O
  PERFORM U150-SEND-MAP-DATAONLY.
```

### Test on Field

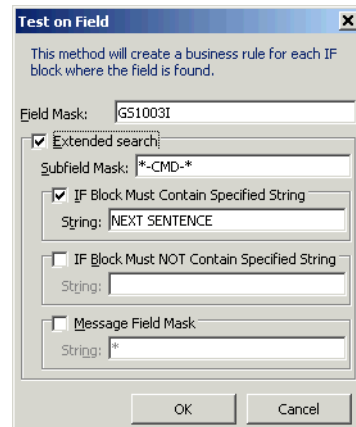
The Test on Field method identifies segments containing code that performs validation of *any* field or its synonym in the structure specified in the mask (Figure 7-9 on page 7-10). A synonym is a data field whose value is related to the value of the matched field — a field whose value is assigned by a MOVE or REDEFINE statement, for example.

The segment includes the IF block that contains the field. Select **Extend search** to include a subfield mask. Use the masks to specify the fields being validated and the fields to which message literals are moved. You can narrow the search by specifying a *complete* string that the IF block

**7-10** Extracting Business Rules  
*Extracting Business Rules with Clipper*

must or must not contain— wildcards are not allowed in the IF block string specification.

Figure 7-9 *Test on Field Method*



The mask in Figure 7-9 identifies segments containing code like this:

```
IF (GSS1003-CMD-CODE-I = 'ENTER')
OR (GSS1003-CMD-CODE-I = 'PF3')
OR (GSS1003-CMD-CODE-I = SPACES OR LOW-VALUES)
NEXT SENTENCE
ELSE
MOVE -1 TO GSS1003-CMD-CODE-L
MOVE WS10-FCST-MSG-4 TO GSS1003-MSG-LINE-O
PERFORM U150-SEND-MAP-DATAONLY.
```

## ***Extracting Business Rules with Clipper***


Clipper is a staging tool that lets you create lists of candidates for business rule extraction. You can use predefined search filters for rules, or define your own filters. Follow the instructions in steps [1-9](#) on page 4-4 to extract business rules with Clipper.

**Note:** Rules autodetected in Clipper are considered unapproved, pending a decision that the detected rule represents valid business logic. Approve a rule as described in [“Editing the Audit Property”](#) on page 7-15.



## Validating Business Rules after Refreshing or Editing Code

Refreshing or editing source code may result in a mismatch between rules and segments. Lines of code may have been added or deleted during the refresh or edit, causing a rule no longer to be synchronized with its segments after re-verification.

The top-level view of the Rules pane lists these *invalidated rules* with a  symbol. You can re-synchronize the segments manually, as described in “[Extracting Business Rules Manually](#)” on page 7-3, or automatically, as described below. A rule is considered invalid until *all* of its invalid segments have been reassigned or deleted.

### To validate rules automatically:

- 1 In the top-level view of the Rules pane (Figure 7-1 on page 7-3), select the rules you want to validate and choose **Validate Segment** in the **Rules** menu. The Validation Options dialog opens (Figure 7-10).


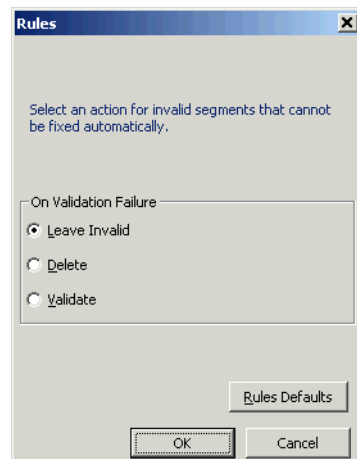
**Tip:** You can also autovalidate individual segments. In the Segments tab (Figure 7-5 on page 7-5), select the segment you want to autovalidate and choose **Validate Segment** in the **Rules** menu. Invalid segments are listed with a .

Figure 7-10 Validation Options Dialog



- 2 In the On Validation Failure pane, choose the radio button for the action you want to take if a segment cannot be autosynchronized:
  - **Leave Invalid** if you want the tool to continue to treat invalid segments as invalid. The rule remains invalid.
  - **Delete** if you want the tool to delete invalid segments from the rule. The rule becomes valid.
  - **Validate** if you want the tool to treat invalid segments as valid. The rule becomes valid.

Click **OK**. You can cancel autovalidation at any time during the process by clicking the **Cancel** button in the upper righthand corner of the HyperView window.

**Note:** You can restore the default settings by clicking **Rules Defaults**, then choosing **Restore Defaults** in the drop-down menu. Choose **Save To** in the drop-down menu to save the settings to a file. Choose **Load From** in the menu to restore the settings from a file.

#### ***Some Limitations on Automatic Rule Validation***

Automatic rule validation will fail to re-synchronize a segment if the segment has been modified in any way during the refresh or edit — if a line has been inserted in the segment code, for example, or if the text of a line has been changed. You must re-synchronize these segments manually.



Note, too, that if the code for a segment is duplicated in a program, the autovalidation method will synchronize the rule with the first instance of the duplicated code, whether or not that instance was originally associated with the rule. The rule will be the same, but the program context may be different from the one you intended. Here, too, you will have to recreate the rule manually.

## Using the Rules Pane

By default, the top-level view in the Rules pane lists every business rule in the project and its properties. Click a business rule in the top-level view to select it. Double-click a rule to open the Rules pane subviews for rule properties and segments.

**Assigning Segments to Rules** To assign a segment to an existing rule, select the rule in the top-level view, then select the segment in the Source or Context pane. Choose **Assign Segment** in the **Rules** menu. HyperView displays the location of the segment in the Segments tab. See [“Guidelines for Assigning Segments” on page 7-2](#) for special assignment rules.

**Deleting Segments from Rules** To delete a segment from a rule, select the segment in the Segments tab and choose **Delete** in the **Rules** menu.

**Flagging Segments in Source** To place a  symbol in the HyperView Source pane display next to each line in the assigned segments, click the  button on the tool bar.

**Renaming Rules** To rename a rule, select it in the top-level view and click the highlight area. In the text box, enter the new name of the rule and press Enter.

**Deleting Rules** To delete a rule, select it in the top-level view and choose **Delete** in the **Rules** menu. You are prompted to confirm the deletion. Click **OK**.

### **Viewing Segments in the Source or Context Pane**

To make it easier to assign segments to business rules, the selection mechanism in the Rules pane departs from the HyperView selection model. Selecting a rule in the top-level view does *not* move the cursor in the Source or Context panes to the segment for the rule. To view the segment in the Source or Context panes, select the rule, then press the space bar.

## Editing Rule Properties

A business rule is initially just a shell. Before the rule will be useful to other members of your organization, you need to document it by editing its properties in the Properties tab. To open the Properties tab for a rule, double-click the rule in the top-level view of the Rules pane. Click the Properties tab (Figure 7-3 on page 7-4).

**Tip:** You can edit rule properties before or after you assign code segments.

### Editing the Category Property

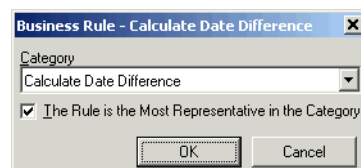
When you create a rule, it is assigned to a default *category* with the same name as the rule. The category is an organizing device for rules that perform the same business function.

The *most representative rule* in a category is the one that best typifies the category. A business analyst might decide, for example, that the code segment for a given rule in a category is more straightforward than the segments for the other rules in the category and therefore better typifies the category.

#### To edit the category property:

- 1 Click the link for the current category name in the **Category** field. The Category dialog opens (Figure 7-11).

Figure 7-11 Category Dialog



- 2 In the category combo box, specify the name of the category.
- 3 If the rule typifies the category, select **The rule is the most representative in the category**.
- 4 Click **OK** to dismiss the dialog and return to the Properties tab.

### ***Editing the Function Property***

The function property identifies the programming task the rule performs.

#### ***To edit the function property:***

- 1** Click the link for the current function in the **Function** field.
- 2** In the pop-up menu, choose:
  - **I/O** if the rule performs an input or output function.
  - **Calculation** if the rule performs a calculation.
  - **Security** if the rule performs a security function.
  - **Decision** if the rule resolves a decision.
  - **Validation** if the rule performs validation.
  - **Other** to define the task the rule performs. Enter the task in the text field in the Function dialog.

### ***Editing the Audit Property***

The audit property identifies whether an autodetected or Clipper-generated rule has been audited and approved.

#### ***To edit the audit property:***

- 1** Click the link for the current audit status in the **Audit** field.
- 2** In the pop-up menu, choose:
  - **Not Approved** if the rule has not been accepted as valid business logic.
  - **Approved** if the rule has been accepted as valid business logic.

**Tip:** Unapproved business rules are denoted by a **!** symbol in the top-level view of the Rules pane.

### ***Editing the Code Location Property***

The code location property identifies the rule as a client or server rule.

#### ***To edit the code location property:***

- 1 Click the link for the current code location in the **Code Location** field.
- 2 In the pop-up menu, choose:
  - **Client** if the code for the rule resides on a client machine.
  - **Other** if the code for the rule resides elsewhere.

### ***Editing the Extraction Status Property***

The extraction status property identifies the status of the rule in an extraction project.

#### ***To edit the extraction status property:***

- 1 Click the link for the current extraction status in the **Extraction Status** field.
- 2 In the pop-up menu, choose:
  - **(none)** if the rule has no extraction status.
  - **Extracted** if the rule has been extracted, but not accepted or rejected.
  - **Working** if the rule is still being extracted.
  - **Accepted** if the rule has been accepted.
  - **Rejected** if the rule has been rejected.

### **Editing the Transition Property**

The transition property identifies the status of the rule in a redevelopment project.

#### **To edit the transition property:**

- 1 Click the link for the current transition status in the **Transition** field.
- 2 In the pop-up menu, choose:
  - **Valid as is** if the rule is valid for redevelopment.
  - **Obsolete** if the rule is obsolete.
  - **Requires modification** if the rule requires modification for redevelopment.
  - **Duplicate** if the rule is a duplicate of a rule being used for redevelopment.

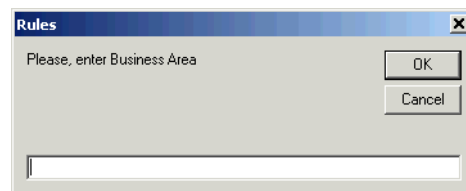
### **Editing the Business Area Property**

The business area property identifies the business domain of the task the rule performs — premium management, for example.

#### **To edit the business area property:**

- 1 Click the link for the current business area in the **Business Area** field. The Business Area dialog opens (Figure 7-12).

Figure 7-12 *Business Area Dialog*



- 2 Enter the business area in the text field.
- 3 Click **OK** to dismiss the dialog and return to the Properties tab.

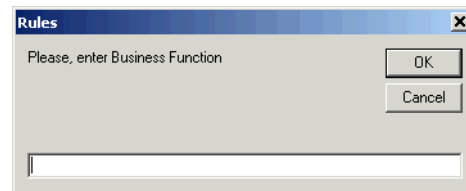
### **Editing the Business Function Property**

The business function property identifies the business task the rule performs — premium qualification, for example.

#### **To edit the business function property:**

- 1 Click the link for the current business function in the **Business Function** field. The Business Function dialog opens (Figure 7-13).

Figure 7-13 *Business Function Dialog*



- 2 Enter the business function in the text field.
- 3 Click **OK** to dismiss the dialog and return to the Properties tab.

### **Editing the Message Property**

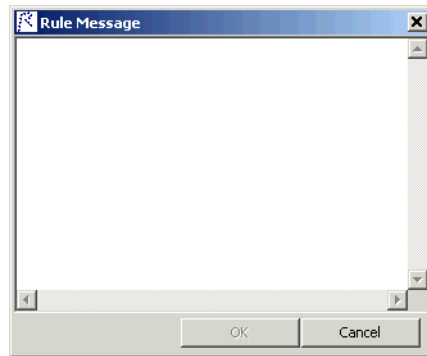
The message property contains the text of the message for autodetected screen message and screen validation rules. The message property is automatically populated by the MOVES to Screen Message, Screen Validation, and Test on Field autodetect methods.

#### **To edit the message property:**

- 1 Click the **none** link for the **Message** field to enter the message text, or if the message text already exists, the **Edit** link to edit the current text. The Rule Message dialog opens (Figure 7-14 on page 7-19).
- 2 Enter the message text in the text area.
- 3 Click **OK** to dismiss the dialog and return to the Properties tab.



Figure 7-14 Rule Message Dialog



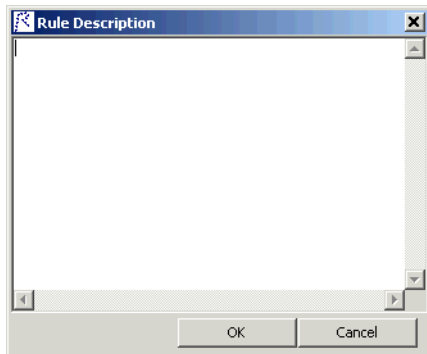
### Editing the Description Property

The description property contains descriptive documentation for the rule. The description is automatically populated by the MOVES to Screen Message, Screen Validation, and Test on Field autodetect methods.

#### To edit the description property:

- 1 Click the **none** link for the **Description** field to create a rule description, or if the description already exists, the **Edit** link to edit the current description. The Rule Description dialog opens (Figure 7-15).

Figure 7-15 Rule Description Dialog



**7-20** Extracting Business Rules  
*Using the Rules Pane*

- 2 Enter the rule description in the text area.
- 3 Click **OK** to dismiss the dialog and return to the Properties tab.

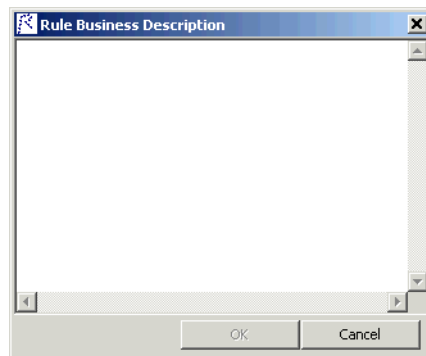
***Editing the Business Description Property***

The business description property contains descriptive documentation about the business role of the rule.

***To edit the business description property:***



- 1 Click the **none** link for the **Business Description** field to create a business description, or if the business description already exists, the **Edit** link to edit the current business description. The Rule Business Description dialog opens (Figure 7-16).

Figure 7-16 *Rule Business Description Dialog*



- 2 Enter the business description in the text area.
- 3 Click **OK** to dismiss the dialog and return to the Properties tab.

### Editing the I/O Data Elements Property

The I/O data elements property identifies the input, output, and input/output fields in the code for the rule. You can populate the property automatically by selecting one or more rules in the top-level view of the Rules pane, then clicking the  button next to the  button on the tool bar and choosing **I/O Data Elements** from the drop-down menu.

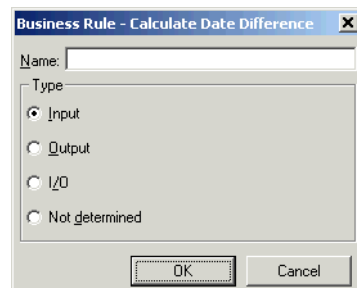
**Note:** The autodetect method for I/O data elements is available only for source files verified with the **Enable Impact Report** option set in the Verification tab of the Project Options window.

The method is based on a set of heuristics designed to produce reasonable output. Results may need to be verified by a detailed impact analysis. For more information on the heuristics, contact support services.

#### To edit the I/O data elements property:

- 1 Click the [here](#) link in the **I/O Data Elements** field. The I/O Data Elements dialog opens (Figure 7-17).

Figure 7-17 I/O Data Elements Dialog



- 2 In the **Name** field, enter the name of the field.
- 3 In the Type pane, choose:
  - **Input** for an input field.
  - **Output** for an output field.
  - **I/O** for an input and output field.
  - **Not determined** if the field type has not been determined.

**Note:** If a data element already exists for a rule, and a data element with the same name but a different type is added to the rule (manually or automatically), the type of the original data element is defined as I/O (input/output).

4 Click **OK** to dismiss the dialog and return to the Properties tab. The specified field is listed in a table in the Properties tab.

**Tip:** To edit a field in the table, select it and choose **Edit** in the pop-up menu. To delete a field, select it and choose **Delete** in the pop-up menu. You are prompted to confirm the deletion. Click **OK**.

5 Repeat steps [1-4](#) for each field you want to add to the table.

### **Assigning Property Values to Multiple Rules**

When you create rules in Clipper, you can specify the properties for every rule in the set of rules Clipper generates ([step 8 on page 4-4](#)). The Rules pane provides similar mechanisms for assigning property values to multiple rules.

#### **Setting Default Property Values**

You can specify that the values of a rule's properties (except the category property) serve as default values for every rule you extract in the project, including rules you autodetect or generate with Clipper. To set default property values, click the [here](#) link in the **Click here to save as project's default** field at the bottom of the Properties tab.

#### **Copying Business Rule Properties**

When you copy the properties of a business rule, you create a rule that has all of the properties of the copied rule except its name. That comes in handy when you need to document a great many rules in a category — you can copy the properties of the first rule you create and assign the appropriate segment to each new version.

To copy a rule's properties, select it in the top-level view of the Rules pane and choose **Copy Properties** in the **Rules** menu. In the Copy Rule dialog, enter the name of new rule and click **OK**.

### Filtering the Business Rules Display

By default, the top-level view in the Rules pane lists every business rule in the project. Filter the display as described below.

#### To filter the business rules display:


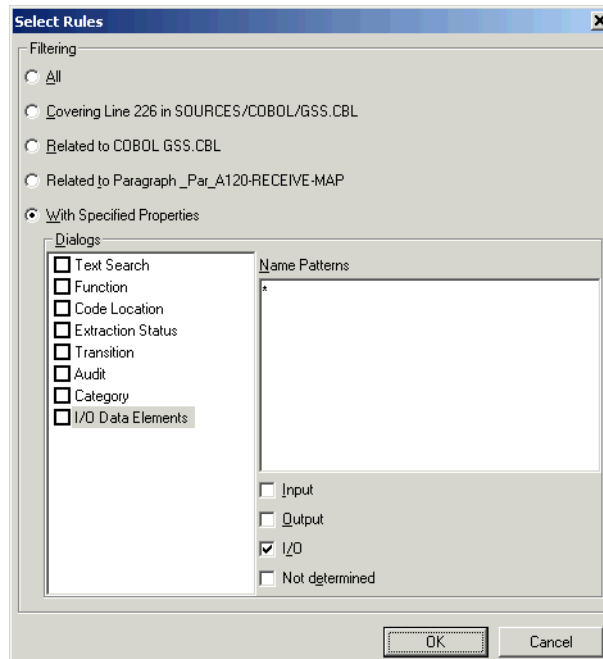
- 1 Click the  button on the tool bar. The Select Rules window opens (Figure 7-18).



Figure 7-18 Select Rules Window (Specified Properties Option)





- 2 In the Filtering pane, choose one of the following:
  - **All** if you want to display every rule in the project.
  - **Covering line *n* in source file** if you want to display only rules with a segment that contains line *n* in the source. The line number corresponds to the first line in the construct or text selected in the Source or Context panes.


- **Related to source file** if you want to display only the rules in the program selected in the Source or Context panes.
- **Related to paragraph** if you want to display only rules with a segment that overlaps the paragraph selected in the Source or Context panes.
- **With specified properties** if you want to display only the rules with the properties specified in the Dialogs pane:
  - Place a check mark next to each property you want to use in the filter. If you do not set the filter to a category, all properties are ANDed in the filter. If you set the filter to a category, all properties are ORed in the filter.
  - Specify the value of each property in the filter by selecting the property and entering its value in the edit fields that open beside the Dialogs pane. You can use wildcards allowed in LIKE statements by Visual Basic for Applications (VBA).

#### **Refreshing the Display**

If the rules display filter is set to anything other than **All** when you generate rules, the rules do not appear in the Rules pane until you refresh the display. To refresh the display, click the  button next to the  button on the tool bar and choose **Refresh** from the drop-down menu.

Keep in mind that HyperView will display only the generated rules in the scope of the filter. If the filter is set to the GSS1 program, for example, only GSS1 rules will be displayed. To display all the rules in the project, click the  button next to the  button and choose **All** from the drop-down menu.


## Generating Business Rule Reports

Business rule reports make it easy to develop the kind of detailed specification you'll need before redeveloping a legacy application in a modern programming language. To generate a report, click the  button on the tool bar and choose the report type from the pull-down menu.

The following reports are available:

- The *Business Rules Report* summarizes the rules in a project. The report lists the name, properties, data elements, and segments of each rule in the project.
- The *Rule Programs Report* cross-references rules with their programs. The report lists the program, name, and extraction status of each rule in the project.
- The *Process Outline Report* shows the context in which business rules are executed. The report lists every paragraph in the selected program, the process outline of the paragraph, every rule with a segment that starts in the paragraph, and the segment itself. A process outline describes the call flow of a paragraph. If you have annotated the paragraph (see [“Annotating a Paragraph”](#) below), the annotation appears in the paragraph column below the paragraph name.
- The *Coverage Report* shows the percentage of program logic that contains business rule segments. The report lists the programs in the project, the total number of lines of code they contain, the number of lines of code with program logic, the number of business rule segments the program contains, and the percentage of program logic that contains business rule segments.

### **Annotating a Paragraph**

Select a paragraph in a HyperView pane and click the  button on the tool bar to open the Annotate dialog, where you can enter a business name and description of the paragraph. The annotation appears in the Paragraph column of the Process Outline Report under the paragraph name.

### ***Printing Reports***

Click **Page Setup** in a report window to set up the page for a printed report. Click **Print** to print the report. The Print dialog opens, where you can set options for the print job. Click **OK**.

### ***Exporting Reports***

You can export business rule reports to a variety of standard formats. Click **Save** in a report window to export a report to HTML, Excel, RTF, Word, or formatted text. A Save As dialog opens, where you can specify the name, location, and file type of the report.

**Note:** Excel is usually the best choice if you need to manipulate the report or perform computations.

### ***What's Next?***

That completes your tour of HyperView! Now you're ready to begin extracting components from your legacy application — self-contained programs that can be reused with other programs in modular fashion. *Creating Components* in the workbench documentation set describes how to extract components.



# Glossary

## Activity Log

The Activity Log is a chronological record of your activities in the current [Asset Transformation Workbench \(ATW\)](#) session.

## ADABAS

ADABAS is a Software AG relational [DBMS](#) for large, mission-critical applications.

## Animator

Animator lets you step through the code displayed in a [HyperView](#) pane. You can choose program branches yourself, or have the animator choose them randomly.

## API

API stands for application programming interface, a set of routines, protocols, and tools for building software applications.

## applet

See [Java applet](#).

## Application Analyzer

Application Analyzer is a set of non-invasive interactive tools used to analyze and document legacy systems.

### **Application Architect**

Application Architect uses advanced algorithms to partition code into new [components](#) and perform [Dead Code Elimination](#).

### **Application Namespace tool**

The Application Namespace tool creates a conveniently organized dictionary that helps you navigate through your system's terminology and modify it as necessary.

### **Application Partitioner**

Application Partitioner identifies legacy subsystems and partitions them into self-contained projects based on an analysis of [repository](#) contents.

### **Application Profiler**

Application Profiler, consisting of [WebGen](#) and [Profiler](#), generates a set of HTML views of a legacy application based on the [object model](#) created in a previous analysis.

### **AS/400**

The AS/400 is a midrange server designed for small businesses and departments in large enterprises.

### **Asset Transformation Workbench (ATW)**

Asset Transformation Workbench (ATW) is a suite of PC-based software products for analyzing, re-architecting, and transforming legacy applications.

### **Batch Application Viewer**

Batch Application Viewer performs low-level analysis of batch processes.

### **batch refresh**

The batch refresh feature lets you register and verify source files in batch mode. Other utilities packaged with the feature let you analyze application [complexity](#), run [WebGen](#), and, if you are licensed to use the Application Architect product, perform [Dead Code Elimination](#).

### **Bird's Eye pane**

The Bird's Eye pane works with the [HyperView](#) Source pane to let you quickly identify the location of a code construct relative to the entire program.

**BMS**

BMS stands for Basic Mapping Support, an interface between application formats and [CICS](#) that formats input and output display data.

**BSTR**

BSTR is a Microsoft format for transferring binary strings.

**business rule**

A business rule is a named container that identifies and documents code segments according to their business function. Business rules encapsulate an application's business logic, making the application easier to understand, document, maintain, and test.

**Business Rule Manager**

Business Rule Manager lets you generate [business rules](#) from code segments extracted manually from source or autodetected.

**Callie pane**

The [HyperView](#) Callie pane displays a diagram that shows the flow of control between paragraphs or procedures in a program.

**CDML**

CDML stands for Cobol Data Manipulation Language, an extension of the [Cobol](#) programming language that enables applications programmers to code special instructions to manipulate data in a [DMS](#) database and to compile those instructions for execution.

**Change Analyzer**

Change Analyzer identifies the class of data items used to perform a business function in a legacy application. Among other uses, it lets you answer the kinds of "What if?" questions posed in the recent past by the industry-wide changes for Y2K, Zip+4, and the Euro dollar: "What if I change the type of this variable, or the length of this field — what *other* fields will I also have to change?"

**CICS**

CICS stands for Customer Information Control System, a program that allows concurrent processing of [transactions](#) from multiple terminals.

### Clipper

The [HyperView](#) Clipper tool lets you create lists of candidates for [business rule](#) extraction, [event injection](#), and other tasks. Each list captures the results of a different stage of your analysis and serves as input for subsequent tasks.

### Cobol

Cobol stands for Common Business-Oriented Language, a high-level programming language used for business applications.

### COM

COM stands for Component Object Model, a software architecture developed by Microsoft to build [component](#)-based applications. COM objects are discrete components, each with a unique identity, which expose interfaces that allow applications and other components to access their features.

### complexity

A [project's](#) complexity is an estimate of how difficult it is to maintain, analyze, transform, and so forth.

### component

A component is a self-contained program that can be reused with other programs in modular fashion.

### Component Maker

The [HyperView](#) Component Maker tool lets you “slice out” [components](#) from legacy applications — not only component executables but associated [Cobol](#) copybooks, [PL/I](#) includes, and [Natural](#) data areas as well.

### computation-based component extraction

Computation-based [component](#) extraction lets you build a component that contains all the code necessary to calculate the value of a variable at a particular point in a program — the value of a DayOfTheWeek variable, for example, where it is used to populate a report attribute or screen.

### Configuration Manager

Configuration Manager is a tool used to enable [Asset Transformation Workbench \(ATW\)](#) products and configure your workbench for the

tools, programming languages, legacy dialects, and character sets in use at your site.

**construct**

A construct is an item in the [parse tree](#) for a source file — a section, statement, condition, variable, or the like. A variable, for example, can be related in the parse tree to any of three other constructs — a declaration, a dataport, or a condition. You view the parse tree for a source file in the HyperView [Context pane](#).

**Context pane**

The [HyperView](#) Context pane displays the [parse tree](#) for the selected source file. The parse tree displays source code constructs — sections, paragraphs, statements, conditions, variables and so forth — in hierarchical form, making it easy to locate code constructs quickly.

**copybook**

A copybook is a common piece of source code to be copied into many [Cobol](#) source programs. Copybooks are functionally equivalent to C and C++ include files.

**CORBA**

CORBA stands for Common Object Request Broker Architecture, an architecture that enables distributed objects to communicate with one another regardless of the programming language they were written in or the operating system they are running on.

**CSD file**

CSD stands for [CICS](#) System Definition. A CSD file is a [VSAM](#) data set containing a resource definition record for every resource defined to [CICS](#).

**database schema**

A database schema is the structure of a database system, described in a formal language supported by the [DBMS](#). In a relational database, the schema defines the tables, the fields in each table, and the relationships between fields and tables.

**dataport**

A dataport is an input/output statement or a call to or from another program.

### **DB/2**

DB/2 stands for Database 2, an IBM system for managing relational databases.

### **DBCS**

DBCS stands for double-byte character string, a character set that uses two-byte (16-bit) characters rather than one-byte (8-bit) characters.

### **DBMS**

DBMS stands for database management system, a collection of programs that enable you to store, modify, and extract information from a database.

### **DDL**

DDL stands for Data Description Language (DDL), a language that describes the structure of data in a database.

### **Dead Code Elimination**

Dead code elimination is a type of [component](#) extraction that removes unused (“dead”) code from a legacy application.

### **decision resolution**

Decision resolution lets you identify and resolve dynamic calls and other relationships that the [parser](#) cannot resolve from static sources.

### **Diagrammer**

Diagrammer lets you view the [relationships](#) between the objects in a [project](#) interactively — programs, files, [DDL](#), [Java](#), screen maps, and more. These relationships describe the ways in which application objects interact. Compare [Quick Diagrammer](#).

### **DMS**

DMS stands for Data Management System, a Unisys database management software product that conforms to the CODASYL (network) data model and enables data definition, manipulation, and maintenance in mass storage database files.

### **domain-based component extraction**

Domain-based component extraction “specializes” a program based on the values of one or more variables. The specialized program is typically intended for reuse “in place” — in the original application but under new external circumstances.

**DPS**

DPS stands for Display Processing System, a Unisys product that enables users to define forms on a terminal.

**ECL**

ECL stands for Executive Control Language, the operating system language for Unisys OS 2200 systems.

**effort**

Effort is an estimate of the time it will take to complete a task related to a [project](#), based on weighted values for selected [complexity](#) metrics.

**EJB**

EJB stands for Enterprise JavaBeans, a [Java API](#) developed by Sun Microsystems that defines a [component](#) architecture for multi-tier client/server systems.

**EMF**

EMF stands for Enhanced MetaFile, a Windows format for graphic images.

**entity**

An entity is an object in the [repository](#) model for a legacy application. The relationships between entities describe the ways in which the elements of the application interact.

**entry point isolation**

Entry point isolation extracts a [component](#) that contains only the functionality and data definitions required for invocation from the selected entry point.

**event injection**

Event injection is a type of [component](#) extraction that adapts a legacy program to asynchronous, event-based programming models.

**Execution Path pane**

The [HyperView](#) Execution Path pane displays a hierarchical view and diagram of the conditions that determine the flow of control in a program.

### **external subroutine extraction**

External subroutine extraction is a type of [structure-based component extraction](#) that replaces a single internal subroutine in a [Natural](#) program with an external subroutine.

### **FCT**

FCT stands for File Control Table (FCT), a [CICS](#) table that contains processing requirements for output data streams received via a remote job entry session from a host system. Compare [PCT](#).

### **Flowchart pane**

The [HyperView](#) Flowchart pane displays a diagram of the flow of control between statements in a paragraph or procedure.

### **Global Data Flow tool**

The [HyperView](#) Global Data Flow tool performs low-level analysis of program data flows.

### **HTML**

HTML stands for HyperText Markup Language, the authoring language used to create documents on the World Wide Web.

### **HyperView**

HyperView is a set of program analysis tools that let you analyze legacy programs interactively, by examining synchronized, complementary views of the same information — source, context, impacts, and so forth.

### **IDL**

IDL stands for Interface Definition Language (IDL), a generic term for a language that lets a program or object written in one language communicate with another program written in an unknown language.

### **IDMS**

IDMS stands for Integrated Database Management System, a Computer Associates database management system for the IBM mainframe and compatible environments.

### **Impact pane**

The [HyperView](#) Impact pane displays a hierarchical view and diagram of the *impact trace* for a program variable. An impact trace describes



how data items interact with each other in a program — exchange values, use each other in computations, and so forth.

**Impact Report pane**

The [HyperView](#) Impact Report pane shows the flow of data from a startup item to every data item that would be affected by its modification. The report is organized in hierarchical form according to the depth of the affected item.

**IMS**

IMS stands for Information Management System, an IBM program product that provides transaction management and database management functions for large commercial application systems.

**Java**

Java is a high-level [object-oriented programming](#) language developed by Sun Microsystems.

**Java applet**

A [Java](#) applet is a program that can be sent with a Web page. Java applets perform interactive animations, immediate calculations, and other simple tasks without having to send a user request back to the server.

**JavaBeans**

JavaBeans is a specification developed by Sun Microsystems that defines how [Java](#) objects interact. An object that conforms to this specification is called a JavaBean.

**JCL**

JCL stands for Job Control Language, a language for identifying a [job](#) to OS/390 and for describing the job's requirements.

**JDBC**

JDBC stands for Java Database Connectivity, a standard for accessing diverse database systems using the [Java](#) programming language.

**job**

A job is the unit of work that a computer operator or a program called a *job scheduler* gives to the operating system. In IBM mainframe operating systems, a job is described with job control language ([JCL](#)).

**job dependencies**

[Batch Application Viewer](#) treats [jobs](#) as dependent if one writes to a dataset and the other reads from the same dataset. Occasionally, you may want to define dependencies between jobs based on other criteria — administrative needs such as scheduling, for example.

**logical component**

A logical component is an abstract [repository](#) object that gives you access to the source files that comprise a [component](#).

**MFS**

MFS stands for Message Format Service, a method of processing [IMS](#) input and output messages.

**Missing Copybooks Resolution tool**

The Missing Copybooks Resolution tool resolves undefined variables in missing [copybooks](#) for [Cobol](#) programs verified with the [relaxed parsing](#) option.

**Model Reference pane**

The [HyperView](#) Model Reference pane displays the [parse tree](#) meta-model in text and diagram form.

**name-based partitioning**

Name-based partitioning is a partitioning algorithm that assigns source files to projects based on text matching of source file names with specified patterns.

**Natural**

Natural is a programming language developed and marketed by Software AG for the enterprise environment.

**object model**

An object model is a representation of an application and its encapsulated data.

**object-oriented programming**

Object-oriented programming organizes programs in terms of objects rather than actions, and data rather than logic.

**ODBC**

ODBC stands for Open Database Connectivity, a standard for accessing diverse database systems.

**orphan**

An orphan is an object that does not exist in the reference tree for any startup object. Orphans can be removed from a system without altering its behavior.

**Orphan Analysis tool**

The Orphan Analysis tool lets you analyze and resolve [orphans](#).

**parser**

The [Asset Transformation Workbench \(ATW\)](#) parser defines the [object model](#) and [parse tree](#) for a legacy application.

**parse tree**

A parse tree defines the relationships among the constructs that comprise a source file — its sections, paragraphs, statements, conditions, variables, and so forth.

**PCT**

PCT stands for Program Control Table, a [CICS](#) table that defines the transactions that the CICS system can process. Compare [FCT](#).

**PL/I**

PL/I stands for Programming Language One, a third-generation programming language developed in the early 1960s as an alternative to assembler language, [Cobol](#), and FORTRAN.

**PL/I Call Diagrammer**

PL/I Call Diagrammer performs low-level analysis of [PL/I](#) programs. Use it to examine call flows for internal procedures that the [Diagrammer](#) is unable to model.

**profile**

Profiles are HTML views into a [repository](#) that show all of the analysis you have done on an application. Profiles are convenient ways to share information about legacy applications across your organization.

### **Profiler**

Profiler is a Web server-based tool that offers company-wide access to [profiles](#) of any repository in your organization. It gives managers, business analysts, testers, and customer support personnel convenient, browser-based access to analyzed legacy code.

### **project**

A project is a logical subdivision of a [workspace](#). You might have a project for the batch portion of your application and another project for the online portion, for example. You can also use a project to collect items for discrete tasks — all the source files affected by a change request, for example.

### **QSAM**

QSAM stands for Queued Sequential Access Method, a type of processing that uses a queue of data records—either input records awaiting processing or output records that have been processed and are ready for transfer to storage or an output device.

### **Quick Diagrammer**

The Quick Diagrammer tool lets you view relationships for selected objects only, rather than an entire project. Compare [Diagrammer](#).

### **refactoring**

Refactoring translates a program into a [component](#) with the same functionality and control flow, but a simpler syntax structure.

### **reference reports**

[Asset Transformation Workbench \(ATW\)](#) offers three related reports that you can use to identify missing or unneeded program elements in application source: an unresolved report, an unREFERRED report, and a cross-reference report.

### **relationship**

The relationships between entities in the [repository](#) model for a legacy application describe the ways in which the elements of the application interact.

### **relationship-based partitioning**

Relationship-based partitioning is a partitioning algorithm that assigns source files to projects based on the extent to which the source

files are related. If two source files reference the same [copybook](#), for example, they can be regarded as “tightly related,” at least as compared with source files that do not reference the same copybook.

**relationship weight**

A relationship weight determines the importance of that [relationship](#) in calculating the connection between source files in [relationship-based partitioning](#).

**relaxed parsing**

Relaxed parsing lets you verify a source file despite errors. Ordinarily, the parser stops at a statement when it encounters an error. Relaxed parsing tells the parser to continue to the next statement.

**repository**

A repository is a database of program objects that comprise the model for a [workspace](#).

**Repository Browser**

The [Asset Transformation Workbench \(ATW\)](#) Repository Browser displays the contents of the current [workspace](#).

**Resource Retriever**

The Resource Retriever tool lets you identify and restore missing [CICS](#) file connectors and transactions in file control tables ([FCT](#)) and program control tables ([PCT](#)) for [Cobol](#) and [PL/I](#) programs.

**Rules pane**

The [HyperView](#) Rules pane lets you create [business rules](#) from code segments extracted manually from source or autodetected. You can also create business rules from candidates listed in [Clipper](#).

**schema**

See [database schema](#).

**scope**

The scope of a diagram determines the objects and [relationships](#) it displays. See [Diagrammer](#).

**seed field**

A seed field is the object of a [Change Analyzer](#) search for the class of data items that need to be changed.

### Source pane

The [HyperView](#) Source pane displays view-only source code for the selected file and included files.

### SQL

SQL stands for Structured Query Language, a standard language for relational database operations

### structure-based component extraction

Structure-based component extraction is a type of [component](#) extraction that builds a component from a range of inline code — [Cobol](#) paragraphs, for example.

### synonym

A synonym is a data field whose value is related to the value of the matched [seed field](#) — a field whose value is assigned by a MOVE or REDEFINE statement, for example.

### system program

A system program is a generic program — a mainframe sort utility, for example — provided by the underlying system and used in unmodified form in the legacy application.

### TIP

TIP stands for Transaction Processing, the Unisys real-time system for processing transactions under Exec control.

### token

In the [Application Namespace tool](#), a token is an element in a program identifier delimited by a hyphen (-) or underscore (\_). In the identifier WS01-CUST-FIELD, for example, there are three tokens: WS01, CUST, and FIELD.

### transaction

A transaction is a sequence of information exchange and related work (such as database updating) that is treated as a unit for the purposes of satisfying a request and for ensuring database integrity.

### Unknown Statements Resolution tool

The Unknown Statements Resolution tool resolves incorrect or unsupported statements in [Cobol](#) programs verified with the [relaxed parsing](#) option.

**User Interface tool**

The User Interface tool lets you analyze the interaction between legacy screens and program logic, and generate an [HTML](#) or [Java](#) GUI based on the interaction.

**VALTAB**

VALTAB stands for Validation Table, which contains the information the system needs to locate, load, and execute transaction programs. See also [TIP](#).

**VSAM**

VSAM stands for Virtual Storage Access Method, an IBM program that controls communication and the flow of data in a Systems Network Architecture network.

**WebGen**

WebGen generates HTML views of the repositories on your workstation. You can publish the views to [Profiler](#), where they can be accessed by any member of your organization with a browser.

**workspace**

A workspace is a named container for an application or a portion of an application. Workspaces can be divided into [projects](#).

**XML**

XML stands for Extensible Markup Language, a specification for creating common information formats.





---

# Bibliography

- *IBM Asset Transformation Workbench v1.1 Getting Started (SC31-6877-00)*
- *IBM Asset Transformation Workbench v1.1 Preparing Projects (SC31-6879-00)*
- *IBM Asset Transformation Workbench v1.1 Analyzing Projects (SC31-6880-00)*
- *IBM Asset Transformation Workbench v1.1 Analyzing Programs (SC31-6878-00)*
- *IBM Asset Transformation Workbench v1.1 Profiling Projects (SC31-6881-00)*
- *IBM Asset Transformation Workbench v1.1 Creating Components (SC31-6876-00)*
- *IBM Asset Transformation Workbench v1.1 Parser Reference (SC31-6882-00)*
- *IBM Asset Transformation Workbench v1.1 Architecture Reference (SC31-6898-00)*



---

# Notices

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

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

IBM Director of Licensing

IBM Corporation

North Castle Drive

Armonk, NY 10504-1785

U.S.A.

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

IBM World Trade Asia Corporation

Licensing

2-31 Roppongi 3-chome, Minato-ku

Tokyo 106, 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:

IBM Corporation  
P.O. Box 12195, Dept. TL3B/B503/B313  
3039 Cornwallis Rd.  
Research Triangle Park, NC 27709-2195  
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 measurement 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.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only. 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.

---

## Trademarks

The following terms are trademarks of the IBM Corporation or its subsidiaries in the United States or other countries or both:

*Table 1. Trademarks*

IBM	MVS
AS400	CICS
IMS	DB/2
Database 2	OS/390
S/390	z/OS

The following terms are trademarks of other companies:

Java and JavaScript are registered trademarks and Sun Solaris and Solaris are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

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

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



# Index

## A

- advanced search facility
  - defining search criteria 3-9
  - executing advanced searches 3-8
  - performing advanced searches 3-1
  - searching for duplicated code 3-6
- Animator, using 6-11
- annotations 7-25
- audit property 7-15
- autodetect methods 7-6

## B

- Bird's Eye pane 1-12
- business area property 7-17
- business function property 7-18
- business rules
  - assigning property values to multiple rules 7-22

- assigning segments 7-13
- autodetecting 7-6
- autodetecting I/O data elements 7-21
- deleting 7-13
- deleting segments 7-13
- editing rule properties 7-14
- exporting reports 7-26
- extracting in Clipper pane 4-1
- extracting manually 7-3
- filtering the display 7-23
- flagging segments in source 7-13
- generating reports 7-25
- overview 7-1
- printing reports 4-10, 7-26
- renaming 7-13
- selecting segments 7-3
- validating 7-11
- viewing segments in source 7-13
- Business Rules Report 7-25

## C

- Callie pane
  - choosing the diagram view 6-3
  - overview 6-2
  - reconstructing diagrams 6-4
  - setting project options 6-5
  - setting user preferences 6-4
- category property 7-14
- change magnitude, specifying 2-4
- Clipper pane
  - creating a list 4-1
  - creating or adding to a project 4-8
  - extracting business rules 4-1
  - generating reports 4-9
  - injecting events 4-7
  - showing impacts 4-7
  - working with categories 4-6
  - working with lists 4-6
- code location property 7-16
- Components pane 1-9
- Context pane 2-7
- Coverage Report 7-25

## D

- Data Flow pane 1-10
- Data View pane 1-10
- description property 7-19, 7-20
- Details Report 4-9
- duplicated code, searching for 3-6

## E

- Execution Path pane, using 6-8
- extraction status property 7-16

## F

- Flow to Field method 7-7
- Flowchart pane, using 6-9

- function property 7-15

## H

- Handle Not Found method 7-8
- HyperView 1-3
  - advanced search facility 3-1
  - Animator 6-11
  - Callie pane 6-2
  - Clipper pane 4-1
  - Context pane 2-7
  - Execution Path pane 6-8
  - Flowchart pane 6-9
  - Impact pane 5-13
  - Impact Report tool 5-2
  - List Browser window 1-18
  - overview 1-1
  - parse trees 1-1
  - Properties window 1-15
  - Rules pane 7-1
  - Source pane 2-1
  - starting 1-3

## I

- I/O Data Elements method 7-21
- I/O data elements property 7-21
- I/O Rules method 7-8
- Impact pane
  - generating an impact trace 5-13
  - relationships 5-8
  - setting options 5-15
- Impact Report tool
  - creating or adding to a project 5-11
  - exporting reports 5-12
  - generating an impact trace 5-3
  - relationships 5-8
  - report depths 5-2
  - setting options 5-6



**L**

List Browser window  
 creating a list 1-20  
 deleting a list 1-20  
 exporting a list 1-20  
 opening a list 1-18  
 overview 1-18

**M**

message property 7-18  
 Metrics Report 4-9  
 Model Reference pane 1-12  
 most representative rule 7-14  
 MOVES to Screen Message method 7-7

**P**

PL/I Call Diagram pane 1-11  
 Process Outline Report 7-25  
 process outlines 7-25  
 program control flows  
   analyzing with Callie 6-2  
   analyzing with the Animator 6-11  
   analyzing with the Execution Path  
     pane 6-8  
   analyzing with the Flowchart pane 6-9  
   overview 6-1  
 project  
   creating or adding to in Clipper 4-8  
   creating or adding to in Impact Report  
     5-11  
 Properties window 1-15

**R**

Rule Programs Report 7-25  
 Rules pane  
   assigning property values to multiple  
     rules 7-22  
   assigning segments 7-13

autodetecting business rules 7-6  
 deleting rules 7-13  
 deleting segments 7-13  
 editing rule properties 7-14  
 exporting business rule reports 7-26  
 extracting business rules manually 7-3  
 filtering business rules display 7-23  
 flagging segments 7-13  
 generating business rule reports 7-25  
 overview 7-1  
 printing business rule reports 4-10, 7-  
   26  
 renaming rules 7-13  
 selecting segments 7-3  
 validating rules 7-11  
 viewing segments in source 7-13

**S**

Screen Validation method 7-8  
 searching for code  
   advanced search facility 3-1  
   simple search facility 2-4  
 segments  
   assigning to business rules 7-13  
   deleting 7-13  
   flagging in source 7-13  
   overview 7-1  
   selecting 7-3  
   viewing in source 7-13  
 signatures 3-6  
 Source pane  
   collapsing paragraphs or subprograms  
     2-6  
   flagging segments 7-13  
   marking list items 1-19  
   navigating to multiple occurrences of  
     a construct 2-3  
   navigating to related constructs 2-3

## Index-4

- overview 2-1
- searching for code 2-4
- selecting and copying code 2-2
- setting options 2-5
- showing source code boundaries 2-6
- specifying the change magnitude 2-4

## **T**

- Test on Field method 7-9
- transition property 7-17





Product Number: 5724-L54

SC31-6878-00



(1P) P/N:5724-L54

