

AllFusion™ Endeavor® Change Manager

Interface for IBM Tivoli
Information/Management
Administration Guide
4.0



Computer Associates®

SP1
ENINM400

This documentation and related computer software program (hereinafter referred to as the "Documentation") is for the end user's informational purposes only and is subject to change or withdrawal by Computer Associates International, Inc. ("CA") at any time.

This documentation may not be copied, transferred, reproduced, disclosed or duplicated, in whole or in part, without the prior written consent of CA. This documentation is proprietary information of CA and protected by the copyright laws of the United States and international treaties.

Notwithstanding the foregoing, licensed users may print a reasonable number of copies of this documentation for their own internal use, provided that all CA copyright notices and legends are affixed to each reproduced copy. Only authorized employees, consultants, or agents of the user who are bound by the confidentiality provisions of the license for the software are permitted to have access to such copies.

This right to print copies is limited to the period during which the license for the product remains in full force and effect. Should the license terminate for any reason, it shall be the user's responsibility to return to CA the reproduced copies or to certify to CA that same have been destroyed.

To the extent permitted by applicable law, CA provides this documentation "as is" without warranty of any kind, including without limitation, any implied warranties of merchantability, fitness for a particular purpose or noninfringement. In no event will CA be liable to the end user or any third party for any loss or damage, direct or indirect, from the use of this documentation, including without limitation, lost profits, business interruption, goodwill, or lost data, even if CA is expressly advised of such loss or damage.

The use of any product referenced in this documentation and this documentation is governed by the end user's applicable license agreement.

The manufacturer of this documentation is Computer Associates International, Inc.

Provided with "Restricted Rights" as set forth in 48 C.F.R. Section 12.212, 48 C.F.R. Sections 52.227-19(c)(1) and (2) or DFARS Section 252.227-7013(c)(1)(ii) or applicable successor provisions.

First Edition, June 2003

© 2003 Computer Associates International, Inc.
All rights reserved.

All trademarks, trade names, service marks, and logos referenced herein belong to their respective companies.

Contents

Chapter 1. Interface Overview	1-1
1.1 Resources Needed to Implement the Interface	1-2
1.2 Interface Functionality	1-3
1.3 Interface Operation	1-4
1.4 The Batch Utility	1-5
1.5 Package Level of the Interface	1-7
1.5.1 Information Available from the Package Level of the Interface	1-8
1.5.2 Uses of the Package Level of the Interface	1-8
1.6 The Action Level of the Interface	1-9
1.6.1 Information Available from the Action Level of the Interface	1-9
Chapter 2. Installing the Interface	2-1
2.1 General Installation Information	2-2
2.1.1 Four Installation Phases	2-2
2.1.2 Installation at Customized Info/Man Sites	2-2
2.1.3 Installation at IIF Sites	2-2
2.1.4 Who Should Perform the Installation?	2-2
2.2 Phase 1. Installing Basic Info/Man	2-3
2.2.1 Step 1. Define Users to Info/Man Classes	2-3
2.2.2 Step 2. Prepare a Session Parameter Member	2-3
2.2.3 Step 3. Put Information/Management Load Library Data Set in LINKLIST	2-4
2.2.4 Step 4. Move BLGISPFDD into ISPPLIB	2-4
2.2.5 Phase 1 Summary	2-4
2.3 Phase 2. Modifying Basic Info/Man for the Interface	2-5
2.3.1 Step 1. Copy Interface Report Shells into the RFT Data Set	2-5
2.3.2 Step 2. Copy Modified BLG6CORQ into the RPANEL0 Data Set	2-6
2.3.3 Step 3. Copy Dictionary Update	2-7
2.3.4 Step 4. Create the Standard IBM PIDT and PIPT Tables	2-8
2.3.5 Step 5. Build a Rule Set	2-9
2.3.5.1 BC1JEI00 JCL	2-10
2.3.6 Step 6. Copy the BLGSESxx Module to STEPLIB	2-11
2.3.7 Summary	2-13
2.4 Phase 3. Connecting the Interface	2-14
2.4.1 Step 1. Modify the @EINFO Macro	2-14
2.4.2 Step 2. Assemble and Link BC1TEI90	2-16
2.4.3 Step 3. Reassemble C1DEFLT5 with the Info/Man Password	2-19
2.4.4 Step 4. Make Sure Required Panels Are in ISPPLIB	2-19
2.4.5 Step 5. Make Sure INFO01, PKEX21, and PKMR02 Are in ISPMLIB	2-20

2.4.6	Step 6. Edit C1SB3000 Skeleton JCL	2-20
2.5	Phase 4. Verifying the Installation	2-23
2.5.1	Required PIDT and PIPT Tables	2-24
2.5.1.1	Change Record PIDT and PIPT Tables	2-25
2.5.1.2	Change Activity PIDT and PIPT Tables	2-25
2.5.1.3	Problem Record PIDT and PIPT Tables	2-26
Chapter 3.	Setting Up the Endeavor/INFO Table — the Batch Utility	3-1
3.1	The Endeavor Side of the Interface	3-2
3.2	Checklist for Implementing the Endeavor Side	3-3
3.3	JCL for the Batch Utility	3-4
3.4	Endeavor/INFO Table Syntax	3-6
3.4.1	Table Syntax Components	3-7
3.4.2	Syntax Diagram Conventions	3-7
3.5	TABLE Statement	3-8
3.6	Control Block	3-9
3.6.1	Control Block Syntax	3-9
3.6.1.1	CTRL Statement	3-9
3.6.1.2	Info/Man Action	3-9
3.6.1.3	Endeavor Entity	3-10
3.6.1.4	PIDT Name	3-10
3.6.1.5	ECTRL Statement	3-10
3.6.2	Possible Requests	3-11
3.6.3	Example 1	3-12
3.6.4	Example 2	3-12
3.7	Use Blocks	3-13
3.7.1	USE Statement	3-14
3.7.1.1	USE	3-14
3.7.1.2	Action	3-14
3.7.1.3	STANDALONE	3-15
3.7.1.4	PROBLEM	3-15
3.7.1.5	Example 1	3-15
3.7.1.6	Example 2	3-15
3.7.2	NOTFOUND Statement	3-16
3.7.3	FOUND Statement	3-17
3.7.3.1	Example 1	3-17
3.7.3.2	Example 2	3-17
3.7.4	Label Blocks	3-18
3.7.5	EUSE Statement	3-18
3.7.5.1	Example 1	3-18
3.8	Label Block	3-19
3.8.1	LABEL Statement	3-19
3.8.1.1	Example	3-19
3.8.2	ELABEL Statement	3-20
3.8.3	CPASS/CFAIL Statements	3-20
3.8.3.1	Example of CFAIL Statement	3-20
3.8.3.2	Example of CPASS Statement	3-21
3.8.3.3	Example of CPASS/CFAIL Statements	3-21
3.8.4	TSP Statements	3-21
3.8.4.1	Example 1	3-22
3.8.4.2	Example 2	3-22

3.8.5	Comments	3-23
3.8.5.1	Example of Single-Line Comments	3-23
3.8.5.2	Example of Multiple-Line Comments	3-23
3.8.5.3	Example of Incorrectly-Coded Comment	3-23
3.8.6	RC Statements	3-24
3.8.6.1	Examples	3-24
3.8.7	MSG Statements	3-25
3.8.7.1	Example	3-25
3.9	Criteria Block	3-26
3.9.1.1	Example 1	3-27
3.9.1.2	Example 2	3-27
3.9.1.3	Example 3	3-28
3.10	Populate Block	3-29
3.10.1.1	Example 1	3-30
3.10.1.2	Example 2	3-30
3.11	FIELD Statements	3-31
3.11.1	FIELD Statement Syntax	3-31
3.11.2	Field Panel (Pname) and Index (S-Word)	3-31
3.11.3	Length	3-32
3.11.4	Value Clause--Criteria Block	3-32
3.11.5	Value Clause--Populate Block	3-33
3.11.6	IFYES/IFNO Statement	3-33
3.11.7	TEXT Clauses	3-33
3.11.8	Example 1. FIELD Statement in a Criteria Block--Info/Man Field	3-34
3.11.9	Example 2. FIELD Statement in a Criteria Block--Endevor Field	3-34
3.11.10	Example 3. FIELD Statement with Equates USE (CREATE, ACTIVITY)	3-35
3.11.11	Example 4. FIELD Statement with Text	3-35
3.12	Endevor Control Block Availability	3-36
3.12.1	Control Blocks for Stand-alone Actions	3-36
3.12.2	Available \$ENVDS Fields	3-37
3.12.3	Available \$ELMDS Fields	3-38
3.12.4	Available \$FILDS Fields	3-43
3.12.5	Available \$ECBDS Fields	3-44
3.12.6	Available \$REQDS Fields	3-45
3.12.7	Control Blocks for Packages and Package Actions	3-46
3.12.8	\$PREQPDS	3-48
3.12.9	\$PHDRDS	3-48
3.12.10	\$PACTREQ	3-49
3.13	Interface Syntax Batch Utility Output Report	3-53
3.13.1	Utility Syntax Report	3-53
3.13.2	Utility Error Report	3-54
Chapter 4. Implementing the Info/Man Side of the Interface		4-1
4.1	Info/Man API Basics	4-2
4.2	Checklist for Setting Up the Info/Man Side	4-3
4.3	Building Info/Man PIDT Tables	4-4
4.3.1	Inquiry PIDT Required Fields	4-5
4.3.2	Retrieve PIDT Required Fields	4-6
4.3.3	Create PIDT Required Fields	4-7

4.3.4 Update PIDT Required Fields	4-8
Chapter 5. Stand-Alone Action Table Syntax Example	5-1
5.1 Using the Syntax Template	5-2
5.1.1 IIF Syntax Templates for Stand-Alone Actions	5-2
5.2 Sample Syntax	5-3
Chapter 6. Package and Package Action Table Syntax Example	6-1
6.1 Syntax Example	6-2
Chapter 7. Displaying Info/Man Information	7-1
7.1 Displaying Information Stored in Info/Man	7-2
7.2 Listing and Displaying CCID Information	7-3
7.2.1 CCID List Panel	7-4
7.2.2 Action Prompt Panel	7-4
7.2.3 CCID Correlation Panel	7-5
7.3 Displaying Package Information	7-6
Chapter 8. Interface Reports	8-1
8.1 About the Interface Reports	8-2
8.2 Running the Reports	8-3
8.2.1 Running the Reports in Native Info/Man	8-3
8.2.2 Running the Reports in Batch	8-3
8.2.2.1 Contention for the ISPF Profile Data Set	8-3
8.2.2.2 Setting Defaults in Info/Man for Output Destination and Class for Reports	8-4
8.3 Customizing the Reports	8-6
8.4 Action Change Record Summary	8-7
8.5 Action Change Record Detail	8-8
8.6 Action Problem Record Summary/Detail	8-9
8.6.1 Summary Report	8-9
8.6.2 Detail Report	8-9
8.7 Package Parent Record Summary	8-11
8.8 Package/Activity Record Summary	8-12
8.8.1 Package Information	8-12
8.8.2 Activity Information	8-13
8.9 Package/Activity Record Detail--Package Detail	8-14
8.10 Package/Activity Detail--Activity Detail	8-15
Appendix A. Information for Advanced Users	A-1
A.1 Modified Assisted Entry Panel (BLG6CORQ)	A-2
A.1.1 PMF Report for BLG6CORQ Panel	A-2
A.1.1.1 PMF Report for BLG6CORQ Panel page 2	A-3
A.2 Customized Dictionary Entry	A-4
A.2.1 The PWORD XREF Report	A-4
A.3 Reserved Info/Man Fields	A-5
A.3.1 Inquiry PIDT Required Fields	A-5
A.3.2 Retrieve PIDT Required Fields	A-6
A.3.3 Create PIDT Required Fields	A-7
A.3.4 Update PIDT Required Fields	A-8

Appendix B. Installing the Interface Under IIF	B-1
B.1 General Installation Information for IIF Sites	B-2
B.1.1 Four Installation Phases	B-2
B.1.2 Who Should Perform the Installation?	B-2
B.2 Phase 1. Installing Basic Info/Man	B-3
B.2.1 Step 1. Define Users to Info/Man Classes	B-3
B.2.2 Step 2. Prepare a Session Parameter Member	B-3
B.2.3 Step 3. Put Information/Management Load Data Set in LINKLIST	B-4
B.2.4 Step 4. Move BLGISPFDD into ISPPLIB	B-4
B.2.5 Phase 1 Summary	B-4
B.3 Phase 2. Modifying Basic Info/Man for the Interface	B-5
B.3.1 Step 1. Copy Interface Report Shells into the RFT Data Set	B-6
B.3.2 Step 2. Copy Modified Panels into the RPANEL0 Data Set	B-7
B.3.3 Step 3. Copy Dictionary Update	B-8
B.3.4 Step 4. Create Computer Associates PIDT and PIPT Tables	B-9
B.3.5 Step 5. Build a Rule Set	B-10
B.3.5.1 BC1JEX00 JCL	B-11
B.3.6 Step 6. Copy the BLGSESxx Module to STEPLIB	B-13
B.3.7 Summary	B-14
B.4 Phase 3. Connecting the Interface	B-15
B.4.1 Step 1. Modify the @EINFO Macro	B-15
B.4.2 Step 2. Assemble and Link BC1TEI90	B-17
B.4.3 Step 3. Reassemble C1DEFLTS with the Info/Man Password	B-19
B.4.4 Step 4. Make Sure Required Panels Are in ISPPLIB	B-20
B.4.5 Step 5. Make Sure INFO01, PKEX21, and PKMR02 Are in ISPMLIB	B-20
B.4.6 Step 6. Edit C1SB3000 Skeleton JCL	B-21
B.5 Phase 4. Verifying the Installation	B-22
B.5.1 Required PIDT and PIPT Tables	B-24
B.5.1.1 Change Record PIDT and PIPT Tables	B-25
B.5.1.2 Change Activity PIDT and PIPT Tables	B-25
B.5.1.3 Problem Record PIDT and PIPT Tables	B-26

Chapter 1. Interface Overview

1.1 Resources Needed to Implement the Interface

Implementing the AllFusion™ Endeavor® Change Manager Interface for IBM Information/Management (Info/Man Interface) to the Information/Family requires that you make decisions about which AllFusion Endeavor Change Manager (hereafter referred to as Endeavor) information you want to track in the Information/Family, and where you want to store that information in the Information/Family. Therefore, to implement the interface most effectively, you should have the following resources available:

- People familiar with how Endeavor is used at your site, and in particular with Endeavor packages and exits.
- People familiar with Information/Family policies and operations at your site.

Information is available from the following publications:

- *Exits Guide.*
- *Packages Guide.*
- *IBM Tivoli Information/Management Application Program Interface Guide* SC34-4592-00.

Note: Hereafter, this manual refers to the interface as the Info/Man interface, and to the Information/Family as Info/Man.

1.2 Interface Functionality

The Endeavor Info/Man interface offers a flexible mechanism for implementing Information/Family (Info/Man) policies as they relate to Endeavor, for controlling Endeavor actions based on Info/Man information, and for logging Endeavor information in Info/Man.

The interface works with both native Info/man and the Info/Man Interactive Facility (IIF).

In addition to managing the initialization and termination of Info/Man sessions under Endeavor, the interface implements package-level and action-level interaction between Endeavor and Info/Man. This allows you to:

- Use the Info/Man approval process for moving packages into production, rather than the Endeavor process.
- Use the Info/Man record as an auditing tool.
- Display package information from Info/Man during an Endeavor session.
- Produce lists of available CCIDs when requesting stand-alone actions in foreground or batch SCL generation.

The interface does this by communicating directly from Endeavor exit points with the Info/Man API. The communication is accomplished through a user-defined rule set, called the E/INFO table, created when the interface is implemented. You create the rule set with a batch utility provided with the interface.

Before the existence of the Info/Man API, Terminal Simulation Programs (TSPs) were the only way to get to Info/Man. The batch utility offers another, flexible way to build a custom rule set for implementing Info/Man policies in Endeavor, and getting data requirements and processing logic from Endeavor to Info/Man.

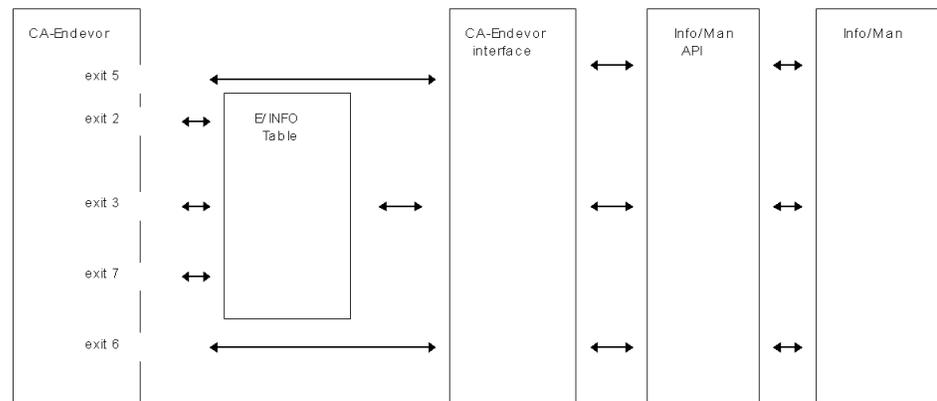
The batch utility allows this rule set to be independent of Info/Man invocation. Moreover, the utility does not require extensive experience with either TSPs or assembler programming.

1.3 Interface Operation

Endevor provides a full set of exit points (before-exit and after-exit) for actions and packages and uses control blocks to provide information to user programs written for these exit points. Info/Man provides an API that uses data structures called PIDT tables to provide logical views of Info/Man database records. The interface operates between the Endevor exit facility and the Info/Man API.

The interface communicates directly with the Info/Man API at exit 5 (initialization) and exit 6 (termination). For exits 2 (before-action), 3 (after-action), and 7 (package functions) the E/INFO table, the load module built using the batch utility, provides a bridge between the Endevor exit facility and the Info/Man API. The rule set contained in the E/INFO table determines how the interface acts.

The diagram below summarizes interface operation.

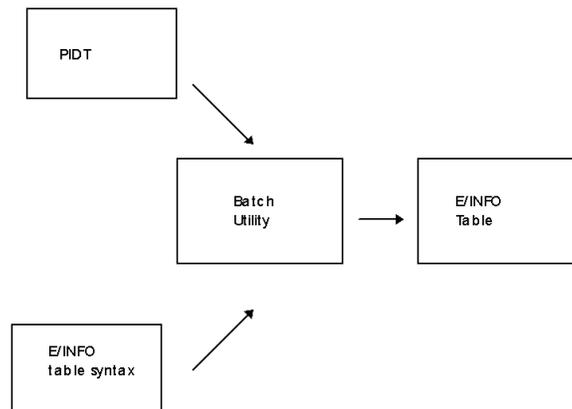


1.4 The Batch Utility

You use the batch utility (program C1BMEI00) to build an assembled rule set to drive the package and action levels of the interface. The utility provides a language that allows you to define this rule set by:

- Identifying Info/Man actions to be performed (Inquiry, Retrieve, Create, Update), and the PIDT table needed for each function.
- Specifying the exit points where the actions are to be performed.
- Establishing criteria on the Endeavor or the Info/Man side for performing an action.
- Specifying the Endeavor information to be recorded in Info/Man, and the Info/Man fields where the information is to be recorded.

The utility assembles the definition syntax into a load module, referred to in this manual as the E/INFO table. The diagram below illustrates this process.



The E/INFO table is loaded at Endeavor initialization, and is independent of Info/Man invocation. The rule set it contains determines how the interface uses the Info/Man API.

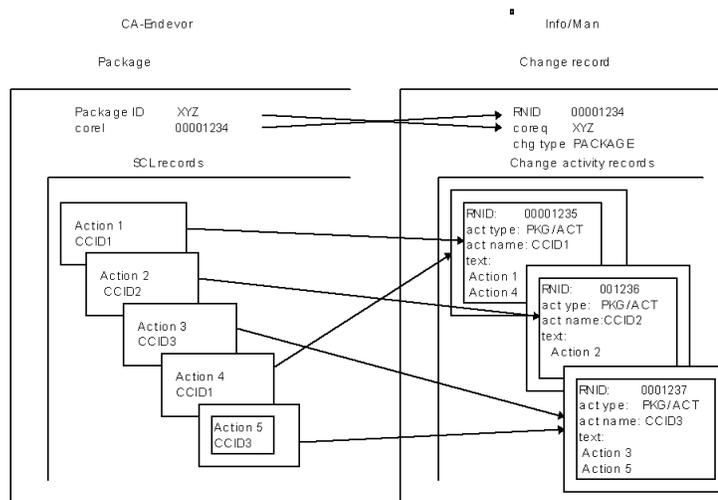
When the table is invoked, Endeavor builds a request for the Info/Man API using the information in the table. The Info/Man API executes the requested functions against the Info/Man database.

1.5 Package Level of the Interface

The package level of the interface allows the capture of both package life cycle information and information about the individual actions in the package. The interface establishes the following relationships:

- The Endeavor package is cross-referenced to an Info/Man change record, with the package ID stored in the COREQUISITE field of the Info/Man record, and the Info/Man RNID stored in the CORRELATION field of the Endeavor package record.
- For each unique CCID in the package action SCL records, the interface maintains a change activity record in Info/Man, with the Endeavor CCID stored in the ACTION NAME field in the Info/Man record. Information about each action associated with the CCID can be stored as freeform text in that activity record. The freeform text field is useful for this purpose because new information is always appended to existing information in the field.

The diagram below summarizes the package level of the interface.



1.5.1 Information Available from the Package Level of the Interface

There are two ways of viewing package information stored in Info/Man:

- In reports. These are documented in Chapter 8, "Interface Reports" in this manual.
- In foreground you can view package correlation data from the Package Display panel.

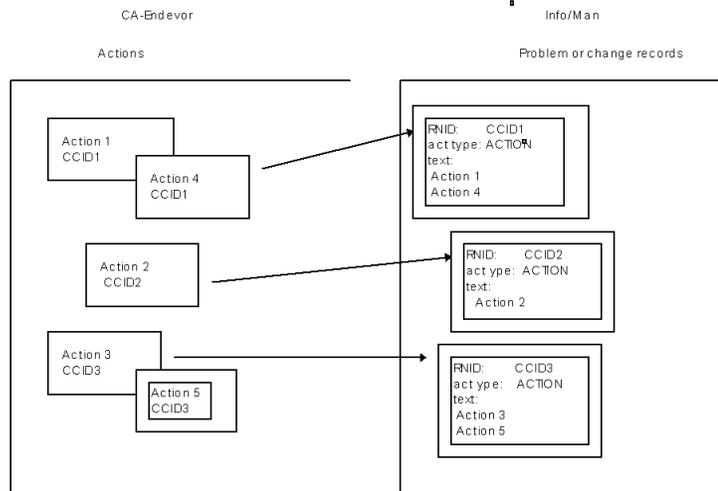
1.5.2 Uses of the Package Level of the Interface

The interface allows sites that use Info/Man as the mechanism for approving packages for production turnover to use the Info/Man approval process instead of the Endeavor approval process for Endeavor packages.

The interface also provides an improved audit trail for packages, and better reporting on packages than is currently available from Endeavor.

1.6 The Action Level of the Interface

Sites that do not use packages, or prefer to use Info/Man Problem records, can utilize the action level of the interface. For standalone Endeavor actions, the interface creates either a Problem or a Change record in Info/Man for each unique CCID. The diagram below shows this relationship.



1.6.1 Information Available from the Action Level of the Interface

When using the action level of the Info/Man interface, the user is allowed to wildcard the CCID field in Endeavor when requesting the following stand-alone actions in foreground and batch: ADD/UPDATE, MOVE, RETRIEVE, GENERATE, DELETE, TRANSFER, and ARCHIVE.

When a CCID value is fully specified, a list is **not** produced. Processing proceeds normally, including processing specified by the interface.

Note: Wildcarding CCIDs is not allowed when building SCL in packages.

Chapter 2. Installing the Interface

2.1 General Installation Information

The target audience for this chapter are those people who have a strong understanding of Info/Man and a general knowledge of Endeavor.

2.1.1 Four Installation Phases

The installation of the interface falls into four phases:

- Phase 1: Installing basic Info/Man.
- Phase 2: Modifying basic Info/Man to support the interface.
- Phase 3: Tying the Endeavor and Info/Man sides together.
- Phase 4: Verifying the installation

2.1.2 Installation at Customized Info/Man Sites

If your Info/Man system has already been modified, the order of installation is as follows:

1. Obtain the data set names from Phase 1.
2. Refer to Appendix A, "Information for Advanced Users," for considerations relevant to customized Info/Man shops or advanced users. The person in charge of customizing the Info/Man product needs to review the interface requirements in this appendix.
3. Perform the tasks listed in Phase 3.

2.1.3 Installation at IIF Sites

If you use the Info/Man Integrated Facility (IIF), see Appendix B, "Installing the Interface," for installation instructions.

2.1.4 Who Should Perform the Installation?

The best combination of people to install the interface are the site Info/Man administrator and the site Endeavor administrator working together.

2.2 Phase 1. Installing Basic Info/Man

The requirements in this section are basic Info/Man installation requirements that exist whether or not the interface is installed.

Note: The Endeavor--Info/Man interface requires that the IBM Information/Management product, version 4.2 or higher, is installed at your site.

Info/Man setup involves the following.

Step	Action
1	Define all interface users to a class in Info/Man.
2	Make sure the Information/Management Load Library data set is in LINKLST, or that you have the data set name available.
3	Prepare a session parameter (BLGSESxx) module to reflect the database, data dictionary, read panel, and RFT data sets that the interface uses when accessing the data base via the API.
4	Make sure that BLGISPFD, an Info/Man supplied ISPF panel, has been moved into your ISPLIB concatenation.

These steps are discussed next.

2.2.1 Step 1. Define Users to Info/Man Classes

Each person authorized to use the interface must be defined to a class in Info/Man. See the Info/Man documentation for instructions.

2.2.2 Step 2. Prepare a Session Parameter Member

Info/Man requires a BLGSESxx module that points to the database, data dictionary, read panels, and RFT data set that the Interface uses when accessing the database via the API or native Info/Man.

This session parameter is well documented in the Info/Man install manuals. It is required of standard Info/Man. The naming standard is BLGSESxx where xx is the user's choice. This suffix is also required for an Endeavor start-up block assembly. Note that the same BLGSESxx member can be used for the interface as well as standard access to Info/Man.

The suffix used to name the actual BLGSESxx module is required in Phase 3. The rest of this install requires the data set names from the BLGSESxx module.

Note: The interface installation includes one modified Assisted Entry panel, which must be copied into your read panel data set concatenation. You can either overlay the IBM version of this data set, or provide a new read panel data set ahead of the standard IBM data set. Either way, the result is that the RPANEL0 label points to the data set used in the install.

2.2.3 Step 3. Put Information/Management Load Library Data Set in LINKLIST

The Info/Man load data set is used to install the interface.

Sometimes this library is part of LINKLIST, and sometimes it is not. If this data set is not in LINKLIST, the person implementing the interface needs to know its name, and must put it into a STEPLIB in the concatenation.

2.2.4 Step 4. Move BLGISPF into ISPLIB

Make sure that BLGISPF, an Info/Man-supplied ISPF panel, has been moved into your ISPLIB concatenation. See your Info/Man administrator if you have any questions about this step.

2.2.5 Phase 1 Summary

In this phase of the implementation you have built a vanilla version of Info/Man. At this point you should have installed Info/Man on your system, assigned users to an Info/Man class, and defined a BLGSESxx session parameter module. For future reference, record the following:

Class name:

BLGSESxx:

Within BLGSESxx, you have defined:

Dictionary DD name:

RFTDS DS name:

RPANEL0 name:

INFOMAN LOAD DS name:

2.3 Phase 2. Modifying Basic Info/Man for the Interface

In this phase, you customize the base Info/Man system set up in Phase 1 by performing the following steps:

Step	Action	JCL used
1	Copy interface report shells to the RFT data set identified in Phase 1.	BC1JEI20
2	Copy Endeavor version of BLG6CORQ into the RPANEL0 data set identified in Phase 1.	BC1JEI15
3	Install the dictionary update.	BC1JEI30
4	Create the standard IBM PIDT and PIPT tables.	BC1JEI45
5	Build a rule set for driving the API using the interface batch utility.	BC1JEI00
6	Assemble or copy the BLGSESxx module into your STEPLIB data set.	BC1JEI05

These steps are discussed next.

2.3.1 Step 1. Copy Interface Report Shells into the RFT Data Set

Copy the interface report shells (RFTs) to your RFT data set. Do this using the BC1JEI20 member in `iprfx.igual.JCLLIB`. Before submitting this JCL, provide a job card, confirm that `iprfx.igual` and `uprfx.igual` are the correct data set qualifiers, and confirm that `tdisk` is a valid unit name.

BC1JEI20 JCL is shown next.

```

/* ( COPY JOBCARD )
/******
/*
/* BC1JEI20 - JCL TO COPY THE ENDEVER INTERFACE REPORT SHELLS      *
/* INTO THE USERS RFTDS DATASET.                                  *
/*                                                                *
/* USER NEEDS TO KNOW THE NAME OF THE RFTDS DATASET POINTED      *
/* TO IN THE BLGSESXX INFO/MAN PARM MEMBER.                       *
/*                                                                *
/******
//STEP1 EXEC PGM=IEBCOPY
//SYSPRINT DD SYSOUT=*
//SYSUT3 DD UNIT=tdisk,SPACE=(TRK,(1))
//SYSUT4 DD UNIT=tdisk,SPACE=(TRK,(1))
//FROM DD DSN=iprfx.iqua1.SOURCE,DISP=SHR
//TO DD DSN=uprfx.uqua1.SBLMFMT,DISP=SHR <==== RFTDS DATASET
//SYSIN DD *
COPY INDD=((FROM,R)),OUTDD=TO
SELECT MEMBER=(EINTACH0)
SELECT MEMBER=(EINTACH1)
SELECT MEMBER=(EINTAPB0)
SELECT MEMBER=(EINTAPB1)
SELECT MEMBER=(EINTPKG0)
SELECT MEMBER=(EINTPKG1)
SELECT MEMBER=(EINTPKG2)
/*

```

This job copies the following report shells:

This member	Is the RFT shell for this report
EINTACH0	Action Change Record Summary
EINTACH1	Action Change Record Detail
EINTAPB0	Action Problem Record Summary
EINTAPB1	Action Problem Record Detail
EINTPKG0	Package Parent Record Summary
EINTPKG1	Package/Activity Record Summary
EINTPKG2	Package/Activity Record Detail

2.3.2 Step 2. Copy Modified BLG6CORQ into the RPANEL0 Data Set

The interface requires that a modified version of the Assisted Entry Panel (BLG6CORQ) be available in the RPANEL0 data set pointed to by the BLGSESxx session parameter member. The modification allows the BLG6CORQ panel to store the 16-character Endeavor package ID. This field has been flagged as a reserved field for the interface.

The modified panel is provided with the interface. Copy it into the RPANEL data set using the JCL in member BC1JEI15 in the iprfx.iqua1.JCLLIB data set.

This job uses the BLGUT6F utility to copy a PDS member that has been created from a Info/Man panel using BLGUT6.

This panel must reside above the IBM panel of the same name. The person who runs the job to copy the modified panel needs only the data set name. You recorded the RPANEL0 data set name in the worksheet at the end of Phase 1.

BC1JEI15 JCL is shown next. Before submitting this JCL, provide a job card, and confirm that iprfx.igual and uprfx.uqual are the correct data set qualifiers.

```
//(JOB CARD)
//*****
//*
//* JOB TO UPLOAD MEMBERS OF A PDS THAT HAVE BEEN CREATED FROM THE
//* BLGUT6F DOWN UTILITY PROGRAM THAT CONVERTS VSAM PANELS INTO
//* MEMBERS IN A PDS FOR DISTRIBUTION. (LRECL=80. UNUSABLE FORMAT)
//*
//* NOTE STEPLIB NOT NECESSARY IF INFO/MAN DATASET ARE IN LINKLST
//*
//*****
//BLGUT6 EXEC PGM=BLGUT6,REGION=2048K
//STEPLIB DD DSN=UPRFX.UQUAL.LOAD,DISP=SHR <== INFO/MAN LOAD LIB
//SYSPRINT DD SYSOUT=*
//BLGPDS DD DISP=SHR,DSN=IPRFX.IQUAL.SOURCE
//BLGPNLS DD DISP=SHR,DSN=UPRFX.UQUAL.ENDPNLS <== RPANEL0
//SYSIN DD *
INCLUDE
BLG6CORQ
/*
```

2.3.3 Step 3. Copy Dictionary Update

The modified Assisted Entry Panel (BLG6CORQ) has an associated dictionary entry. This entry must be copied into the dictionary identified in the Phase 1 Summary. Run JCL job BC1JEI30 to install this dictionary entry.

CAUTION:

Run this job only if you are installing basic Info/Man. If your site has customized Info/Man, refer to Appendix A, "Information for Advanced Users," for instructions on updating the dictionary entry.

BC1JEI30 JCL is shown next. Before submitting this JCL, provide a job card, and confirm that iprfx.igual and uprfx.uqual are the correct data set qualifiers.

```

/*  JOBCARD
//*****
/* COPY PDS MEMBER TO SEQUENTIAL DSN          *
//*****
//COPYMEM EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DSN=iprfx.iqual.SOURCE(INFODICT),DISP=SHR
//SYSUT2 DD DSN=&&TEMPDICT,DISP=(,PASS),
//      SPACE=(TRK,(1,2)),
//      DCB=(RECFM=VBA,LRECL=137,BLKSIZE=1370,DSORG=PS)
//SYSIN DD DUMMY
//*****
/* LOAD INFO DICTIONARY
//*****
//LOADMEM EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//COPY DD DSN=&&TEMPDICT,DISP=(OLD,DELETE)
//SYSIN DD *
REPRO OUTDATASET(uprfx.uqual.DICT)      +
      INFILE(COPY)                      +
      REPLACE
/*

```

2.3.4 Step 4. Create the Standard IBM PIDT and PIPT Tables

Create the PIDT and PIPT tables by executing InfoMan's BLGUT8 utility program. This step is necessary so that the updated dictionary entry is picked up. These tables are required by the interface to support batch and real-time interface execution, CCID list support and display services.

Use the BC1JEI45 member in iprfx.iqual.JCLLIB. This JCL is shown next. Before submitting this JCL, provide a job card and confirm that the iprfx.iqual and uprfx.uqual are the correct data set qualifiers.

```

/* ( COPY JOBCARD )
//STEP1 EXEC PGM=BLGUT8,REGION=2048K
//STEPLIB DD DSN=uprfx.uqual.LOAD,DISP=SHR <== INFO/MAN LOAD LIB
//BLGPNLS DD DSN=iprfx.iqual.SOURCE(BLG6CORQ),DISP=SHR
//      DD DSN=uprfx.uqual.SBLMPNLS,DISP=SHR <== INFO/MAN PANEL LIB
//BLGDICT DD DSN=uprfx.uqual.DICT,DISP=SHR <== DICT DATASET
//BLGRFT DD DSN=uprfx.uqual.SBLMFMT,DISP=SHR <== RFTDS
//SYSPRINT DD SYSOUT=*
//SYSIN DD DSN=uprfx.uqual.ABLMSAMP(TS0B06AS),DISP=SHR
//      DD DSN=uprfx.uqual.ABLMSAMP(TS0B06CS),DISP=SHR
//      DD DSN=uprfx.uqual.ABLMSAMP(TS0B06IS),DISP=SHR
//      DD DSN=uprfx.uqual.ABLMSAMP(TS0B06RS),DISP=SHR
//      DD DSN=uprfx.uqual.ABLMSAMP(TS0B06US),DISP=SHR
//      DD DSN=uprfx.uqual.ABLMSAMP(TS0B07CS),DISP=SHR
//      DD DSN=uprfx.uqual.ABLMSAMP(TS0B07IS),DISP=SHR
//      DD DSN=uprfx.uqual.ABLMSAMP(TS0B07RS),DISP=SHR
//      DD DSN=uprfx.uqual.ABLMSAMP(TS0B07US),DISP=SHR
//      DD DSN=uprfx.uqual.ABLMSAMP(TS0032CS),DISP=SHR
//      DD DSN=uprfx.uqual.ABLMSAMP(TS0032IS),DISP=SHR
//      DD DSN=uprfx.uqual.ABLMSAMP(TS0032RS),DISP=SHR
//      DD DSN=uprfx.uqual.ABLMSAMP(TS0032US),DISP=SHR

```

2.3.5 Step 5. Build a Rule Set

In addition to setting up Info/Man to support the interface, you need to build a rule set to drive the Info/Man API. You do this using the batch utility provided by the interface.

Before proceeding, review:

- Chapter 3, "Setting Up the Endeavor/INFO Table - the Batch Utility."
- The samples provided in iprfx.igual.SOURCE. There are three samples provided:

Sample	Description
EISCRIP1	Batch utility input for stand-alone actions, using Info/Man Change records.
EISCRIP2	Batch utility input for stand-alone actions, using Info/Man Problem records.
EISCRIP3	Batch utility input for package functions and package actions.

Next, create a rule set. It is advisable to keep it simple at first. Do this by taking one or two actions or package functions and writing utility syntax input for them. Then run the batch utility to create an initial rule set. The JCL for executing the utility to build the rule set is in member BC1JEI00 of iprfx.igual.JCLLIB.

A successful run of the utility produces a load module called by a name you specify. This load module name is used in Phase 3. Assemble this load module into a STEPLIB data set.

Remember:

- To change what you use the interface for is a simple run of the utility. It requires no further Info/Man or Endeavor work to enhance the rule set.
- If you point to any PIDT tables other than the defaults, verify that they reside in the RFTDS data set pointed to in your BLGSESxx member.

The JCL for executing the utility to build the rule set is in member BC1JEI00 of iprfx.igual.JCLLIB. See 2.3.5.1, "BC1JEI00 JCL" on page 2-10. Before submitting this JCL, provide a job card, confirm that iprfx.igual and uprfx.uqual are the correct data set qualifiers, confirm that tdisk is a valid unit name and, if necessary, change the size of the WORK1 or WORK2 data sets.

2.3.5.1 BC1JEI00 JCL

```
/**(JOB CARD)
/**-----*
/**
/** (C) 2002 COMPUTER ASSOCIATES INTERNATIONAL, INC.
/**
/** NAME: BC1JEI00
/**
/** THE PURPOSE OF THIS JOB TO RUN THE BATCH UTILITY.
/**
/** INPUT TO THIS JOB IS THE BATCH UTILITY SYNTAX. OUTPUT FROM
/** THIS IS THE E-PIDT TABLE. THIS TABLE NAME MUST BE INCLUDE IN
/** THE ASSEMBLY OF THE BC1TEI90 STARTUP BLOCK.
/**
/** NOTE: NDVRIN POINTS TO THE BATCH SYNTAX THE CUSTOMER WANTS TO
/** USE. THERE ARE THREE SAMPLE SCRIPTS DELIVERED WITH THE
/** PRODUCT.
/**
/** EISCRIP1 =====> USE FOR STAND ALONE ACTIONS - CHANGE RECORDS
/** EISCRIP2 =====> USE FOR STAND ALONE ACTIONS - PROBLEM RECORDS
/** EISCRIP3 =====> USE FOR PACKAGE LEVEL INTERFACE
/**
/** REPLACE THE NDVRIN MEMBER NAME IMBMR WITH THE MEMBER YOU CHOSE.*
/**-----*
```

```

//*****
//*   RUN UTILITY TO ASSEMBLE SYNTAX THAT CREATES E-PIDT LOADMOD   *
//*****
//BATCH   EXEC   PGM=NDVRC1,PARM=C1BMEI00
//STEPLIB DD DISP=SHR,DSN=UPRFX.UQUAL.AUTHLIB
//        DD DISP=SHR,DSN=IPRFX.IQUAL.AUTHLIB
//CONLIB  DD DISP=SHR,DSN=IPRFX.IQUAL.CONLIB
//NDVRIN  DD DISP=SHR,DSN=IPRFX.IQUAL.SOURCE(IMBMR)
//*
//*****
//*   THE ABLMSAMP LIBRARY IS A STANDARD LIBRARY PROVIDED WITH   *
//*   THE INSTALL OF INFO/MAN.                                     *
//*   THESE POINT TO THE IBM SUPPLIED PIDT SOURCE. IF YOU HAVE   *
//*   YOUR OWN PIDTS, MODIFY ACCORDINGLY. IF YOU ARE USING THE   *
//*   PROBLEM LEVEL INTERFACE, USE THE LINES THAT POINT TO THE   *
//*   MEMBERS THAT START WITH 'TS0032' OTHERWISE USE USE THE LINES *
//*   THAT POINT TO THE MEMBERS THAT START WITH 'TS0B06' AND     *
//*   'TS0B07'.                                                  *
//*****
//NDVRPIDT DD DSN=UPRFX.UQUAL.ABLMSAMP(TS0B06IS),DISP=SHR
//        DD DSN=UPRFX.UQUAL.ABLMSAMP(TS0B06RS),DISP=SHR
//        DD DSN=UPRFX.UQUAL.ABLMSAMP(TS0B06CS),DISP=SHR
//        DD DSN=UPRFX.UQUAL.ABLMSAMP(TS0B06US),DISP=SHR
//        DD DSN=UPRFX.UQUAL.ABLMSAMP(TS0B06AS),DISP=SHR
//*
//        DD DSN=UPRFX.UQUAL.ABLMSAMP(TS0B07IS),DISP=SHR
//        DD DSN=UPRFX.UQUAL.ABLMSAMP(TS0B07RS),DISP=SHR
//        DD DSN=UPRFX.UQUAL.ABLMSAMP(TS0B07CS),DISP=SHR
//        DD DSN=UPRFX.UQUAL.ABLMSAMP(TS0B07US),DISP=SHR
//*
//        DD DSN=UPRFX.UQUAL.ABLMSAMP(TS0032US),DISP=SHR
//        DD DSN=UPRFX.UQUAL.ABLMSAMP(TS0032CS),DISP=SHR
//        DD DSN=UPRFX.UQUAL.ABLMSAMP(TS0032IS),DISP=SHR
//        DD DSN=UPRFX.UQUAL.ABLMSAMP(TS0032RS),DISP=SHR
//*
//NDVRRPT1 DD SYSOUT=*
//NDVRRPT2 DD SYSOUT=*
//NDVRRPT3 DD SYSOUT=*
//*
//*SAMPLE JCL FOR BC1JEI00 NDVRWK1 AND NDVRWK2 SHOULD BE AS FOLLOWS
//*
//NDVRWK1  DD DSN=UPRFX.UQUAL.WORK1,DISP=(,PASS),
//        UNIT=TDISK,SPACE=(TRK,(10))
//NDVRWK2  DD DSN=UPRFX.UQUAL.WORK2,DISP=(,PASS),
//        UNIT=TDISK,SPACE=(TRK,(10))
//*
//SYSLMOD  DD DISP=SHR,DSN=UPRFX.UQUAL.AUTHLIB   <== E-PIDT STORED HERE
//SYSUT1   DD UNIT=TDISK,SPACE=(TRK,(1,1))
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*

```

2.3.6 Step 6. Copy the BLGSESxx Module to STEPLIB

Assemble or copy the BLGSESxx member into your STEPLIB data set. See iprfx.igual.JCLLIB(BC1JEI05) for sample JCL.

BC1JEI05 JCL is shown next. Before submitting this JCL, provide a job card, confirm that `iprfx.uqual` and `uprfx.uqual` are the correct data set qualifiers, confirm that `tdisk` is a valid unit name, and replace `xx` in the string `BLGSESxx` with a session identifier.

```

/* ( COPY JOBCARD )
/*****
/*
/* BC1JEI05 - BUILD INFORMATION/MANAGEMENT SESSION TABLE.          *
/* THIS JOB IS DESCRIBED IN IBM'S PLANNING AND                      *
/* INSTALLING THE INFORMATION/FAMILY MANUAL.                        *
/*                                                                    *
/* STEP1 WILL ASSEMBLE THE MEMBER SPECIFIED INLINE                 *
/*                                                                    *
/* STEP2 WILL LINKEDIT THE MEMBER AND STORE IN USER LOADLIB       *
/*                                                                    *
/*****
//STEP1 EXEC PGM=ASMA90,PARM='NODECK,OBJECT,NOTERM'
//SYSLIB DD DISP=SHR,DSN=uprfx.uqual.SBLMMACS
// DD DISP=SHR,DSN=SYS1.MACLIB
//SYSLIN DD DISP=(NEW,PASS,DELETE),DSN=&&SYSLIN,
// UNIT=tdisk,SPACE=(TRK,(3,5),RLSE),
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//SYSPUNCH DD DUMMY
//SYSUT1 DD UNIT=tdisk,SPACE=(CYL,(5,3))
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
        TITLE 'BLGVDSN - INFORMATION/MANAGEMENT SESSION PARAMETERS'
BLGVDSN CSECT
        BLGPARMS DICTDS=DICTDS,                                X
                RFTDS=RFTDS,                                  X
                RPANLDS=(RPANEL0,RPANEL1)
SYSTEM  BLGCLUST NAME=5,                                       X
        PRODUCT=MGMT,                                         X
        SDDS=MGTSDDS,                                         X
        SDIDS=MGTSDIDS,                                       X
        SDLDS=MGTSDLDS
MGTSDDS BLGCLDSN DSN=uprfx.uqual.SDDS
MGTSDIDS BLGCLDSN DSN=uprfx.uqual.SDIDS
MGTSDLDS BLGCLDSN DSN=uprfx.uqual.SDLDS
RPANEL0 BLGCLDSN DSN=uprfx.uqual.ENDPNLS <== Endeavor PANEL
RPANEL1 BLGCLDSN DSN=uprfx.uqual.IBMPNLS <== IBM PANELS
RFTDS   BLGCLDSN DSN=uprfx.uqual.SBLMFMT,FILE=RFTDD
DICTDS  BLGCLDSN DSN=uprfx.uqual.DICT
        BLGGEN ,
        END
/*
/**
//STEP2 EXEC PGM=IEWL,PARM='LIST,NCAL,XREF,SIZE=(256K,64K)',
// COND=(0,NE)
//SYSLIN DD DISP=(OLD,DELETE),DSN=&&SYSLIN
// DD *
        ENTRY BLGVDSN
        NAME BLGSESxx
//SYSLMOD DD DISP=SHR,DSN=uprfx.uqual.LOADLIB
//SYSUT1 DD UNIT=tdisk,SPACE=(CYL,(5,3))
//SYSPRINT DD SYSOUT=*

```

2.3.7 Summary

The pieces are now in place for the interface on both the Info/Man and the Endeavor sides. In Phase 3, these pieces are connected to enable the interface.

2.4 Phase 3. Connecting the Interface

You must complete the tasks described for Phase 1 and Phase 2 before starting the tasks described in this section.

This phase of the installation process involves setting up the Info/Man and Endeavor sides of the interface to work together, as described next.

Step	Action
1	Modify the @EINFO macro.
2	Assemble and link BC1TEI90 into a LINKLIST or authorized library.
3	Assemble C1DEFLTS with Info/Man password.
4	Make sure that required panels are in the ISPPLIB data set.
5	Make sure that members INFO01 and PKMR02 are in your ISPMLIB data set.
6	Edit C1SB3000 skeleton JCL.

These steps are described in the following sections.

2.4.1 Step 1. Modify the @EINFO Macro

Various parts of the interface access the user-assembled module BC1TEI90 to obtain user parameter information. BC1TEI90 is a load module used during exits 5, 2, 3, and 7 for start-up parameters, during display services, and during batch execution. The user must assemble this load module. Once assembled, this module should be linked into a LINKLIST or authorized library.

The @EINFO macro, supplied with the interface, provides input to BC1TEI90. The macro can be found embedded in member BC1JEI10 in iprfx.igual.JCLLIB. Edit this macro based on the information contained in the Phase 1 checklist and your site standards.

The @EINFO macro is shown next.

```
@EINFO ENTRY=START,
      BLGSESS=00,
      APISEID=USERID,
      EPIDT=Endevor,
      INVCLASS=MASTER,
      WAITTIME=300,
      RECTYPE=CHANGE,
@EINFO ENTRY=END
```

The entries in this macro are described next.

Parameter	Description
ENTRY=START	Indicates the beginning of the BC1TEI90 table.
BLGSES=	Info/Man looks for session parameters in a module named BLGSESxx, where xx is a two-character identifier for the module. This module is described in the Info/Man documentation. You recorded the components of this module during Phase 1 of this installation, as well as the two-character identifier for the module. When modifying the @EINFO macro supplied with the interface, replace the default value 00 with the value you specified in Phase 1. The default is 00. This is the same as standard Info/Man.
APISEID=	The API requires a session ID during initialization. This session can be called USERID or can be a predefined Info/Man API ID. The default is USERID. Leaving this default allows the same start-up module to be used by many people, with the interface substituting the user ID of the caller for the literal USERID. If you include this keyword, you must provide a value.
EPIDT=	This keyword contains the name of the assembled E/INFO table produced by the batch utility. The default is Endevor. If you include this keyword, you must provide a value.
INVCLASS=	Info/Man has its own security system. It is broken into classes with privileges. Every user or API ID is assigned to at least one CLASS. The API requires a class name. The default is MASTER.
WAITTIME=	The API allows the application to define how long it will wait for a response from the Info/Man API server. If the user does not use this keyword, the default will be 300 seconds.

Parameter	Description
RECTYPE=	Users must select one kind of Info/Man record for recording information about stand-alone actions. They can use either Change or Problem records. Display services also need this information. The default is CHANGE. If coded, it cannot be blank.
ENTRY=END	Required. This statement denotes the end of the BC1TEI90 table.

2.4.2 Step 2. Assemble and Link BC1TEI90

Assemble and link BC1TEI90 using the JCL in member BC1JEI10 in iprfx.igual.JCLLIB. BC1JEI10 JCL is shown next. Before submitting this JCL, provide a job card, confirm that iprfx.igual and uprfx.uqual are the correct data set qualifiers, and confirm that tdisk is a valid unit name.

```

/* ( COPY JOBCARD )
/******
/*
/* BC1JEI10 - THIS JOB IS USED TO CUSTOMIZE AND BUILD THE      *
/* ENDEVOR-OS/390 INFORMATION/MANAGEMENT INTERFACE            *
/* STARTUP BLOCK CALLED BC1TEI90                               *
/*
/* THE FOLLOWING UPDATES MUST BE MADE TO THIS JCL BEFORE     *
/* IT CAN BE EXECUTED:                                        *
/*
/* 1. UPDATE THE JOBCARD TO REFLECT CORRECT SITE INFORMATION *
/* 2. REVIEW THAT THE SOURCE AND CONLIB DATA SET NAMES ARE *
/*    CORRECT.                                               *
/*    - IPRFX.IGUAL.SOURCE                                   *
/*    - UPRFX.UQUAL.LOADLIB                                  *
/*
/******
//STEP1 EXEC PGM=ASMA90,PARM='NODECK,OBJECT,NOTERM'
//SYSLIB DD DISP=SHR,DSN=IPRFX.IGUAL.SOURCE
//      DD DISP=SHR,DSN=SYS1.MACLIB
//SYSLIN DD DISP=(NEW,PASS,DELETE),DSN=&&SYSLIN,
//          UNIT=TDISK,SPACE=(TRK,(3,5),RLSE),
//          DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//SYSPUNCH DD DUMMY
//SYSUT1 DD UNIT=TDISKSPACE=(CYL,(5,3))
//SYSPRINT DD SYSOUT=*
//SYSIN DD *

```

```

*****
*   THE PURPOSE OF THIS JOB IS TO ASSEMBLE THE  BC1TEI90 START   *
*   UP LOAD MODULE                                           *
*   *                                                         *
*   THIS JOB SHOULD BE RUN AFTER THE E-PIDT HAS BEEN CREATED *
*   AND AFTER AN INFO/MAN SESSION PARAMETER BLOCK HAS BEEN   *
*   ESTABLISHED. BOTH OF THOSE VALUES ARE REQUIRED FOR THIS  *
*   ASSEMBLY.                                               *
*   *                                                         *
*   *                                                         *
*****
BC1TEI90 TITLE 'ENDEVOR/INFO STARTUP BLOCK'
*****
*   THIS IS A SAMPLE ENDEVOR/INFO STARTUP BLOCK. DURING EXIT 5 PROCESS-*
*   ING THIS BLOCK WILL BE PROVIDE THE INTERFACE THE REQUIRED PARMS *
*   AND SESSION ID TO INITIALIZE THE INTERFACE AND THE INFO/MAN *
*   API SERVER.                                             *
*****
*   *                                                         *
*   THERE ARE FIVE MAIN PIECES OF INPUT.                   *
*   *                                                         *
*   BLGSESS=00      <=== STANDARD INFO/MAN SESSION *
*                   SUFFIX. DEFAULT IS 00. *
*                   (BLGSES00) *
*   *                                                         *
*   APISESID=USERID <=== THE 4.2 API WILL ALLOW USERS *
*                   TO HAVE SESSION IDS. *
*                   IF THIS FIELD IS NOT MODIFIED *
*                   THEN THE REAL TIME USERID WILL *
*                   BE SUBSTITUTED AT EXEC TIME. *
*   *                                                         *
*   ENDVPIDT=ENDEVOR <=== THE NAME OF THE LOAD MODULE *
*                   THAT WAS CREATED BY THE BATCH *
*                   UTILITY. THIS IS THE TABLE *
*                   NAME. DEFAULT IS ENDEVOR. *
*   *                                                         *
*   INVCLASS=MASTER <=== INFO/MAN HAS ITS OWN SECURITY *
*                   SYSTEM. THROUGH THE USE OF *
*                   CLASSES, USERIDS ARE REGISTERED *
*                   IN ONE OR MORE CLASSES. EACH *
*                   CLASS HAS A SET OF PRIVLEDGES. *
*                   DEFAULT IS MASTER. *
*   WAITTIME=300    <=== THE MAX TIME THE INTERFACE *
*                   WILL WAIT FOR A REPSONSE BACK *
*                   FROM THE INFO/MAN SERVER. *
*                   DEFAULT IS 300 SECS. *
*****
@EINFO ENTRY=START, X
          BLGSESS=00, X
          APISESID=USERID, X
          ENDVPIDT=Endevor, X
          INVCLASS=MASTER, X
          WAITTIME=300, X
          RECTYPE=CHANGE
@EINFO ENTRY=END

```

2.4 Phase 3. Connecting the Interface

```
/*
/*****
/* THE INPUT SHOULD BE ASSEMBLED AND LINKED INTO A STEPLIB DATASET. *
/* PLEASE REFER TO THE INTERFACE CHAPTER COVERING IMPLEMENTATION AND*
/* INSTALLATION. *
/* *
/*****
/*
//STEP2 EXEC PGM=IEWL,PARM='LIST,NCAL,RENT,XREF,SIZE=(256K,64K)',
// COND=(0,NE)
//SYSLIN DD DISP=(OLD,DELETE),DSN=##SYSLIN
//SYSLMOD DD DISP=SHR,DSN=uprfx.uqua1.LOADLIB(BC1TEI90)
//SYSUT1 DD UNIT=t disk,SPACE=(CYL,(5,3))
//SYSPPRINT DD SYSOUT=*
```

2.4.3 Step 3. Reassemble C1DEFLT5 with the Info/Man Password

Include the Info/Man password in your C1DEFLT5 table. The password parameter is in the TYPE=MAIN section of the defaults table, as shown below.

```

C1DEFLT5 TYPE=MAIN,
  ACCSTBL=,                ACCESS SECURITY TABLE          X
  ACMIDXUP=N,              CROSS-REFERENCE DATA UPDATE    X
  ACMROOT=,                ROOT DATABASE                   X
  ACMXREF=,                XREF DATABASE                   X
  APRVFLG=N,              APPROVAL PROCESSING (Y/N)      X
  ASCM=N,                  ASCM CONTROL OPTION            X
  BATCHID=0,              BATCH UID FROM JOBNAME/USER=   X
  CIPODSN=,                CCID VALIDATION DSN           X
  CSP=N,                   CSP CONTROL OPTION             X
  CUNAME='*** PUT YOUR COMPANY NAME HERE ***', (50 CHAR) X
  DB2=N,                   DB2 CONTROL OPTION            X
  ELINK=N,                 Endeavor/LINK CONTROL OPTION   X
  ESSI=N,                  ESSI CONTROL OPTION           X
  INFO=N,                  INFOMAN CONTROL OPTION        X
  LIBENV=,                 LIBRARIAN (LB), PANVALET (PV)  X
  LIBENVP=N,              LIBRARIAN/PANVALET OPTION     X
  LIBRPGM=,               LIBRARIAN BATCH PROGRAM NAME  X
  LINESPP=60,             LINES PER PAGE                X
  MACDSN='IPRFX.IQUAL.SOURCE', E/OS/390 SOURCE LIBRARY X
  PKGDSN='UPRFX.UQUAL.PACKAGE', PACKAGE DATASET NAME X
  PKGTSO=N,               FOREGROUND PACKAGE EXEC (Y//N) X
  PDM=N,                   PDM CONTROL OPTION            X
  PKGSEC=,                 PACKAGE SECURITY               X
  PKGCSEC=N,              PACKAGE CAST SECURITY (Y/N)    X
  PKGCVAL=0,              PKG COMPONENT VALIDATION (Y/O) X
  PROC=N,                  PROCESSOR OPTION              X
  RACFGRP=,               E/OS/390 RACF GROUP NAME      X
  RACFPWD=,               E/OS/390 RACF OPTION          X
  RACFUID=,               E/OS/390 RACF USERID         X
  SITEID=0,               E/OS/390 SITE ID             X
  SMFREC#=0,              SMF RECORD NUMBER            X
  SPFEDIT=SPFEDIT,        DEFAULT PDS RESERVE           X
  SYSEWL=SYSEWL,         DEFAULT PDS/LINK EDIT RESERVE X
  UIDLOC=(1,7),           UID/JOBNAME START/LENGTH POS  X
  VIOUNIT=TDISK,          UNIT FOR VIO-ELIGIBLE ALLOC   X
  WRKUNIT=TDISK,          UNIT NAME FOR WORK SPACE      X
  WORKVOL=                 VOL SER NUMBER FOR WRKUNIT

```

Reassemble the C1DEFLT5 table with the JCL in member BC1JDEFT in iprfx.igual.JCLLIB.

2.4.4 Step 4. Make Sure Required Panels Are in ISPLLIB

Make sure that the panels listed next reside in your ISPLLIB. These panels can be found in iprfx.igual.ISPLLIB.

Panel ID	Description
C1EILIST, CITILIST	CCID list panel and its associated tutorial panel.
C1SEIBRW, CITEIBRW	Info/Man record display for packages and stand-alone actions and its associated tutorial panel.

Panel ID	Description
C1SP1000, C1SP2000, C1SP3000, C1SP4000, C1SP5000, C1SP6000, C1SP7000	Package panels with capability to display correlation data.
CITP1000, CITP2000, CITP3000, CITP4000, CITP5000, CITP6000, CITP7000	Tutorials for package panels with capability to display correlation data.

2.4.5 Step 5. Make Sure INFO01, PKEX21, and PKMR02 Are in ISPMLIB

Members INFO01, PKEX21, and PKMR02 contain interface messages, and must reside in your ISPMLIB. These members can be found in iprfx.igual.ISPMLIB.

2.4.6 Step 6. Edit C1SB3000 Skeleton JCL

Member C1SB3000 is used to submit Endeavor package and batch processing requests. The member can be found in iprfx.igual.ISPSLIB. Before submitting this JCL, confirm that iprfx.igual are the correct data set qualifiers, and confirm that tdisk is a valid unit name.

```

)CM THIS SKELETON IS USED TO GENERATE ENDEVOR FOR OS/390 JCL FOR BATCH.
)SEL &C1BJC1  ␣= &Z
&C1BJC1
)ENDSEL
)SEL &C1BJC2  ␣= &Z
&C1BJC2
)ENDSEL
)SEL &C1BJC3  ␣= &Z
&C1BJC3
)ENDSEL
)SEL &C1BJC4  ␣= &Z
&C1BJC4
)ENDSEL
//*
```

```

//*****
//* Endeavor JCL STATEMENTS *
//*****
//NDVRBAT EXEC PGM=NDVRC1,DYNAMNBR=1500,REGION=4096K,
)SEL &C1BLDPKG ^= P
// PARM='C1BM3000'
)ENDSEL
)ENDSEL
)SEL &C1BLDPKG = P
// PARM='C1BM3000,,&VPHPKGID'
)ENDSEL
//CONLIB DD DSN=iprfx.iqual.CONLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//*****
//* SORT WORK FILES *
//*****
//SORTWK01 DD UNIT=t disk,SPACE=(CYL,(2,1))
//SORTWK02 DD UNIT=t disk,SPACE=(CYL,(2,1))
//SORTWK03 DD UNIT=t disk,SPACE=(CYL,(2,1))
//SORTWK04 DD UNIT=t disk,SPACE=(CYL,(2,1))
)SEL &VARSICSP = Y
//*****
//* CSP SUPPORT *
//*****
//DCAWORK DD DISP=SHR,
// DSN=&SYSUID..DCAWORK
//DCALOAD DD DISP=SHR,
// DSN=AEPRFX.AELOAD
// DD DISP=SHR,
// DSN=AEPRFX.ADLOAD
//DCAEZED DD DISP=SHR,
// DSN=UALFPRFX.EZEMSG
//DCAMAPD DD DISP=SHR,
// DSN=UALFPRFX.FZEMAPDS
//DCATESD DD DISP=SHR,
// DSN=UALFPRFX.FZETUTOR
//DCAHECD DD DISP=SHR,
// DSN=UALFPRFX.FZEMSG
)ENDSEL
//*****
//* PANVALET SUPPORT *
//*****
//C1TPDD01 DD UNIT=t disk,SPACE=(CYL,5),
// DCB=(RECFM=VB,LRECL=260,BLKSIZE=6160)
//C1TPDD02 DD UNIT=t disk,SPACE=(CYL,5),
// DCB=(RECFM=VB,LRECL=260,BLKSIZE=6160)
//C1TPLSIN DD UNIT=t disk,SPACE=(CYL,5),
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=6160)
//C1TPLSOU DD UNIT=t disk,SPACE=(CYL,5)
//C1PLMSGs DD SYSOUT=*
//*****
//* OUTPUT DATA SETS *
//*****
//C1MSGs1 DD SYSOUT=*
//C1PRINT DD SYSOUT=*,DCB=(RECFM=FBA,LRECL=121,BLKSIZE=6171)
//SYSABEND DD SYSOUT=*
//SYSOUT DD SYSOUT=*
)SEL &C1BLDPKG ^= P

```

2.4 Phase 3. Connecting the Interface

```
/******  
/* REQUEST DATA SET *  
/******  
//BSTIPT01 DD DSN=&VNBDFDSN,DISP=SHR  
)ENDSEL  
)SEL &C1BLDPKG = P  
//BSTIPT01 DD UNIT=tdisk,SPACE=(CYL,3),  
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=6160,DSORG=PS)  
)ENDSEL  
/*  
/*  
)SEL &VNBINCF = Y
```

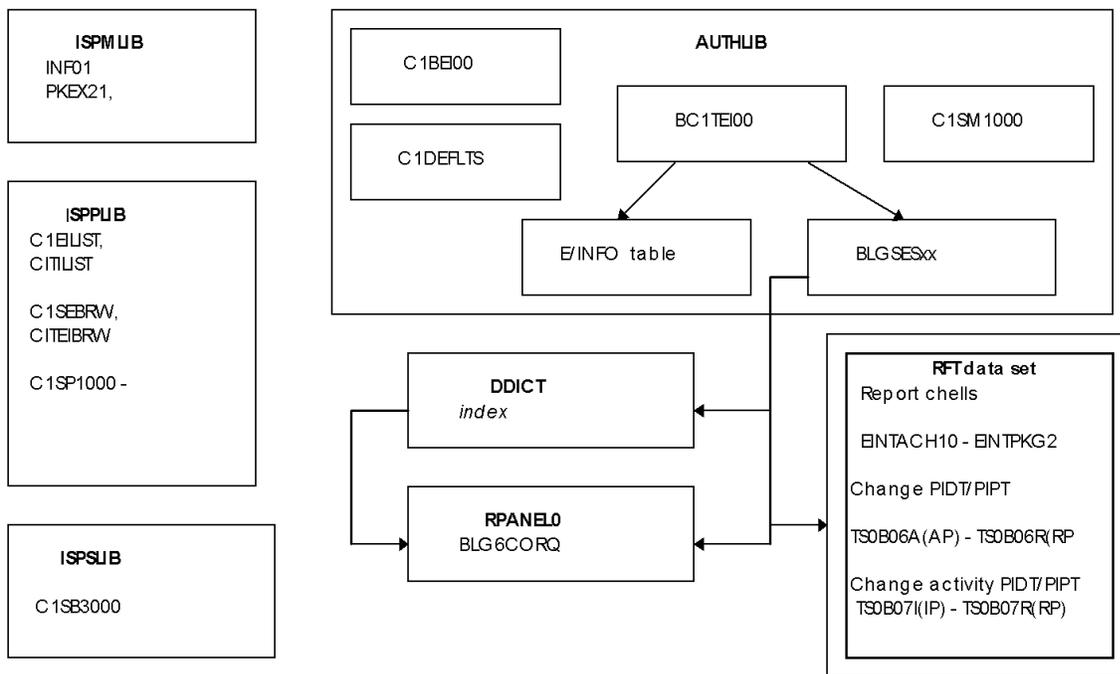
2.5 Phase 4. Verifying the Installation

The list next shows the members that should reside in the designated data sets for the interface to function as designed. If any of them are not there, the interface will not work as designed.

This data set	Must contain these members or information
AUTHLIB	<p>Load modules, as well as the following:</p> <ul style="list-style-type: none"> ■ C1BMEI00 ■ C1DEFLTS table. See 2.4.3, “Step 3. Reassemble C1DEFLTS with the Info/Man Password” on page 2-19. ■ C1SM1000 ■ BC1TEI90, the module containing user parameters, which you assembled in 2.4.2, “Step 2. Assemble and Link BC1TEI90” on page 2-16. ■ E/INFO table. This is the rule set that you build using the batch utility. See 2.3.5, “Step 5. Build a Rule Set” on page 2-9. ■ BLGSESxx, the module containing the Info/Man session parameters. See 2.2.2, “Step 2. Prepare a Session Parameter Member” on page 2-3.
CONLIB	<p>The Endeavor load modules installed from iprfx.igual.CONLIB. Make sure to save the current copy of C1BEXITS before installing the new interface. This allows you to go back to the old interface during testing.</p>
RFTDS	<p>EINTACH0, EINTACH1, EINTAPB0, EINTAPB1, EINTPKG0, EINTPKG1, EINTPKG2.</p> <p>PIDT and PIPT tables. See the list in the next section. These panels are found in iprfx.igual.SOURCE.</p> <p>BLGISPF, the Info/Man supplied ISPF panel.</p>
RPANEL0	<p>BLG6CORQ, the modified version of the Assisted Entry panel. See 2.3.3, “Step 3. Copy Dictionary Update” on page 2-7.</p>
DDICT	<p><i>index</i> RNCC/CCV15. This is the modified dictionary entry associated with the modified Assisted Entry Panel. See 2.3.4, “Step 4. Create the Standard IBM PIDT and PIPT Tables” on page 2-8.</p>
iprfx.igual. ISPPLIB	<p>C1EILIST, C1SEIBRW, C1SP1000 C1SP7000 and the associated tutorials (CITILIST, CITEIBRW, and CITP1000-CITP7000.</p>

This data set	Must contain these members or information
iprfx.igual. ISPMLIB	INFO01, PKEX21, PKMR02.
iprfx.igual. ISPSLIB	C1SB3000.

The graphic below shows how these modules tie the interface together, and indicates the important members that should be at each location.



2.5.1 Required PIDT and PIPT Tables

For the interface to work correctly the following PIDT and PIPT tables must be in the RFT data set pointed to by the BLGSESxx module.

There are three categories of PIDT/PIPT tables: Change, Change Activity, and Problem. These tables are listed next, and can be found in iprfx.igual.SOURCE.

2.5.1.1 Change Record PIDT and PIPT Tables

The following PIDT/PIPT tables related to Change records must be installed.

Table identifier	Description
TS0B06A	Change record, Add, PIDT
TS0B06AP	Change record, Add, PIPT
TS0B06I	Change record, Inquiry, PIDT
TS0B06IP	Change record, Inquiry, PIPT
TS0B06C	Change record, Create, PIDT
TS0B06CP	Change record, Create, PIPT
TS0B06U	Change record, Update, PIDT
TS0B06UP	Change record, Update, PIPT
TS0B06R	Change record, Retrieve, PIDT
TS0B06RP	Change record, Retrieve, PIPT

2.5.1.2 Change Activity PIDT and PIPT Tables

The following PIDT/PIPT tables related to Change Activity records must be installed.

Table identifier	Description
TS0B07I	Change activity record, Inquiry, PIDT
TS0B07IP	Change activity record, Inquiry, PIPT
TS0B07C	Change activity record, Create, PIDT
TS0B07CP	Change activity record, Create, PIPT
TS0B07U	Change activity record, Update, PIDT
TS0B07UP	Change activity record, Update, PIPT
TS0B07R	Change activity record, Retrieve, PIDT
TS0B07RP	Change activity record, Retrieve, PIPT

2.5.1.3 Problem Record PIDT and PIPT Tables

The following PIDT/PIPT tables related to Problem records must be installed.

Table identifier	Description
TS0032I	Problem record, Inquiry, PIDT
TS0032IP	Problem record, Inquiry, PIPT
TS0032C	Problem record, Create, PIDT
TS0032CP	Problem record, Create, PIPT
TS0032U	Problem record, Update, PIDT
TS0032UP	Problem record, Update, PIPT
TS0032R	Problem record, Retrieve, PIDT
TS0032RP	Problem record, Retrieve, PIPT

Chapter 3. Setting Up the Endeavor/INFO Table — the Batch Utility

3.1 The Endeavor Side of the Interface

Endeavor communicates with the Info/Man API through exit points, using a control block referred to here as the Endeavor/INFO table.

The purpose of the Endeavor/INFO table is to provide a flexible, easy way to implement site Info/Man policy through the Info/Man API. In particular, the Endeavor/INFO table allows the user to:

- Utilize custom PIDT tables to provide logical views of Info/Man records.
- Explicitly specify the exit points where actions are to be performed.
- Establish criteria on the Endeavor or the Info/Man side for performing an action.
- Specify the Endeavor information to be recorded in Info/Man, and the Info/Man fields where the information is to be recorded.
- Identify Info/Man actions to be performed (Inquiry, Retrieve, Create, Update), and the PIDT table needed for each action.

You build the Endeavor/INFO table during interface implementation using a batch utility provided with the interface. This utility assembles the table into a load module that is invoked at the exit points defined in the table.

When the table is invoked, Endeavor builds a request for the Info/Man API using the information in the table. The Info/Man API executes the requested actions against the Info/Man database, using the requested PIDT tables.

This chapter describes the syntax for the Endeavor/INFO table.

3.2 Checklist for Implementing the Endeavor Side

Before writing the table definition statement, complete this checklist.

1. Determine what kind of Endeavor information you want to store in Info/Man: package, package action, or stand-alone action information.
2. Be familiar with the necessary Endeavor exit points and exit control blocks for the kinds of information you plan to track. Refer to the *Administration Guide* for this information.
3. Know which control block fields for the exit points you wish to utilize contain information eligible for storage in Info/Man.
4. Decide which of the available control block fields you want to track in Info/Man.
5. Decide Info/Man fields you want to use to store the Endeavor information.

After writing the table definition statement, submit it to the Endeavor batch utility.

3.3 JCL for the Batch Utility

A sample jobstream for the batch utility is in `iprfx.igual.JCLLIB`. `BC1JEI00` JCL is shown below.

```
// (JOB CARD)
/*****
/* THE PURPOSE OF THIS JOB IS TO RUN THE BATCH UTILITY.          */
/* INPUT TO THIS JOB IS THE BATCH UTILITY SYNTAX. OUTPUT FROM   */
/* THIS IS THE E-PIDT TABLE. THIS TABLE NAME MUST BE INCLUDE  */
/* THE ASSEMBLY OF THE BC1TEI90 STARTUP BLOCK.                  */
/* NOTE: NDVRIN POINTS TO THE BATCH SYNTAX THE CUSTOMER WANTS   */
/* USE. THERE ARE THREE SAMPLE SCRIPTS DELIVERED WITH THE      */
/* PRODUCT.                                                       */
/* EISCRIP1 ==> USE FOR STAND ALONE ACTIONS - CHANGE RECORDS    */
/* EISCRIP2 ==> USE FOR STAND ALONE ACTIONS - PROBLEM RECORDS   */
/* EISCRIP3 ==> USE FOR PACKAGE LEVEL INTERFACE                 */
/* REPLACE THE NDVRIN MEMBER NAME IMBMR WITH THE MEMBER YOU     */
/* CHOSE.                                                         */
/*****
/* RUN UTILITY TO ASSEMBLE SYNTAX THAT CREATES E-PIDT LOADMOD  *
/*****
//BATCH EXEC PGM=NDVRC1,PARM=C1BMEI00
/*
//CONLIB DD DSN=iprfx.igual.CONLIB,DISP=SHR
/*
//NDVRIN DD DSN=iprfx.igual.SOURCE(IMBMR),DISP=SHR <=== SEE NOTES
/*****
/* THE ABLMSAMP LIBRARY IS A STANDARD LIBRARY PROVIDED WITH THE *
/* INSTALL OF INFO/MAN.                                          *
/*                                                                *
/* THESE POINT TO THE IBM SUPPLIED PIDT SOURCE. IF YOU HAVE    *
/* YOUR OWN PIDTS, MODIFY ACCORDINGLY. IF YOU ARE USING THE    *
/* PROBLEM LEVEL INTERFACE, USE THE LINES THAT POINT TO THE    *
/* MEMBERS THAT START WITH 'TS0032' OTHERWISE USE THE LINES    *
/* THAT POINT TO THE MEMBERS THAT START WITH 'TS0B06' AND       *
/* 'TS0B07'.                                                    *
/*****
//NDVRPIDT DD DSN=uprfx.igual.ABLMSAMP(TS0B06IS),DISP=SHR <= CHANGE
//          DD DSN=uprfx.igual.ABLMSAMP(TS0B06RS),DISP=SHR
//          DD DSN=uprfx.igual.ABLMSAMP(TS0B06CS),DISP=SHR
//          DD DSN=uprfx.igual.ABLMSAMP(TS0B06US),DISP=SHR
//          DD DSN=uprfx.igual.ABLMSAMP(TS0B06AS),DISP=SHR
/*
//          DD DSN=uprfx.igual.ABLMSAMP(TS0B07IS),DISP=SHR <= ACTIVITY
//          DD DSN=uprfx.igual.ABLMSAMP(TS0B07RS),DISP=SHR
//          DD DSN=uprfx.igual.ABLMSAMP(TS0B07CS),DISP=SHR
//          DD DSN=uprfx.igual.ABLMSAMP(TS0B07US),DISP=SHR
/*
//          DD DSN=uprfx.igual.ABLMSAMP(TS0032US),DISP=SHR <= PROBLEM
//          DD DSN=uprfx.igual.ABLMSAMP(TS0032CS),DISP=SHR
//          DD DSN=uprfx.igual.ABLMSAMP(TS0032IS),DISP=SHR
//          DD DSN=uprfx.igual.ABLMSAMP(TS0032RS),DISP=SHR
/*
```

```
//NDVRRPT1 DD SYSOUT=*
//NDVRRPT2 DD SYSOUT=*
//NDVRRPT3 DD SYSOUT=*
//*
//*SAMPLE JCL FOR BC1JEI00 NDVRWK1 AND NDVRWK2 SHOULD BE AS FOLLOWS
//*
//NDVRWK1 DD DSN=uprfx.uqua1.WORK1,DISP=(,PASS),
//          UNIT=tdisk,SPACE=(TRK,(10))
//NDVRWK2 DD DSN=uprfx.uqua1.WORK2,DISP=(,PASS),
//          UNIT=tdisk,SPACE=(TRK,(10))
//*
//SYSLMOD DD DSN=uprfx.uqua1.LOADLIB,DISP=SHR <== E-PIDT STORED HERE
//SYSUT1  DD UNIT=VIO,SPACE=(TRK,(1,1))
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
```

3.4 Endeavor/INFO Table Syntax

Here is an example of the syntax used to produce an Endeavor/INFO table. The rest of the chapter discusses this example.

```

TABLE NAME (ENDEAVOR);

CTRL;
  IACT (TS0B06I);          /* Inquiry Change for actions */
  RACT (TS0B06R);          /* Retrieve Change for actions */
  CACT (TS0B06C);          /* Create Change for actions */
  UACT (TS0B06U);          /* Update Change for actions */
ECTRL;
  USE(ADD,BEFORE,STANDALONE);
  NOTFOUND (CREATE,ADD0010);
  FOUND      (ADD0020);
  LABEL ADD0010;
  CRITERIA;
  ECRITERIA;
  POPULATE;
    FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=USER);
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
    FIELD PANEL(BLG6DSAB) INDEX(S0E0F)
      VALUE('API CREATED CHG RECORD');
  EPOPULATE;
  RC (0);
  ELABEL;
  LABEL ADD0020;
  CRITERIA;
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
  ECRITERIA;
  POPULATE;
  EPOPULATE;
  CPASS ADDPASS;
  CFAIL ADDFAIL;
  ELABEL;
  LABEL ADDPASS;
  RC (0);
  ELABEL;
  LABEL ADDFAIL;
  RC (8);
  ELABEL;
  EUSE;
  USE(ADD,AFTER,STANDALONE);
  FOUND (ADD0010);
  NOTFOUND (RETURN);
  LABEL ADD0010;
  CRITERIA;
  ECRITERIA;
  POPULATE;
    FIELD PANEL(BLG0C010) INDEX(S0E01)
    TEXT('PROCESSING COMPLETED FOR ELEMENT =EV2ELEM ' ) -
      ('ACTION: =ACTION EXECUTED BY: =USER ' ) -
      ('ON =DATE =TIME!');
  EPOPULATE;
  ELABEL;
  EUSE;
ETABLE;

```

3.4.1 Table Syntax Components

The major components of the Endeavor/INFO table syntax are shown below.

Component	Number	Purpose
Table statement	One per table	Starts, names, and ends the table.
Control (CTRL) block	One per table	Identifies the Info/Man PIDT tables to be used by the requests produced by the Endeavor/INFO table.
Use blocks	One or more per table	Identify the exit points where the table is invoked and associate processing logic with each exit point.
Label blocks	One or more per Use block	Specify processing for an exit point.
Criteria blocks	One or more per Use block	Establish true/false conditions. When there is a true condition, the interface populates Info/Man records with Endeavor information.
Populate blocks	One or more per Use block	Identify Endeavor information to be recorded in Info/Man.

The statements that can be coded for each block are described later in this chapter.

3.4.2 Syntax Diagram Conventions

The following conventions are used in the syntax diagrams in this chapter:

- User-supplied variables are presented in italics. For example:
TABLE NAME (*name*);
- The acceptable values for the variables are supplied in the discussion of the syntax.
- Optional syntax components are enclosed in brackets. For example, in the syntax below the components STANDALONE and PROBLEM are optional.
USE(function, {BEFORE|AFTER|ACTIVITY} [, STANDALONE] [, PROBLEM])
- Required syntax components for which there is more than one acceptable value are shown enclosed in braces, with the acceptable values separated by vertical lines. For example, in the syntax below the second argument between parentheses must be BEFORE, AFTER, or ACTIVITY.
USE(function, {BEFORE|AFTER|ACTIVITY} [, STANDALONE] [, PROBLEM])

3.5 TABLE Statement

The TABLE statement starts a table, names the table, and ends the table.

The syntax is:

```
TABLE NAME (name);
```

```
:
```

```
ETABLE;
```

Syntax components are described below.

Component	Description
TABLE NAME	Required. This must be the first statement in the Endeavor/INFO table syntax.
name	Required. A 1- to 8-alphanumeric character name for the table. This name is also used as the output member name of the load module that interacts with the Info/Man API. Parentheses on the name are optional. The name must be followed by a semicolon.
ETABLE	Required. This must be the final statement in the Endeavor/INFO table syntax. It must be followed by a semicolon.

Example:

```
TABLE NAME (NDVR);
```

```
...
```

```
ETABLE;
```

This example builds an Endeavor/INFO table called NDVR.

3.6 Control Block

The Control (CTRL) block specifies the:

- Info/Man functions that the table may request.
- Endeavor entities affected by the table.
- PIDT tables that must be available to the Endeavor/INFO table.

The CTRL block must follow the TABLE statement. There must be a statement in this block for each function that you want to perform against the Info/Man database. The statements in this block must identify all the PIDT tables required by the requested functions.

Note: Users of standard Info/Man should always use the standard CTRL statements included in the samples.

3.6.1 Control Block Syntax

The syntax for the Control block is:

```
CTRL;
```

```
    Info/Man action Endeavor entity (PIDT name)  
    ...
```

```
ECTRL;
```

3.6.1.1 CTRL Statement

The CTRL keyword is required. It must be the first line in the control statement and must be followed by a semicolon. The parentheses surrounding the PIDT name are optional; however, the PIDT name must be followed by a semi-colon.

3.6.1.2 Info/Man Action

An Info/Man action is required. There must be at least one Info/Man action specified in a control statement. Acceptable values are

- I (Inquiry)
- R (Retrieve)
- C (Change)
- U (Update)
- A (Add)

Use these actions as shown below:

Use this action	If you want to
Inquiry (I)	Check to see if an Info/Man record exists.
Retrieve (R)	Test a field in an Info/Man record for a specific value.
Create (C)	Create records in Info/Man based on criteria in the Endeavor/INFO table.
Update (U)	Update existing records in Info/Man.
Add (A)	Use the Info/Man Change Activity record type to store information about package actions.

The Info/Man action appears together with an Endeavor entity to which it applies. See the next section for a description of Endeavor entities.

3.6.1.3 Endeavor Entity

An Endeavor entity is required and specifies an entity for which you want to track information. Acceptable values are ACT (stand-alone actions), PKG (packages), and PKGA (package actions). For a table listing possible combinations of Info/Man actions and Endeavor entities, see 3.6.2, "Possible Requests" on page 3-11.

3.6.1.4 PIDT Name

A PIDT name is required. Each request clause must identify an Info/Man PIDT table. The Info/Man API uses this table to log Endeavor information in Info/Man.

Parentheses are optional. The PIDT table name must be followed by a semicolon.

3.6.1.5 ECTRL Statement

An ECTRL statement is required and must be the final line in the control statement. It must be followed by a semicolon.

3.6.2 Possible Requests

The following API requests may be defined in this block.

This request	Must refer to a PIDT table defined for	And is required if a Use block contains the following
AACT	Add activity record creations.	Package action exit points, Create Activity, and one or more Populate blocks.
CACT	Action Change or Problem record creations.	Action exit points and one or more populate blocks.
CPKG	Package Change record creations.	Package exit points and one or more Populate blocks.
CPKGA	Package Activity record creations.	Package action exit points and one or more Populate blocks.
IACT	Action Change or Problem record inquiries.	Stand-alone action exit points.
IPKG	Package Change record inquiries.	Package exit points.
IPKGA	Package Activity record inquiries.	Package action exit points. Create activity.
RACT	Action Change or Problem record retrievals.	Action exit points and one or more Criteria blocks.
RPKG	Package Change record retrievals.	Package exit points and one or more Criteria blocks.
RPKGA	Package Activity record retrievals.	Package action exit points and one or more Criteria blocks.
UACT	Action Change or Problem record updates.	Action exit points, the STANDALONE keyword and one or more Populate blocks.

This request	Must refer to a PIDT table defined for	And is required if a Use block contains the following
UPKG	Package Change record updates.	Package exit points and one or more Populate blocks.
UPKGA	Package Activity record updates.	Package action exit points and one or more Populate blocks. Create Activity and one or more Populate blocks.

3.6.3 Example 1

To store and maintain stand-alone action information in Info/Man, use this CTRL statement:

```
CTRL;
  IACT (TS0B06I); /* Inquiry Change for actions */
  RACT (TS0B06R); /* Retrieve Change for actions */
  CACT (TS0B06C); /* Create Change for actions */
  UACT (TS0B06U); /* Update Change for actions */
ECTRL;
```

This statement identifies four Change record PIDT tables (TS0B06I, TS0B06R, TS0B06C, TS0B06U), and indicates that the table may request any of the available API actions (Inquiry, Retrieve, Create, or Update).

3.6.4 Example 2

To store and maintain package and package action information in Info/Man, use this CTRL statement:

```
CTRL;
  IPKG (TS0B06I); /* Inquiry Change table for packages */
  IPKGA (TS0B07I); /* Inquiry Activity table for package actions */
  RPKG (TS0B06R); /* Retrieve Change table for packages */
  RPKGA (TS0B07R); /* Retrieve Activity table for package actions */
  CPKG (TS0B06C); /* Change Change table for packages */
  CPKGA (TS0B07C); /* Change Activity table for package actions */
  UPKG (TS0B06U); /* Update Change table for packages */
  UPKGA (TS0B07U); /* Update Activity table for package actions */
ECTRL;
```

This statement tells the Info/Man API that this Endeavor/INFO table may request access to any of four Change record PIDT tables (TS0B06I, TS0B06R, TS0B06C, TS0B06U) for storing package information, and four Activity record tables (TS0B07I, TS0B07R, TS0B07C, TS0B07U) for storing package action information.

3.7 Use Blocks

A Use block identifies one or more exit points where the Endeavor for OS/390 Info/Man interface is to be invoked. A table definition can have multiple Use blocks.

All exit points at which you want to communicate with Info/Man must appear in a Use block. A single USE block can refer to one, or more than one exit point. If a Use block refers to more than one exit point, all exit points in the Use block must:

- Be of the same kind. For example, a Use block can refer to multiple stand-alone action exit points or multiple package exit points, but cannot refer to both stand-alone action exit points and package exit points.
- Utilize the same control blocks. Actions that can appear in the same Use block are:
 - ADD, UPDATE, RESTORE
 - RETRIEVE, ARCHIVE
 - TRANSFER, MOVE
 - GENERATE, DELETE
 - BACKIN, BACKOUT, CAST, COMMIT, EXECUTE, RESET, REVIEW, SHIP

The following actions must have their own Use block:

- CREATE Activity
- CREATE Before
- CREATE After
- PDELETE Before
- PDELETE After
- GENPKGID

Each Use block has the following components:

- A USE statement to identify one or more exit points where the Endeavor/INFO table is to be invoked. Each USE statement has one or more required CTRL block statements.
- A NOTFOUND statement to indicate what to do if an Info/Man record does not exist for this exit point, and which points to a Label block that specifies processing.
- A FOUND statement to indicate what to do if an Info/Man record exists for this exit point, and which points to a Label block that specifies processing.

- One or more Label blocks, to specify the processing associated with the exit points named in the USE statement.
- A EUSE statement to mark the end of the Use block.

3.7.1 USE Statement

The USE statement is the first statement in a Use block, and identifies the exit point or points referred to by the Use block.

The syntax for referring to a single exit point is:

```
USE(action, {BEFORE|AFTER|ACTIVITY} [,STANDALONE] [,PROBLEM] );
```

The syntax for referring to multiple exit points is:

```
USE((action, {BEFORE|AFTER|ACTIVITY} [,STANDALONE] [,PROBLEM] )
```

```
... ( action, {BEFORE} [ .,STANDALONE] [,PROBLEM] ));
```

3.7.1.1 USE

The USE statement is required and must be the first word in the Use block.

3.7.1.2 Action

This variable identifies the action for the Use block. The action along with the subparameter Before/After/Activity, uniquely identifies a Endeavor exit point. Valid combinations are:

Action	Subparameter
Backin	Before/After
Backout	Before/After
Cast	Before/After
Commit	Before/After
Confirm	After
Create	Before/After
Create	Activity. This exit point builds activity records for package actions at after-cast time.
Pdelete	Before/After (Package Delete)
Execute	Before/After
GENPKGID	Before/After
Modify	Before/After
Reset	Before/After

Action	Subparameter
Review	Before/After
Ship	Before/After
Add	Before/After
Update	Before/After
Restore	Before/After
Retrieve	Before/After
Archive	Before/After
Generate	Before/After
Delete	Before/After
Move	Before/After
Transfer	Before/After

3.7.1.3 STANDALONE

This subparameter is only valid with action functions. When used, it identifies actions running outside of the package environment.

3.7.1.4 PROBLEM

This subparameter can only be used with stand-alone actions. When used, it means that a Problem record will be used instead of a Change record when recording information in the Info/Man database.

3.7.1.5 Example 1

Here is an example of USE statement syntax for a single exit point:

```
USE(ADD, BEFORE, STANDALONE);
```

This example indicates that the Use block refers to a before-action exit for ADD actions, and that the ADD actions are performed in stand-alone mode (not as part of packages).

3.7.1.6 Example 2

Here is an example of USE statement syntax for multiple exit points:

```
USE ((ADD, AFTER),
    (UPDATE, AFTER),
    (RESTORE, AFTER));
```

This example indicates that the Use block applies to the after-exit point for the named actions, when the actions are included in a package.

3.7.2 NOTFOUND Statement

NOTFOUND and FOUND statements must follow the USE statement. Each statement can appear only once for a Use block.

Once an exit point is driven, a search is made to locate the Info/Man record from the database. This statement instructs the interface how to proceed when the Info/Man record has not been created yet. This statement is required for:

- All stand-alone action exit points.
- The following package exit points: Create, Modify, Cast, Delete.

This statement is not valid when used with the package exit points for Backout, Backin, Commit, Confirm, Create Activity, Execute, Reset, Review, and Ship.

Statement syntax is:

```
NOTFOUND ({[CREATE,] label | RETURN});
```

Component	Description
NOTFOUND	Required. NOTFOUND must be the first word in this statement.
CREATE	<p>Optional. The CREATE keyword tells the interface to create one of the following in Info/Man:</p> <p>A package change record. If you want to do this, the CTRL block must contain a CPKG statement.</p> <p>A change activity record for a package action. If you want to do this, the CTRL block must contain CPKGA and AACT statements.</p> <p>A change or problem record for a stand-alone action. If you want to do this, the CTRL block must contain a CACT statement.</p> <p>The criteria for populating the new Info/Man record are contained in the block identified by the label parameter.</p> <p>If you want to update existing Info/Man records, omit CREATE from the NOTFOUND statement, and make sure that UPKG, UPKGA, or UACT statements are in the CTRL block.</p>
<i>label</i>	Optional. This identifies the label name where execution resumes when a NOTFOUND condition occurs. Required if CREATE is the first parameter.
RETURN	Optional. Instructs the interface to return to Endeavor.

Examples of NOTFOUND statements appear in the section on FOUND statements.

3.7.3 FOUND Statement

This statement directs the logic process when the Info/Man record has been found.

```
FOUND ({[DELETE,] label | DELETE,RETURN | RETURN} );
```

Syntax components are described below.

Component	Description
FOUND	Required. FOUND must be the first word in this statement.
DELETE	Optional. This parameter is only valid on the Package Delete (PDELETE) After exit point. This option deletes the Package Change record along with all its corresponding activity records. If you use this parameter, make sure that PDELETE is the first parameter in the USE statement.
<i>label</i>	Optional. This identifies the label name where execution resumes when a FOUND condition occurs. Required if DELETE is the first parameter.
RETURN	Optional. Instructs the interface to return to Endeavor.

3.7.3.1 Example 1

Here is an example of NOTFOUND/FOUND statements

```
NOTFOUND (CREATE,CMD0010);
FOUND (CMD0020);
```

This example indicates that if the interface does not find a record on the Info/Man side, it is to create that record. The information needed to create the record can be found later in the table at label CMD0010.

The example also indicates that if the record is found in Info/Man, the interface should execute the processing specified in label CMD0020.

3.7.3.2 Example 2

Here is a second example of NOTFOUND/FOUND statements:

```
NOTFOUND (ERR0010);
FOUND (RETURN); /* GREAT — JUST RETURN */
```

This example indicates that if the record is not found, the interface is to execute the processing specified in label ERR0010, and that if the record is found, the interface should return to Endeavor.

3.7.4 Label Blocks

See 3.8, “Label Block” on page 3-19 for information.

3.7.5 EUSE Statement

Each USE statement must have a corresponding EUSE statement as the last statement in a Use block. Its syntax is:

```
EUSE;
```

3.7.5.1 Example 1

This example shows the overall structure of a Use block:

```
USE(GENERATE, AFTER, STANDALONE);
  FOUND (GEN0010);
  NOTFOUND (RETURN);
  LABEL GEN0010;
    CRITERIA;
    ECRITERIA;
    POPULATE;
      FIELD PANEL(BLG0C010) INDEX(S0E01)
        TEXT ('Action processing completed rc = =ACTRC')
        TEXT ('ELEMENT: =EV1ENV / =EV1SYS / =EV1SUB ') -
          ('/ =EV1TYP / =EV1STAGE / =EV1ELM')
        TEXT ('ACTION: =ACTION USER: =USER DATE: =DATE ') -
          ('TIME: =TIME');
      EPOPULATE;
    ELABEL;
  EUSE;
```

For more information about specific statements in this example, see the appropriate section of this chapter.

3.8 Label Block

Label blocks specify processing that is to occur related to the exit point in a Use block. A Label block must be within a Use block. Multiple Label blocks are allowed per exit point.

Label blocks must contain:

- A LABEL statement
- An ELABEL statement

The Label block can also contain:

- CPASS/CFAIL statements
- TSP statements
- Return code (RC) statements
- Message (MSG) statements
- Populate blocks
- Criteria blocks

The LABEL statement marks the beginning of the block. All statements within the Label block are processed. Once the ELABEL statement is reached, control is returned to Endeavor unless a CPASS/CFAIL statement is included in the block.

3.8.1 LABEL Statement

The LABEL statement indicates the start of a Label block, and must also contain the name of that block. The syntax is:

```
LABEL name;
```

Syntax components are described below.

Component	Description
LABEL	Required. LABEL must be the first word in a Label block.
<i>name</i>	Required. A 1- to 8-character alphanumeric name for the label. The name must be followed by a semicolon.

3.8.1.1 Example

This example shows the use of a comment with the LABEL statement:

```
LABEL GEN0010; /* Change Record not found: Create one. */
```

3.8.2 ELABEL Statement

The ELABEL statement must be the last statement in a Label block. The syntax is:
ELABEL;

The semicolon at the end of the statement is required.

3.8.3 CPASS/CFAIL Statements

Use CPASS/CFAIL statements to pass control to another Label block when the current block's ELABEL statement is reached, based on the conditions set by the Criteria block. The syntax is:

```
{CPASS|CFAIL} label-name;
```

Syntax components are described below.

Component	Description
CPASS	Indicates a true condition.
CFAIL	Indicates a false condition.
<i>label name</i>	The 1- to 8-character name of the label where processing resumes when the true or false condition has been met.

There can be one CPASS/CFAIL statement per Label block. The CPASS and CFAIL statement may exist anywhere in the block, but the condition is only tested and the branch occurs just prior to the ELABEL.

In order to code these statements, a Criteria and a Populate block must be present in the Label block. CPASS represents a true condition and CFAIL represents a false condition. CPASS and CFAIL may be coded independently.

Coding a null Criteria block sets a CPASS condition. However, if you want to pass control to another Label block, you must still code the CPASS statement, specifying the Label block to which you want to pass control.

3.8.3.1 Example of CFail Statement

Here is an example of the use of CFail:

```
LABEL ARC0020;          /* Change Record found */  
  RC (0);  
  CRITERIA;  
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');  
  ECRITERIA;  
    POPULATE;  
    EPOPULATE;  
  CFail ARC0022;  
ELABEL;
```

In this example, a PIDT field in Info/Man would be checked for the value OPEN. The CFAIL statement tells Endeavor to execute the processing specified in Label block ARC0022 if the field does not contain the value OPEN.

3.8.3.2 Example of CPASS Statement

Here is an example of the use of CPASS:

```

LABEL CST0010;          /* CHANGE RECORD NOT FOUND:  CREATE ONE.  */
  CRITERIA;
  ECRITERIA;
  POPULATE;
    /*****
    * PECBUSER:  CURRENT USER ID          *
    *****/
    FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=PECBUSER);
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
    FIELD PANEL(BLG0C010) INDEX(S0E01)
    VALUE('ENDEAVOR/INFO CREATED CHANGE RECORD');
  EPOPULATE;
  CPASS CST0020;
ELABEL;

```

In this example, the interface first executes the processing in this Label block to create a change record in Info/Man. The CPASS statement instructs Endeavor to execute the processing specified in Label block CST0020, because a null Criteria block sets a true condition.

3.8.3.3 Example of CPASS/CFAIL Statements

You can use CPASS and CFAIL statements together when you want to test an Endeavor or an Info/Man field for the presence of a value, then execute one Label block if the value exists, and another Label block if the value does not exist.

```

LABEL CRT0020;          /* LOOKING FOR ACTION:  ADD          */
  CRITERIA;
    FIELD (=PACTACTN) VALUE('ADD      ');
  ECRITERIA;
  POPULATE;
  EPOPULATE;
  CPASS GRP0010;
  CFAIL CRT0025;
ELABEL;

```

In this example, Endeavor tests the PACTACTN field for the presence of the value ADD. The CPASS statement instructs Endeavor to continue processing at Label block GRP0010 if ADD is the value in this field. The CFAIL statement instructs Endeavor to resume processing at Label block CRT0025 if ADD is not the value in the field.

3.8.4 TSP Statements

There can be one TSP statement per Label block. A TSP statement must be placed before any Criteria or Populate blocks. The syntax is:

```
TSP;
```

A TSP statement instructs the Info/Man API to invoke a TSP. The user is responsible for developing this TSP. The TSP is invoked within the Info/Man TSP driver BLGAPI00. The user must update the Info/Man TSP driver to include his/her TSP by including a LINK statement for transaction 111. For more information regarding TSPs and the Info/Man see the *IBM Tivoli Information/Management Application Program Interface Guide SC34-4592-00*.

A TSP statement can appear only in the initial Label block of a chain of Label blocks. TSP statements cannot be coded in a block to which control has been passed by a CPASS or CFAIL statement.

3.8.4.1 Example 1

The TSP statements in this example are allowed, because LABEL (ABC) and LABEL (CDE) each start a chain of Label blocks and therefore can each contain a TSP statement.

```
NOTFOUND (ABC);
FOUND (CDE);
.
LABEL ABC;
TSP;
.
CPASS FGH;
ELABEL;
LABEL CDE;
TSP;
.
CPASS KLM;
ELABEL;
```

3.8.4.2 Example 2

The TSP statement in LABEL (CDE) below is not allowed, because LABEL (CDE) is the second block in a chain, pointed to by the CFAIL statement in LABEL (ABC).

```
NOTFOUND (ABC);
FOUND (CDE);
.
LABEL ABC;
TSP;
.
CFAIL CDE;
ELABEL;
LABEL CDE;
TSP;
.
CPASS KLM;
ELABEL;
```

3.8.5 Comments

You can put comments anywhere within the Endeavor/INFO table.

A comment must begin with the character string `/*` and must end with the character string `*/`. Comments can span multiple lines. The terminating character string, `*/`, must be on the same line.

Comments must appear after the semicolon that ends a statement.

3.8.5.1 Example of Single-Line Comments

Here is an example of single-line comments:

```
CTRL;
  IACT (TS0B06I);    /* INQUIRY CHANGE FOR ACTIONS */
  RACT (TS0B06R);    /* RETRIEVE CHANGE FOR ACTIONS */
  CACT (TS0B06C);    /* CREATE CHANGE FOR ACTIONS */
  UACT (TS0B06U);    /* UPDATE CHANGE FOR ACTIONS */
ECTRL;
```

3.8.5.2 Example of Multiple-Line Comments

Here is an example of a comment that spans multiple lines:

```
POPULATE;
  /*****
  * PREQCOMM: PACKAGE COMMENT *
  * PREQWSD: EXECUTION WINDOW START DATE *
  * PREQWST: EXECUTION WINDOW START TIME *
  * PREQWED: EXECUTION WINDOW END DATE *
  * PREQWET: EXECUTION WINDOW END TIME *
  * PECBUSER: CURRENT USER ID *
  * PECBFNM: PACKAGE FUNCTION *
  * PECBBANM: BEFORE OR AFTER *
  * PECBNDRC: NDVR HIGH RETURN CODE *
  * PHDRSTAT: PACKAGE STATUS *
  * PHDRCRD: PACKAGE CREATION DATE *
  * PHDRCRT: PACKAGE CREATION TIME *
  *****/
```

In this example, the starting string (`/*`) and the ending string (`*/`) of the comment are shown in bold.

3.8.5.3 Example of Incorrectly-Coded Comment

The comment below is invalid because it appears between the end of the statement and the semicolon that ends the statement.

```
CTRL;
  IACT (TS0B06I) /* INQUIRY CHANGE FOR ACTIONS */;
```

The correct way to code this comment is shown below:

```
CTRL;
  IACT (TS0B06I); /* INQUIRY CHANGE FOR ACTIONS */
```

3.8.6 RC Statements

Use RC statements to specify a return code back to Endeavor. This statement may appear anywhere within a Label block. The syntax is:

```
RC (return code);
```

Acceptable return code values are 0 (success) or 8 (failure). If the interface returns an “8” on a before-action or before-package exit, Endeavor aborts the function.

Multiple RC statements are allowed within the same block. When multiple RC statements are used, the interface returns the highest value.

3.8.6.1 Examples

Here is an example of an RC statement at the end of a Label block:

```
LABEL MOV0010;      /* CHANGE RECORD NOT FOUND: CREATE ONE */
  CRITERIA;
  ECRITERIA;
  POPULATE;
    FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=USER);
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
    FIELD PANEL(BLG6DSAB) INDEX(S0E0F)
      VALUE('API CREATE CHG RECORD')
  EPOPULATE;
  RC(0);
ELABEL;
```

Here is an example of an RC statement at the start of a Label block:

```
LABEL MOV0020;      /* CHANGE RECORD FOUND */
  RC (0);
  CRITERIA;
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
  ECRITERIA;
  POPULATE;
  EPOPULATE;
  CFAIL MOV0022;
ELABEL;
```

In both cases, the RC statement instructs the interface to send a return code of 0 back to Endeavor when it has completed the processing specified in the Label block.

3.8.7 MSG Statements

Use message (MSG) statements to include a message in the Endeavor Execution Report. This statement may appear anywhere within a Label block. Multiple occurrences are allowed.

The syntax is:

```
MSG ('text');
```

The text must be enclosed in single quotes. The maximum length of a message is 100 characters. Expanded text in excess of 100 characters is truncated.

Note: Endeavor keywords can be included in message text. The report prints with the corresponding Endeavor term. For example:

```
MSG ('ACTIVITY RECORD FOR CCID: =CCID MISSING');
```

When this message prints on the execution report, =CCID is replaced with the value of the CCID for which the record was not found.

3.8.7.1 Example

Here is an example of a MSG statement:

```
LABEL DEL0022;          /* STATUS NOT OPEN - CHECK FOR INITIAL */
  CRITERIA;
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('INITIAL');
  IFNO;
    RC (8);
    MSG ('INVALID STATUS - MUST BE INITIAL OR OPEN');
  EIF;
  ECRITERIA;
  POPULATE;
  EPOPULATE;
ELABEL;
```

3.9 Criteria Block

Use a Criteria block to specify a test that will result in a true/false condition. A test is established by including FIELD statements in this block. If a true condition prevails, the interface populates the Info/Man record with information from other FIELD statements, found in the Populate block.

When no test is specified in the Criteria block, a true condition is assumed to exist, and the Populate block FIELD statements are executed automatically.

Criteria blocks are optional. One Criteria block is allowed per Label block.

The syntax is:

```
CRITERIA;
    FIELD statements
    IFYES/IFNO statements
ECRITERIA;
```

The components of this syntax are described below.

Statement	Description
CRITERIA	Required. CRITERIA must be the first word in a Criteria block, and must be followed by a semicolon.
FIELD statements	FIELD statements establish the conditions for populating Info/Man records. FIELD statements are described in a later section.
IFYES/IFNO statements	Valid only in Criteria block FIELD statements. Used to specify processing when the test established in the FIELD statement is true (IFYES) or false (IFNO). IFYES/IFNO statements are described in a later section.
ECRITERIA	Required. ECRITERIA must be the last word in a Criteria block, and must be followed by a semicolon.

If you want to populate an Info/Man record at this exit point without testing any fields, you must code a null Criteria block. A null Criteria block looks like this:

```
CRITERIA;
ECRITERIA;
```

When coding Criteria blocks, keep in mind that the interface supports storing information only in Info/Man X-type records (freeform text) and R-type records (Response records). Of these two record types, the interface allows you to retrieve for scanning only the R-type records. This means that FIELD statements in Criteria blocks should examine only Info/Man fields that contain R-type records.

For information about X-type records and R-type records, refer to the Info/Man documentation.

3.9.1.1 Example 1

Here is an example when there is no criteria, but the user wants to populate an Info/Man record. This use of the CRITERIA statement is common within a Label block associated with a NOTFOUND statement, as shown below.

```
USE (ARCHIVE, BEFORE, STANDALONE);
NOTFOUND (CREATE,ARC0010);
FOUND (ARC0020);
LABEL ARC0010;          /* CHANGE RECORD NOT FOUND: CREATE ONE */
CRITERIA;
ECRITERIA;
POPULATE;
    FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=USER);
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
    FIELD PANEL(BLG6DSAB) INDEX(S0E0F)
        VALUE('API CREATE CHG RECORD');
EPOPULATE;
RC(0);
ELABEL;
```

3.9.1.2 Example 2

Here is an example of FIELD statements in both the Criteria and Populate blocks.

```
LABEL CFM0010;          /* SHIP CONFIRM AFTER - LOG SECTION      */
CRITERIA;
    FIELD (=PECBSFNM) VALUE ('CONFIRM ');
ECRITERIA;
POPULATE;
    /******
    * PREQDEST: SHIP DESTINATION          *
    * PREQSCNF: SHIP CONFIRMATION TYPE   *
    * PREQSRES: SHIP CONFIRMATION RESULTS *
    * PREQSRCV: SHIP CONFIRM RC VALUE     *
    /******/
    FIELD PANEL(BLG0C010) INDEX(S0E01)
    TEXT ('SHIP: DESTINATION =PREQDEST CONFIRM TYPE =PREQS CNF') -
        (' CONFIRMATION RESULTS =PREQSRES =PREQSRCV');
EPOPULATE;
ELABEL;
```

3.9.1.3 Example 3

Here is an example that includes IFYES/IFNO conditions in the Criteria block, in addition to FIELD statements.

```
LABEL RET0022;      /* STATUS NOT OPEN - CHECK FOR INITIAL */
  CRITERIA;
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('INITIAL');
    IFNO;
      RC (8);
      MSG ('INVALID STATUS - MUST BE INITIAL OR OPEN');
    EIF;
  ECRITERIA;
    POPULATE;
    EPOPULATE;
ELABEL;
```

3.10 Populate Block

Use a Populate block to identify the Endeavor information to be logged on an Info/Man database record. Populate blocks can only contain FIELD statements.

One Populate block is allowed per Label block, and must follow the Criteria block. If you include a Populate block, you must also include a Criteria block.

Note: A Label block is not required to have the Criteria/Populate blocks.

The syntax is:

```
POPULATE;  
    FIELD statements  
EPOPULATE;
```

These components are described below:

Statement	Description
POPULATE	Required. POPULATE must be the first word in a Populate block, and must be followed by a semicolon
FIELD statements	FIELD statements establish the information to be recorded in Info/Man records. FIELD statements are described in a later section.
EPOPULATE	Required. EPOPULATE must be the last word in a Criteria block, and must be followed by a semicolon.

If you do not want to populate an Info/Man record at this exit point, you must code a null Populate block. A null populate block looks like this:

```
POPULATE;  
EPOPULATE;
```

Note: The interface supports only Info/Man X-type records (freeform text) and R-type records (Response records). This means that when coding FIELD statements in a Populate block, you can refer only to Info/Man fields that store X-type or R-type information.

For information about X-type records and R-type records, refer to the Info/Man documentation.

3.10.1.1 Example 1

Here is an example of a null Populate block:

```
LABEL RET0022;      /* STATUS NOT OPEN - CHECK FOR INITIAL */
  CRITERIA;
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('INITIAL');
    IFNO;
      RC (8);
      MSG ('INVALID STATUS - MUST BE INITIAL OR OPEN');
    EIF;
  ECRITERIA;
  POPULATE;
  EPOPULATE;
ELABEL;
```

3.10.1.2 Example 2

Here is an example of a Populate block:

```
LABEL CFM0010;      /* SHIP CONFIRM AFTER - LOG SECTION      */
  CRITERIA;
    FIELD (=PECBSFNM) VALUE ('CONFIRM ');
  ECRITERIA;
  POPULATE;
  /*****
  * PREQDEST:  SHIP DESTINATION                *
  * PREQSCNF:  SHIP CONFIRMATION TYPE          *
  * PREQSRES:  SHIP CONFIRMATION RESULTS       *
  * PREQSRCV:  SHIP CONFIRM RC VALUE           *
  *****/
  FIELD PANEL(BLG0C010) INDEX(S0E01)
  TEXT ('SHIP: DESTINATION =PREQDEST CONFIRM TYPE =PREQSCNF') -
    ('          CONFIRMATION RESULTS =PREQSRES =PREQSRCV');
  EPOPULATE;
ELABEL;
```

3.11 FIELD Statements

FIELD statements may only appear in Criteria and Populate blocks.

- The interface uses Criteria block FIELD statements to determine a true or false condition. A true condition results in the execution of the Populate Block.
- When FIELD statements appear in the Populate block, the interface uses them to update an Info/Man database record.

3.11.1 FIELD Statement Syntax

FIELD statement syntax differs slightly depending on whether the syntax is for a Criteria Block or a Populate block. When coding, values within parentheses must remain on the same line.

Criteria FIELD statement syntax to test an Info/Man field is:

```
FIELD PANEL(pname) INDEX(s-word) [LENGTH(length)]
  VALUE({Endevor field|'literals'|NONBLANK|BLANK});
  [IFYES statement] [IFNO statement]
```

Note: You cannot specify FIELD statements in a Criteria block associated with a Create Activity Use block.

Criteria FIELD statement syntax to test an Endeavor field is:

```
FIELD (Endevor field) [LENGTH(length)]
  VALUE({'literal'|NONBLANK|BLANK});
  [IFYES statement] [IFNO statement]
```

Populate FIELD statement syntax is:

```
FIELD PANEL(pname) INDEX(s-word) [LENGTH(length)]
  VALUE({Endevor field|'literals'|=DATE|=TIME});
```

or

```
FIELD PANEL(pname) INDEX(s-word) TEXT(text);
```

Note: Population of Endeavor field control blocks is not allowed.

The syntax components are explained in the next section.

3.11.2 Field Panel (Pname) and Index (S-Word)

Together, the FIELD PANEL and INDEX clauses identify a particular Info/Man record field.

- When used in a Criteria block FIELD statement, this identifies the field against which the string in the VALUE clause is tested.
- When used in a Populate block FIELD statement, this identifies the field to be populated by the string in the VALUE or TEXT clause.

The Pname identifies the Info/Man panel containing the field to be updated. The S-word identifies the field to be updated. These fields must also be defined in the appropriate PIDT table.

3.11.3 Length

Length is a numeric value that must be greater than zero but less than or equal to 256. The length value determines both the number of characters moved during a MOVE action, and the number of characters compared to determine the true or false condition.

The table below shows how the MOVE action and compare logic use the length value.

If the LENGTH value is	Then the MOVE action or compare logic uses
Greater than the target.	The target length.
Less than the target.	The LENGTH value, with any remaining fields padded with blanks.
Omitted.	The smaller of source or target.

Note: When Endeavor fields are converted from binary to character form, all leading zeros are stripped. For example, 'x0010' becomes 'C'16'. During a compare operation between an Info/Man field and an Endeavor field that has undergone a binary to character conversion, all leading zeros in the Info/Man field are stripped before the comparison.

3.11.4 Value Clause--Criteria Block

When used in a Criteria block FIELD statement, the VALUE clause establishes the criteria against which the Info/Man or Endeavor field is tested. Acceptable test criteria include the following.

Criterion	Description
Endeavor field	A value starting with an equal sign identifies an Endeavor field. Example: VALUE (=PECBUSER) Valid when testing Info/Man fields only.
Literal	Alphanumeric values enclosed in single quotes. Example: VALUE ('UPDATE ')
Blanks	Use this value to test whether the field is empty. A true condition exists when the field contains blanks.
Nonblank	Use this value to test for the presence of information in a field. A true condition exists when the field contains any nonblank characters.

3.11.5 Value Clause--Populate Block

When used in a Populate block FIELD statement, the VALUE clause identifies the value to be used to populate the Info/Man field. Acceptable populate values include the following.

Criterion	Description
Endevor field	A value starting with an equal sign identifies an Endevor field. Example: VALUE (=PECBUSER)
Literal	Alphanumeric values enclosed in single quotes. Example: VALUE ('UPDATE ')
=DATE	Represents the current date. When used, the current date is retrieved to populate an Info/Man database field. Format: mm/dd/yy.
=TIME	Represents the current time. When used, the current time is retrieved to populate an Info/Man database field. Format: hh:mm.

3.11.6 IFYES/IFNO Statement

IFYES and IFNO clauses may only be used in FIELD statements within the Criteria block.

- The IFYES statement is invoked when the test of the Info/Man or Endevor field results in a true condition.
- The IFNO statement is invoked when the test of the Info/Man or Endevor field results in a false condition.

IFYES syntax is:

```
IFYES;
  [MSG (text);] [RC (return code);]
EIF;
```

IFNO syntax is:

```
IFNO;
  [MSG (text);] [RC (return code);]
EIF;
```

3.11.7 TEXT Clauses

Use TEXT clauses to specify information to be used to populate Info/Man fields. You must use this format against Info/Man fields defined as freeform text fields. This format is allowed against other field types.

The text data must be enclosed in single quotes. You can specify text to continue on the same line.

For example:

```
TEXT ('This is an example of a free form text build setup ') -
('statement line one')
```

This text clause is recorded in Info/Man as follows:

```
This is an example of a free form text build setup statement line oe
```

You can also specify text on separate lines. For example:

```
TEXT ('This is an example of a free form text build setup ')
TEXT ('Statement line two')
TEXT ('This is statement line three - Package RC =PECBNVR')
```

This text clause is recorded in Info/Man as follows:

```
This is an example of a free form text build setup
Statement line two
This is statement line three - Package RC 0
```

You can include an Endeavor field equate in specified message text:

```
TEXT('CCID: =PACTCCID IN PACKAGE: =PECBPKID');
```

The text recorded in Info/Man for this example includes the CCID for the package action in place of the =PACTCCID equate, and the ID of the associated package in place of the =PECBPKID equate.

3.11.8 Example 1. FIELD Statement in a Criteria Block--Info/Man Field

```
LABEL RET0022;          /* STATUS NOT OPEN - CHECK FOR INITIAL */
CRITERIA;
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('INITIAL');
  IFNO;
  RC (8);
  MSG ('INVALID STATUS - MUST BE INITIAL');
  EIF;
ECRITERIA;
  POPULATE;
  EPOPULATE;
ELABEL;
```

3.11.9 Example 2. FIELD Statement in a Criteria Block--Endeavor Field

```
LABEL CRT0020;          /* LOOKING FOR ACTION: ADD          */
CRITERIA;
  FIELD (=PACTACTN) VALUE('ADD      ');
ECRITERIA;
  POPULATE;
  EPOPULATE;
```

3.11.10 Example 3. FIELD Statement with Equates USE (CREATE, ACTIVITY)

```

LABEL CRT0010;          /* MAINLINE - ALL PKG ACTION POINTS.      */
CRITERIA;
ECRITERIA;
POPULATE;
    FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=PECBUSER);
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
    FIELD PANEL(BLG6DSAB) INDEX(S0E0F)
        TEXT('CCID: =PACTCCID IN PACKAGE: =PECBPKID');
    FIELD PANEL(BLG6PTYP) INDEX(S0C09) VALUE('PKG/ACT');
EPOPULATE;
CPASS CRT0020;
ELABEL;

```

3.11.11 Example 4. FIELD Statement with Text

```

LABEL GRP0020;          /* FOR PACKAGE ACTIONS: GENERATE      */
RC (0);                /*                                */
CRITERIA;
ECRITERIA;
POPULATE;
    /******
    * PACTACTN: ACTION NAME          *
    * PACTCOMM: ACTION COMMENT      *
    * PACTSENV: ENVIRONMENT NAME    *
    * PACTSSYS: SYSTEM NAME         *
    * PACTSSBS: SUBSYSTEM NAME      *
    * PACTSTYP: TYPE                *
    * PACTSSTG: STAGE NAME          *
    * PACTSELM: ELEMENT NAME        *
    *****/
    FIELD PANEL(BLG0C030) INDEX(S0E01)
    TEXT ('ACTION: =PACTACTN =PACTCOMM')
    TEXT ('ELEMENT: =PACTSENV / =PACTSSYS / =PACTSSBS / ') -
    ('=PACTSTYP / =PACTSSTG / =PACTSELM ');
EPOPULATE;
ELABEL;

```

3.12 Endeavor Control Block Availability

The Endeavor/INFO table uses information stored in exit control blocks when building requests to pass to the Info/Man API. You identify the control block fields you want to use, and the target Info/Man database fields for that information, when you code the Criteria and Populate blocks for your Endeavor/INFO table.

3.12.1 Control Blocks for Stand-alone Actions

The information in the \$ECBDS and \$REQPDS control blocks is available to all stand-alone actions. In addition, the information in the environment (\$ENVDS), element (\$ELMDS), and file (\$FILDS) control blocks is available for either the source (SRC) or the target (TGT) location of an action. The table below summarizes which control blocks are available to which groups of actions.

Actions that can appear together in Use blocks	\$ENVDS		\$ELMDS		\$FILDS		\$ECBDS		\$REQPDS	
	src	tgt	src	tgt	src	tgt	src	tgt	src	tgt
ADD, UPDATE		x		x	x			x		x
RETRIEVE, ARCHIVE	x		x			x		x		x
GENERATE, DELETE	x		x					x		x
MOVE, TRANSFER, RESTORE	x	x	x	x				x		x

The sections that follow explain the specific fields from each control block, and the equate values used in the Endeavor/INFO table syntax to reference these fields.

3.12.2 Available \$ENVDS Fields

The table below lists the fields in the \$ENVDS control block that are available to the Endeavor/INFO interface, the information contained in each field, and the equate value to use in the syntax for the Endeavor/INFO table for both source and target locations.

This \$ENVDS field	Contains this source or target information	And has these equate values	
		Source	Target
ENVSITE	Endeavor site ID.	EV1SITE	EV2SITE
ENVENVM	Environment name.	EV1ENV	EV2ENV
ENVSYSYTM	System name.	EV1SYS	EV2SYS
ENVSUBSY	Subsystem name.	EV1SUB	EV2SUB
ENVTYPE	Type name.	EV1TYP	EV2TYP
ENVSTAGE	Stage name.	EV1STAGE	EV2STAGE
ENVSTGID	Stage number for the action (1 or 2).	EV1STG#	EV2STG#
ENVSTGCD	Stage ID.	EV1STGID	EV2STGID
ENVELEMT	Name of the element.	EV1ELM	EV2ELM
ENVEVER	Version number for the element.	EV1VER	EV2VER
ENVELVL	Level for the element.	EV1LVL	EV2LVL
ENVSPEC	Indicates the kind of location. The location can be: <ul style="list-style-type: none"> ■ An Endeavor environment. ■ External to Endeavor ■ An archive data set. 	EV1LOC	EV2LOC
ENVEAOFF	Actual element name length.	EV1ELMLN	EV2ELMLN
ENVEAOFF	Full element name.	EV1ELMFN	EV2ELMFN

3.12.3 Available \$ELMDS Fields

The table below lists the fields in the \$ELMDS control block that are available to the Endeavor/INFO interface, the information contained in each field, and the equate value to use in the syntax for the Endeavor/INFO table for both source and target locations.

This \$ELMDS field	Contains this source or target information	And has these equate values	
		Source	Target
ELMNAME	First 10 characters of the element name.	EL1NAME	EL2NAME
ELMMVERS	Element version.	EL1VER	EL2VER
ELMMLLVL	Element level.	EL1LVL	EL2LVL
ELMM1STL	Base level of the element at this location.	EL1BASE	EL2BASE
ELMMINS	Number of inserts.	EL1INS	EL2INS
ELMMDEL	Number of deletes.	EL1DEL	EL2DEL
ELMCCID	CCID associated with the element.	EL1CCID	EL2CCID
ELMMBTOT	Number of statements in the base level of the element.	EL1BTOT	EL2BTOT
ELMMBDTE	Date associated with the base level.	EL1BDATE	EL2BDATE
ELMMBTIM	Time associated with the base level.	EL1BTIME	EL2BTIME
ELMBCOM	Comment associated with the base level.	EL1BCOM	EL2BCOM
ELMBUSID	User ID associated with the base level.	EL1BUSER	EL2BUSER
ELMPUSID	User ID associated with the last generate of the element.	EL1GUSER	EL2GUSER
ELMPDSTP	Date the element was last generated.	EL1GDATE	EL2GDATE
ELMPTSTP	Time the element was last generated.	EL1GTIME	EL2GTIME

This \$ELMDS field	Contains this source or target information	And has these equate values
ELMPRFLG:	Last processor executed against the element: <ul style="list-style-type: none"> ■ 0--no processor executed ■ 1--processor failed ■ 2--last processor was generate ■ 3--last processor was move ■ 4--last processor was delete ■ 5--restore processor not specified 	EL1PFG EL2PFG
ELMLPROD	Date of last processor execution.	EL1PDATE EL2PDATE
ELMLPROT	Time of last processor execution.	EL1PTIME EL2PTIME
ELMLPROU	User ID associated with last processor executed against the element.	EL1PUSER EL2PUSER
ELMLPRON	Name of the processor last run against the element.	EL1PNAME EL2PNAME
ELMMPRC	Processor return code.	EL1PRC EL2PRC
ELMPCOM	Comment associated with the last action that generated the element.	EL1PCOM EL2PCOM
ELMMRC	Endeavor return code (NDVR RC) for the last execution of a processor against the element.	EL1CRC EL2CRC
ELMFDSN	Name of the data set from which the element was last added, updated, or restored.	EL1DSN EL2DSN
ELMFMBR	Member name of the element before it was last added, updated, or restored.	EL1MBR EL2MBR
ELMMVDTE	Date element was last moved.	EL1MDATE EL2MDATE
ELMMVTIM	Time element was last moved.	EL1MTIME EL2MTIME
ELMMUSID	User ID associated with the last move of the element.	EL1MUSER EL2MUSER

This \$ELMDS field	Contains this source or target information	And has these equate values
ELMRDSTP	Date element was last retrieved.	EL1RDATE EL2RDATE
ELMRTSTP	Time element was last retrieved.	EL1RTIME EL2RTIME
ELMRIID	User ID associated with the last retrieve of the element.	EL1RUSER EL2RUSER
ELMRCOM	Comment associated with the last retrieve of the element.	EL1RCOM EL2RCOM
ELMTDSN	Name of the data set to which the element was last retrieved.	EL1RDSN EL2RDSN
ELMTMBR	Member name of the element at the target data set of the last retrieve.	EL1RMBR EL2RMBR
ELMCCOM	Current level comment for the element.	EL1CCOM EL2CCOM
ELMMLDTE	Date associated with the current level of the element.	EL1LDATE EL2LDATE
ELMMLTME	Time associated with the current level of the element.	EL1LTIME EL2LTIME
ELMLUSID	User ID associated with the current level of the element.	EL1LUSER EL2LUSER
ELMMLACT	Most recent action against the element.	EL1LACT EL2LACT
ELMMLTOT	Total number of source statements in the current level of the element.	EL1LTOT EL2LTOT
ELMRCCID	CCID associated with the last RETRIEVE action against the element.	EL1RCCID EL2RCCID
ELMGCCID	CCID associated with the last GENERATE action against the element.	EL1GCCID EL2GCCID
ELMODACT	Last action executed against the element that modified the element.	EL1LMACT EL2LMACT
ELMLCCID	CCID associated with the last action that modified the element.	EL1LCCID EL2LCCID

This \$ELMDS field	Contains this source or target information	And has these equate values
ELMLCOMM	Comment associated with the last action that modified the element.	EL1LCOMM EL2LCOMM
ELMLADT	Date associated with the last action that modified the element.	EL1MLADT EL2MLADT
ELMLATM	Time associated with the last action that modified the element.	EL1MLATM EL2MLATM
ELMLUID	User ID associated with the last action that modified the element.	EL1MLUID EL2MLUID
ELMUPDT	Element name in the delta library.	EL1NUPDT EL2NUPDT
ELMBASE	Element name in the base library.	EL1NBASE EL2NBASE
ELMIPCTG	Percentage of inserts deleted.	EL1IPCTG EL2IPCTG
ELMDPCTG	Percentage of deletes reinserted.	EL1DPCTG EL2DPCTG
ELMFPESD	Name of footprinted object (ESD) module.	EL1FPESD EL2FPESD
ELMEDFMT	Delta format for the element.	EL1EDFMT EL2EDFMT
ELMENPAC	Indicates whether element base is packed.	EL1ENPAC EL2ENPAC
ELMLPROV	Version number of last processor used against this element.	EL1LPROV EL2LPROV
ELMPROL	Level number of last processor used against this element.	EL1PROL EL2PROL

This \$ELMDS field	Contains this source or target information	And has these equate values
ELMFMID	Record format ID: <ul style="list-style-type: none"> ▪ 21 Endeavor for MVS Release 2.1 ▪ 25 Endeavor for MVS Release 2.5 ▪ 35 Endeavor for MVS Release 3.5 ▪ 36 Endeavor for MVS Release 3.6 	EL1FMID EL2FMID
ELMXDNAM	Component list delta member name.	EL1XDNAM EL2XDNAM
ELMXBVER	Component list delta version number.	EL1XBVER EL2XBVER
ELMXBTOT	Component list base total.	EL1XBTOT EL2XBTOT
ELMXLTOT	Component list last level total.	EL1XLTOT EL2XLTOT
ELMXBLVL	Component list base level number.	EL1XBLVL EL2XBLVL
ELMXLLVL	Component list last level number.	EL1XLLVL EL2XLLVL
ELMXINS	Inserts at last level.	EL1XINS EL2XINS
ELMXDLS	Deletes at last level.	EL1XDLS EL2XDLS
ELMXRGIN	Component list regression insert percent.	EL1XRGIN EL2XRGIN
ELMXRGDL	Component list regression delete percent.	EL1XRGDL EL2XRGDL
ELMXBDTE	Component list base date.	EL1XBDTE EL2XBDTE
ELMXBTIM	Component list base time.	EL1XBTIM EL2XBTIM
ELMXLDTE	Component list last level date.	EL1XLDTE EL2XLDTE
ELMXLTIM	Component list last level time.	EL1XLTIM EL2XLTIM
ELMXFLAG	Component list component status flag.	EL1XFLAG EL2XFLAG
ELMXDFMT	Component list delta format.	EL1XDFMT EL2XDFMT
ELMXUPDT	Indicates whether component list base is in delta library.	EL1XUPDT EL2XUPDT

This \$ELMDS field	Contains this source or target information	And has these equate values
ELMSPKG	ID of the last processed package that included the source for this element.	EL1SPKG EL2SPKG
ELMSPKGT	Time stamp of the last processed package that included the source for this element.	EL1SPKGT EL2SPKGT
ELMOPKG	ID of package associated with the outputs of this element.	EL1OPKG EL2OPKG
ELMOPKGT	Time stamp of package associated with the outputs of this element	EL1OPKGT EL2OPKGT
ELMPRGRP	Processor group name.	EL1PRGRP EL2PRGRP
ELMFFFLG	From file type flag.	EL1FFFLG EL2FFFLG
ELMTFFLG	To file type flag.	EL1TFFLG EL2TFFLG
ELMFPAOFF	From path name length	EL1FPALN EL2FPALN
ELMFPAOFF	From path name	EL1FPANM EL2FPANM
ELMFNAOFF	From file name length	EL1FFILN EL2FFILN
ELMFNAOFF	From file name	EL1FFINM EL2FFINM
ELMTPAOFF	To path name length	EL1TPALN EL2TPALN
ELMTPAOFF	To path name	EL1TPANM EL2TPANM
ELMTNAOFF	To file name length	EL1TFILN EL2TFILN
ELMTNAOFF	To file name.	EL1TFINM EL2TFINM
ELMEAOFF	Element name length.	EL1ELMLN EL2ELMLN
ELMEAOFF	Full element name.	EL1ELMFN EL2ELMFN

3.12.4 Available \$FILDS Fields

Use these equate values for \$FILDS fields.

This \$FILDS field	Contains this source or target information	And has these equate values
		Source Target
FILDSN	Data set name of the external file used by the current action.	FI1DSN FI2DSN

This \$FILDS field	Contains this source or target information	And has these equate values
FILDSMEM	Member name of the element associated with the current action.	FI1MBR FI2MBR
FILDDN	DDname associated with the data set named above.	FI1DDN FI2DDN
FILDSTY	The kind of data set (DA, PSU, PO, IS, and the like)	FI1DTY FI2DTY
FILPAOFF	Path name length.	FI1PALN FI2PALN
FILPAOFF	Path name.	FI1PANM FI2PANM
FILNAOFF	File name length.	FI1FILN FI2FILN
FILNAOFF	File name.	FI1FINM FI2FINM

3.12.5 Available \$ECBDS Fields

Use these equate values for \$ECBDS fields.

This \$ECBDS field	Contains this information	And has this equate value
ECBEXTN	Number identifying the exit being invoked.	EXITNO
ECBUSER	User ID associated with the current session. Modifiable by exit 5.	USER
ECBMODE	Indicates whether the current action was requested in foreground (T) or in batch (B).	MODE
ECBFUNC	Action code. For example, 1 is the code for ADD, 7 is the code for MOVE.	ACTID
ECBFUNAM	Literal describing the current action (for example ADD, MOVE).	ACTION

Note: The EXITRC equate value maps to the ECBRTCD field in the \$ECBDS control block. This field is updated by the RC keyword value in the Endeavor/INFO table syntax.

3.12.6 Available \$REQDS Fields

Use these equate values for \$REQDS fields.

This \$REQDS field	Contains this information	And has this equate value
REQCOMM	Comment associated with the current action.	COMM
REQCCID	CCID associated with the current action.	CCID
REQSISOF	Indicates whether the current action requested a signout override.	SISO
REQSISOC	Applicable for RETRIEVE action. Indicates whether NO SIGNOUT was specified.	COPY
REQEXPND	Applicable for RETRIEVE action. Indicates whether EXPAND INCLUDES was specified.	EXPAND
REQTCOD	Action return code.	ACTRC
REQOVRWR	Applicable for RETRIEVE action. Indicates whether REPLACE was specified.	REPL
REQDEL	Applicable for ADD, ARCHIVE, DELETE, MOVE, TRANSFER, and UPDATE. Indicates whether or not a member or element is deleted after completing the current action.	DEL
REQMOVWH	Applicable for MOVE action. Indicates whether MOVE WITH HISTORY was specified.	WHIST
REQUPDT	Applicable for ADD action. Indicates whether UPDATE IF PRESENT was specified.	UPD
REQOSRCH	Applicable for GENERATE and RETRIEVE actions. Indicates whether Endeavor is to search the map when building a list.	SEARCH
REQORSGN	Applicable for MOVE and TRANSFER actions. Indicates whether RETAIN SIGNOUT was specified.	RETSIGN
REQOJUMP	Applicable for MOVE action. Indicates whether Endeavor allows you to move an element when a version of the element exists at an intermediate stage that is not on the map.	JUMP
REQOSGNI	Applicable for MOVE and TRANSFER actions. Indicates whether or not to sign the target location was specified.	SIGNIN

This \$REQDS field	Contains this information	And has this equate value
REQSYNC	Applicable for TRANSFER actions. Indicates whether you want to transfer an element with history when the current level of the element at the target differs from the base level at the from-location.	SYNC

3.12.7 Control Blocks for Packages and Package Actions

The control block information available to package actions is shown below.

Package actions that can appear together in Use blocks	\$PECBDS	\$PREQPDS	\$PHDRDS	\$PACTREQ
BACKIN, BACKOUT, CAST, COMMIT, EXECUTE, RESET, REVIEW, SHIP for before-exit and after-exit points	x	x	x	
CREATE /MODIFY				
before	x	x		
after	x	x	x	
CREATE ACTIVITY	x	x	x	x
GENPKGID				
before	x	x		
after	x	x		
PDELETE				
before	x	x	x	
after	x	x		

Not all the fields in the \$PECBDS control block are available. Those fields that are available are shown in the table below.

This \$PECBDS field	Contains this information	And has this equate value
PECBPKID	Package ID.	PECBPKID
PECBFNCD	Package function code	PECBFNCD
PECBSFCD	Package subfunction code.	PECBFSCD
PECBBACD	Package before/after code.	PECBBACD
PECBFNNM	Package function literal	PECBFNNM
PECBSFNM	Package subfunction literal.	PECBSFNM
PECBBANM	Package before/after literal.	PECBBANM
PECBACTE	Action record existence flag.	PECBACTE
PECBAPPE	Approver record existence flag.	PECBAPPE
PECBBODE	Backout record existence flag.	PECBBODE
PECBNDRC	Endevor high return code.	PECBNDRC
PECBNDAB	Endevor abend code.	PECBNDAB
PECBNDMG	Endevor message text.	PECBNDMG
PECBNDML	Endevor message text length.	PECBNDML
PECBUSER	TSO user or batch job name.	PECBUSER
PECBMODE	Mode setting (online or batch).	PECBMODE

The fields not available include:

- The before/after exit switches (PECBBIBE through PECBSCAF)
- The request for data flags (PECBACTR through PECBSCLR), as well as PECBRQRC, the return code field for user requests for data.
- PECBUECB, PECBUREQ, PECBUFIL, PECBUAPP, PECB#APG.

The following fields are set by the Endeavor/INFO interface, through the table syntax:

- PECBRTCD
- PECBMGCD
- PECBMGLN
- PECBMSG

3.12.8 \$PREQPDS

Use these equate values for \$PREQPDS fields.

This \$PREQPDS field	Contains this information	And has this equate value
PREQCOMM	Package comment.	PREQCOMM
PREQBOEN	Backout enabled flag.	PREQBOEN
PREQDEL	Delete enabled flag.	PREQDEL
PREQPSHR	Sharable package flag.	PREQPSHR
PREQPAPD	Append package flag.	PREQPAPD
PREQEWS	Execution window start date.	PREQEWS
PREQEWST	Execution window start time.	PREQEWST
PREQEWED	Execution window end date.	PREQEWED
PREQEWET	Execution window end time.	PREQEWET
PREQIPKG	Source package ID (copy subfunction).	PREQIPKG
PREQDEST	Shipment destination.	PREQDEST
PREQSTYP	Shipment type.	PREQSTYP
PREQSCMP	Ship complementary data set flag.	PREQSCMP
PREQSCNF	Ship confirmation type.	PREQSCNF
PREQSRES	Ship confirmation result.	PREQSRES
PREQSRCV	Ship confirmation return code.	PREQSRCV
PREQAPGP	Override approver group.	PREQAPGP
PREQAPID	Override approver ID.	PREQAPID

3.12.9 \$PHDRDS

Use these equate values for \$PHDRDS fields.

This \$PHRDS field	Contains this information	And has this equate value
PHDRTYPE	Package type (standard or emergency).	PHDRTYPE
PHDRSTAT	Package status.	PHDRSTAT
PHDRBOST	Package backout status.	PHDRBOST

This \$PHRDS field	Contains this information	And has this equate value
PHDRCRD	Package creation date.	PHDRCRD
PHDRCRT	Package creation time.	PHDRCRT
PHDRXRC	Package execution return code.	PHDRXRC
PHDRPUD	Date of last update to package SCL.	PHDRPUD
PHDRPUT	Time of last update to package SCL.	PHDRPUT
PHDRCD	Cast date.	PHDRCD
PHDRCT	Cast time.	PHDRCT
PHDRAD	Final approval/denial date.	PHDRAD
PHDRAT	Final approval/denial time.	PHDRAT
PHDRWSD	Execution window start date.	PHDRWSD
PHDRWST	Execution window start time.	PHDRWST
PHDRWED	Execution window end date.	PHDRWED
PHDRWET	Execution window end time.	PHDRWET
PHDRXD	Execution date.	PHDRXD
PHDRXT	Execution time.	PHDRXT
PHDRCMD	Commit date.	PHDRCMD
PHDRCMT	Commit time.	PHDRCMT
PHDRBOD	Backout date.	PHDRBOD
PHDRBOT	Backout time.	PHDRBOT
PHDRBID	Backin date.	PHDRBID
PHDRBIT	Backin time.	PHDRBIT
PHDREXD	Date when execution completed.	PHDREXD
PHDREXT	Time when execution completed.	PHDREXT

3.12.10 \$PACTREQ

Use these equate values for \$PACTREQ fields.

This \$PACTREQ field	Contains this information	And has this equate value
PACTSEQ#	Sequence number of the action.	PACTSEQ#
PACTACTN	Endevor action.	PACTACTN

This \$PACTREQ field	Contains this information	And has this equate value
PACTCCID	Action CCID.	PACTCCID
PACTCOMM	Action comment.	PACTCOMM
PACTNDRC	Endevor high return code.	PACTNDRC
PACTPRRC	Processor high return code.	PACTPRRC
PACTBEXD	Begin execution date.	PACTBEXD
PACTBEXT	Begin execution time.	PACTBEXT
PACTEEXD	End execution date.	PACTEEXD
PACTEEXT	End execution time.	PACTEEXT
PACTSSIT	Site ID at the source of the action.	PACTSSIT
PACTSENV	Environment at the source location.	PACTSENV
PACTSSYS	System at the source location.	PACTSSYS
PACTSSBS	Subsystem at the source location.	PACTSSBS
PACTSELM	First 10 characters of the element name at the source location.	PACTSELM
PACTSTYP	Type at the source location.	PACTSTYP
PACTSSTG	Stage name at the source location.	PACTSSTG
PACTSSTI	Stage ID at the source location.	PACTSSTI
PACTSVL	Version/level at the source location.	PACTSVL
PACTSDD	Delta/base creation date at source location.	PACTSDD
PACTSDT	Delta/base creation time at source location.	PACTSDT
PACTSGD	Date of last GENERATE action at the source location.	PACTSGD
PACTSGT	Time of last GENERATE action at the source location.	PACTSGT
PACTSPD	Date of last processor run at the source location.	PACTSPD
PACTSPT	Time of last processor run at the source location.	PACTSPT
PACTSDSN	External data set name.	PACTSDSN
PACTSMBR	External member name.	PACTSMBR

This \$PACTREQ field	Contains this information	And has this equate value
PACTSPPI	Previous package ID associated with the source.	PACTSPPI
PACTSCTS	Cast time stamp associated with source location.	PACTSCTS
PACTTSIT	Site ID at the target of the action.	PACTTSIT
PACTTENV	Environment at the target location.	PACTTENV
PACTTSYS	System at the target location.	PACTTSYS
PACTTSBS	Subsystem at the target location.	PACTTSBS
PACTTELM	First 10 characters of the element name at the target location.	PACTTELM
PACTTTYP	Type at the target location.	PACTTTYP
PACTTSTG	Stage name at the target location.	PACTTSTG
PACTTSTI	Stage ID at the target location.	PACTTSTI
PACTTDD	Delta/base creation date at target location.	PACTTDD
PACTTDT	Delta/base creation time at target location.	PACTTDT
PACTTGD	Date of last Generate action at the target location.	PACTTGD
PACTTGT	Time of last Generate action at the target location.	PACTTGT
PACTTPD	Date of last processor run at the target location.	PACTTPD
PACTTPT	Time of the last processor run at the target location.	PACTTPT
PACTTDSN	External data set name at the target location.	PACTTDSN
PACTTMBR	External member name at the target location.	PACTTMBR
PACTTPPI	Previous package ID associated with the source at the target location.	PACTTPPI
PACTTCTS	Cast time stamp associated with source at the target location.	PACTTCTS
PACTSFLAG	Source file type indicator.	PACTSFLAG
PACTTFLAG	Target file type indicator.	PACTTFLAG

This \$PACTREQ field	Contains this information	And has this equate value
PACTSEAOFF	Source full element name length.	PACTSELMLN
PACTSEAOFF	Source full element name.	PACTSELMFN
PACTSPAOFF	Source path name length.	PACTSPALN
PACTSPAOFF	Source path name.	PACTSPANM
PACTSNAOFF	Source file name length.	PACTSFILN
PACTSNAOFF	Source file name.	PACTSFINM
PACTTEAOFF	Target full element name length.	PACTTEMLN
PACTTEAOFF	Target full element name.	PACTTELMFN
PACTTPAOFF	Target path name length.	PACTTPALN
PACTTPAOFF	Target path name.	PACTTPANM
PACTTNAOFF	Target file name length.	PACTTFILN
PACTTNAOFF	Target file name.	PACTTFINM

3.13 Interface Syntax Batch Utility Output Report

When the Endeavor/INFO interface utility builds a load module from the table syntax that you submit, it also produces an output report about the compile process. The output report contains the following sections:

- A syntax report, which lists the input table syntax.
- An error report, listing any messages produced by the utility.

3.13.1 Utility Syntax Report

The Utility Syntax Report lists the syntax input into the utility.

```

COPYRIGHT (C) COMPUTER ASSOCIATES, INC., 1987-2000                                24FEB00 10:30:51      PAGE 1
CA-ENDEAVOR/INTERFACE SYNTAX          RELEASE N.N  SERIAL XNNNNX
LINE # STATEMENT(S)
00001 TABLE NAME (NDVR);
00002 CTRL;
00003 IPKG (TS0B06I);          /* INQUIRY CHANGE TABLE FOR PACKAGES */
00004 IPKA(TS0B07I);          /* INQUIRY ACTIVITY TABLE FOR PKG ACTIONS */
00005 RPKG (TS0B06R);          /* RETRIEVE CHANGE TABLE FOR PACKAGES */
00006 RPKGA(TS0B07R);          /* RETRIEVE ACTIVITY TABLE FOR PKG ACTIONS */
00007 CPKG (TS0B06C);          /* CREATE CHANGE TABLE FOR PACKAGES */
00008 CPKGA(TS0B07C);          /* CREATE ACTIVITY TABLE FOR PKG ACTIONS */
00009 UPKG (TS0B06U);          /* UPDATE CHANGE TABLE FOR PACKAGES */
00010 UPKGA(TS0B07U);          /* UPDATE ACTIVITY TABLE FOR PKG ACTIONS */
00011 AACT (TS0B06A);          /* ADD CHANGE ACTIVITY TABLE FOR PKG ACTION*/
00012 ECTRL;
00013
00014
00015 /*****
00016 * EXIT POINTS: CREATE/AFTER  MODIFY/ AFTER  *
00017 *****/
00018 USE ((CREATE, AFTER),
00019       (MODIFY, AFTER));
00020
00021 NOTFOUND (CREATE,CMD0010);
00022 FOUND   (CMD0020);
00023
00024 LABEL CMD0010;          /* CHANGE RECORD NOT FOUND: CREATE ONE */
00025 CRITERIA;
00026 ECRITERIA;
00027 POPULATE;
00028
00029          /******
00030          * PECBUSER: CURRENT USER ID *
00031          *****/
00031          FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=PECBUSER);
00032          FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
00033 EPOPULATE;
00034 CPASS CMD0020;
00035 ELABEL;

```

3.13.2 Utility Error Report

Any errors in the batch utility syntax are reported in the Endeavor/INFO Interface Syntax Errors report. For example, the syntax shown below contains an incorrectly placed ECRITERIA statement in line 00033.

```

00015 /*****
00016 * EXIT POINTS: CREATE/AFTER  MODIFY/ AFTER  *
00017 *****/
00018 USE ((CREATE, AFTER),
00019      (MODIFY, AFTER));
00020
00021 NOTFOUND (CREATE,CMD0010);
00022 FOUND   (CMD0020);
00023
00024 LABEL CMD0010;      /* CHANGE RECORD NOT FOUND: CREATE ONE */
00025   CRITERIA;
00026   POPULATE;
00027
00028   /*****
00029   * PECBUSER: CURRENT USER ID *
00030   *****/
00031   FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=PECBUSER);
00032   FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
00033   EPOPULATE;
00034   ECRITERIA;
00035   CPASS CMD0020;
00036   ELABEL;

```

The Endeavor/INFO Interface Syntax Errors report for this syntax shows the following errors.

```

COPYRIGHT (C) COMPUTER ASSOCIATES, INC., 1987-2000                24FEB00 10:44:14    PAGE 1
                          CA- E N D E V O R / I N F O   I N T E R F A C E   S Y N T A X   E R R O R S   R E L E A S E   N . N   S E R I A L   X N N N N X
LINE # ERROR/WARNING/INFORMATIONAL MESSAGES
00026 UNKNOWN TOKEN POPULATE, WILL RESUME AFTER STATEMENT END
00032 UNKNOWN TOKEN EPOPULATE, WILL RESUME AFTER STATEMENT END
00034 UNKNOWN TOKEN CPASS, WILL RESUME AFTER STATEMENT END

```

Chapter 4. Implementing the Info/Man Side of the Interface

This section provides an overview of the Info/Man API and some guidelines for setting up the Info/Man side of the Endeavor Info/Man interface.

4.1 Info/Man API Basics

This section contains an overview of the Info/Man API, and is intended to provide an orientation for people responsible for implementing the Endeavor Info/Man interface. For more information, consult Info/Man experts at your site, or *IBM Tivoli Information/Management Application Program Interface Guide* SC34-4592-00.

The Info/Man database consists of 16 record types. The interface uses the following four Info/Man record types to store Endeavor information.

- Problem records, to record information about stand-alone actions. Stand-alone actions are actions that are not associated with a package.
- Change records, to record information about stand-alone actions and packages.
- Change Activity records, to record information about package actions.
- The Add Change Activity record, to relate package actions to a parent package record.

Four actions can be performed against each record: data can be created (Create), data can be updated (Update), inquiries can be made against the record (Inquiry), and the record can be retrieved (Retrieve). The Endeavor Info/Man interface supports all these access modes.

Info/Man allows logical views of these records, in the form of Program Interface Data Table (PIDT) tables. PIDT tables define data fields that can be passed between an application (in this case Endeavor) and the API.

Info/Man comes with a predefined PIDT table for each record type and use. For example, the Problem record type has four PIDT tables: Problem record create, Problem record update, Problem record inquiry, and Problem record retrieve. Each PIDT table provides a view of the problem record.

The Info/Man API uses PIDT tables to access or update data from the Info/Man database.

You can run the interface with the standard PIDT tables, or you can create your own. If you create your own PIDT tables, you must do so for each record type/action that you intend to use, and each custom table must contain certain required fields. These fields are listed later in this chapter.

These PIDT definition statements are fed into the Info/Man BLGUT8 batch utility. This utility produces two machine-loadable data structures for each table definition: the PIDT (Program Interface Data Table) and the Program Interface Pattern Table (PIPT). The PIPT defines the pattern validation criteria for each field found in the PIDT.

4.2 Checklist for Setting Up the Info/Man Side

Use this checklist when setting up the Info/Man side of the Endeavor Info/Man interface.

Note: It may be necessary to obtain detail information from both the Endeavor and the Info/Man administrators.

1. Decide which Endeavor information you want to track in Info/Man.
2. Decide which Info/Man record type you want to use to record stand-alone action information: Problem records or Change records.
3. Make sure that all Info/Man fields required by the interface are available.
4. Decide which functions you want to perform against the record types you have selected:

This Function	Allows You To
Inquiry	Check for the existence of Info/Man records.
Retrieve	Test Info/Man fields for specific values.
Create	Create records in Info/Man.
Update	Update existing records.

5. After making these decisions, build PIDT tables for the record types and functions that you plan to use.

For example, if you decide to record Endeavor package information in Info/Man, you would create PIDT tables for:

- Inquiry against Change records, for packages.
- Inquiry against Activity records, for package actions.
- Retrieve against Change records, for packages.
- Retrieve against Activity records, for package actions.
- Create Change records, for packages.
- Create Activity records, for package actions.
- Update Change records, for packages.
- Update Activity records, for package actions.

4.3 Building Info/Man PIDT Tables

The quickest way to get started using the interface is to use the IBM-supplied PIDT tables. If you are using the standard IBM PIDT tables, you can skip this section. Make sure, however, that the Info/Man fields reserved by the interface are not being used at your site to store other information.

If you are not using the standard PIDT tables, first complete the checklist in the previous section. You must then build PIDT tables for the record types (Problem, Change, or Change Activity) that you want to use, and the actions (Inquiry, Create, Update, Retrieve) that you want the interface to perform.

See *IBM Tivoli Information/Management Application Program Interface Guide* SC34-4592-00 for information about building PIDT tables.

Certain fields must appear in these PIDT tables. These required fields are listed in this section, for each possible record type and use.

Note: A four-character (for example IACT) or five-character (for example CPKGA) identifier appears in the following lists. These identifiers are used in the Control block of the batch utility syntax to build the E/INFO table. These identifiers are explained in Chapter 3, "Setting Up the Endeavor/INFO Table - the Batch Utility."

4.3.1 Inquiry PIDT Required Fields

Inquiry against Change records, for packages (IPKG):

This PIDT field	Is reserved for this information
FIELD PANEL (BLG00000) INDEX(S0B06)	Change type record
FIELD PANEL (BLG6CORQ) INDEX(S0CD4)	Package ID
FIELD PANEL (BLG6PTYP) INDEX(S0C09)	Type: PACKAGE
FIELD PANEL (BLG6ACCN) INDEX(S0CCF)	RNID value

Inquiry against Change Activity records, for package actions (IPKGA):

This PIDT field	Is reserved for this information
FIELD PANEL (BLG0F000) INDEX(S0B07)	Activity type record
FIELD PANEL (BLG6RNOR) INDEX(S0CD0)	Parent RNID
FIELD PANEL (BLG6ACNM) INDEX(S0CBC)	CCID
FIELD PANEL (BLG6PTYP) INDEX(S0C09)	Type: PKG/ACT
FIELD PANEL (BLG6ACCN) INDEX(S0CCF)	RNID value

Inquiry against Change records, for actions (IACT):

This PIDT field	Is reserved for this information
FIELD PANEL (BLG00000) INDEX(S0B06)	Change type record
FIELD PANEL(BLG6PTYP) INDEX(S0C09)	Type: ACTION
FIELD PANEL(BLG6ACCN) INDEX(S0CCF)	RNID value

Inquiry against Problem records, for actions (IACT):

This PIDT field	Is reserved for this information
FIELD PANEL(BLG00000) INDEX(S0032)	Problem type record
FIELD PANEL(BLG6PTYP) INDEX(S0C09)	Type: ACTION
FIELD PANEL(BLG6ACCN) INDEX(S0CCF)	RNID value

4.3.2 Retrieve PIDT Required Fields

Retrieve against Change records, for packages (RPKG):

This PIDT field	Is reserved for this information
FIELD PANEL(BLG00000) INDEX(S0B06)	Change type record
FIELD PANEL(BLG6CORQ) INDEX(S0CD4)	Package ID
FIELD PANEL(BLG6PTYP) INDEX(S0C09)	Type: PACKAGE

Retrieve against Change Activity records, for package actions (RPKGA):

This PIDT field	Is reserved for this information
FIELD PANEL(BLGLAL31) INDEX(S0B07)	Activity type record
FIELD PANEL(BLG6RNOR) INDEX(S0CD0)	Parent RNID
FIELD PANEL (BLG6PTYP) INDEX(S0C09)	Type: PKG/ACT
FIELD PANEL(BLG6ACNM) INDEX(S0CBC)	CCID

Retrieve against Change records, for actions (RACT):

This PIDT field	Is reserved for this information
FIELD PANEL(BLG00000) INDEX(S0B06)	Change type record
FIELD PANEL(BLG6PTYP) INDEX(S0C09)	Type: ACTION

Retrieve against Problem records, for actions (RACT):

This PIDT field	Is reserved for this information
FIELD PANEL(BLG00000) INDEX(S0032)	Problem type record
FIELD PANEL(BLG6PTYP) INDEX(S0C09)	Type: ACTION

4.3.3 Create PIDT Required Fields

Create Change records for packages (CPKG):

This PIDT field	Is reserved for this information
FIELD PANEL(BLG6ACNM) INDEX(S0CBC)	Package ID
FIELD PANEL(BLG6PTYP) INDEX(S0C09)	Type: PACKAGE
FIELD PANEL(BLG6URN0) INDEX(S0CCF)	RNID value

Create Change Activity records for package actions (CPKGA):

This PIDT field		Is reserved for this information
FIELD PANEL(BLGLAL31)	INDEX(S0B07)	Activity type record
FIELD PANEL(BLG6RNOR)	INDEX(S0CD0)	Parent RNID
FIELD PANEL(BLG6CORQ)	INDEX(S0CD4)	CCID
FIELD PANEL (BLG6PTYP)	INDEX(S0C09)	Type: PKG/ACT
FIELD PANEL(BLG6URN0)	INDEX(S0CCF)	RNID value

Create Change records for actions (CACT):

This PIDT field		Is reserved for this information
FIELD PANEL(BLG00000)	INDEX(S0B06)	Change type record
FIELD PANEL(BLG6PTYP)	INDEX(S0C09)	Type: package or action
FIELD PANEL(BLG6URN0)	INDEX(S0CCF)	RNID value

Create Problem records for actions (CACT):

This PIDT field		Is reserved for this information
FIELD PANEL(BLG00000)	INDEX(S0032)	Change type record
FIELD PANEL(BLG6PTYP)	INDEX(S0C09)	Type: package or action
FIELD PANEL(BLG6URN0)	INDEX(S0CCF)	RNID value

4.3.4 Update PIDT Required Fields

Update Change records for packages (UPKG):

This PIDT field		Is reserved for this information
FIELD PANEL(BLG00000)	INDEX(S0B06)	Change type record

Update Change Activity records for package actions (UPKGA):

This PIDT field	Is reserved for this information
FIELD PANEL(BLGLAL31) INDEX(S0B07)	Activity type record
FIELD PANEL (BLG6PTYP) INDEX(S0C09)	Type: PKG/ACT

Update Change records for actions (UACT):

This PIDT field	Is reserved for this information
FIELD PANEL(BLG00000) INDEX(S0B06)	Change type record

Update Problem records for actions (UACT):

This PIDT field	Is reserved for this information
FIELD PANEL(BLG00000) INDEX(S0032)	Problem type record

Chapter 5. Stand-Alone Action Table Syntax Example

This chapter contains a syntax template for building an E/INFO table for stand-alone actions.

5.1 Using the Syntax Template

There are two ways to track stand-alone actions: using Info/Man Change records or using Info/Man Problem records. Although the template described here is designed to track stand-alone actions using Info/Man Change records, you can easily modify the template to include syntax for using Problem records.

The syntax template described in this chapter is contained in member EISCRIP1 in iprfx.igual.SOURCE. The table built using this syntax updates/references Info/Man Change records. A syntax template for utilizing Problem records is provided in member EISCRIP2 in iprfx.igual.SOURCE.

If you have not modified standard Info/Man, you can use these syntax templates as-is. Consider, however, reviewing the syntax, editing it as necessary, before submitting it to the Batch Utility.

This sample table syntax uses fields that exist in both Change and Problem record types. If you customize this sample syntax, do not assume that all Change record fields are also defined within Problem records.

The following pages contain a walk-through of EISCRIP1. Even if you want to use Info/Man Problem records (EISCRIP2), studying the walk-through will be beneficial.

5.1.1 IIF Syntax Templates for Stand-Alone Actions

Info/Man Interactive Facility (IIF) users can find syntax templates for change and problem records in iprfx.igual.SOURCE. Member EISCRXX1 contains the template for building a table to use IIF Change records, and member EISCRXX2 contains a template for building a table to use IIF Problem records.

5.2 Sample Syntax

To familiarize you with the syntax, this section provides descriptions for the first few syntax blocks.

```
TABLE NAME (ENDEV02);
```

The TABLE NAME clause starts the table, and names it. In this case the name of the table is ENDEV02. See 3.5, “TABLE Statement” on page 3-8 for more information.

```
CTRL;
  IACT (TS0B06I);          /* INQUIRY CHANGE FOR ACTIONS */
  RACT (TS0B06R);          /* RETRIEVE CHANGE FOR ACTIONS */
  CACT (TS0B06C);          /* CREATE CHANGE FOR ACTIONS */
  UACT (TS0B06U);          /* UPDATE CHANGE FOR ACTIONS */
ECTRL;
```

This Control block indicates that Endevor may request the Info/Man API to:

- Inquire against PIDT table TS0B06I.
- Retrieve PIDT table TS0B06R.
- Create a Change record in PIDT table TS0B06C.
- Update and existing record in PIDT table TS0B06U.

See 3.6, “Control Block” on page 3-9 for more information.

```
USE(ADD,BEFORE,STANDALONE);
```

This USE statement indicates that the processing in this Use block applies to the before-exit point for stand-alone ADD actions. See 3.7, “Use Blocks” on page 3-13 for more information.

```
NOTFOUND (CREATE,ADD0010);
FOUND    (ADD0020);
```

These NOTFOUND/FOUND statements indicate that:

- If there is no Change record in Info/Man for the CCID associated with an ADD action, the interface is to create a record, using the processing specified in Label block ADD0010.
- If there is a record in Info/Man for the CCID associated with an ADD action, the interface is to execute the processing specified in Label block ADD0020.

See 3.7.3, “FOUND Statement” on page 3-17 for more information.

```
LABEL ADD0010;          /* CHANGE RECORD NOT FOUND: CREATE ONE */
```

This LABEL statement starts the Label block referred to by the NOTFOUND statement earlier in the syntax. See 3.8, “Label Block” on page 3-19 for more information.

```
CRITERIA;
ECRITERIA;
```

Even if there are no criteria for populating an Info/Man record, there must be a Criteria block within a Label block, if that block also contains a POPULATE statement. When there are no criteria, code the Criteria block this way. See 3.9, “Criteria Block” on page 3-26 for more information.

```
POPULATE;
```

This statement starts a Populate block. The FIELD and INDEX statements in a Populate block specify the fields to be populated in Info/Man. The VALUE statements specify the values to use to populate those fields. See 3.10, “Populate Block” on page 3-29 for more information.

```
FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=USER);
```

This statement tells the Info/Man API to populate the BLG6REQN field (index S0B59) with the Endeavor user ID. See 3.11, “FIELD Statements” on page 3-31 for more information.

```
FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
```

This statement tells the Info/Man API to populate the BLG6STAT field (index S0BEE) with the literal OPEN.

```
FIELD PANEL(BLG6DSAB) INDEX(S0E0F)  
VALUE('API CREATE CHG RECORD');
```

This statement tells the Info/Man API to populate the BLG6DSAB field (index S0E0F) with the text API CREATE CHG RECORD. See 3.11, “FIELD Statements” on page 3-31 for more information.

```
EPOPULATE;
```

This statement ends the Populate block. See 3.10, “Populate Block” on page 3-29 for more information.

```
RC(0);
```

This statement tells the API to send a return code of zero back to Endeavor after successfully executing the processing in this Label block. See 3.8.6, “RC Statements” on page 3-24 for more information.

```
ELABEL;
```

This statement ends Label block ADD0010. See 3.8, “Label Block” on page 3-19 for more information.

```
LABEL ADD0020; /* CHANGE RECORD FOUND */
```

This statement starts the label block referenced in the FOUND statement earlier in the syntax. See 3.8, “Label Block” on page 3-19 for more information.

```
RC (0);
```

This statement tells the API to send a return code of zero back to Endeavor after successfully executing the processing in this Label block. See 3.8.6, “RC Statements” on page 3-24 for more information.

```
CRITERIA;
```

This is a required statement to start a Criteria block for this Label block. See 3.9, “Criteria Block” on page 3-26 for more information.

```
FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
```

This statement tells the interface to query field BLG6STAT (index S0BEE) in the Info/Man Change record associated with the CCID for this ADD action. If the value in the field is the literal OPEN, the criteria block is considered true. In this case, this means that it considers the record as found. See 3.11, “FIELD Statements” on page 3-31 for more information.

```
ECRITERIA;
```

This is a required statement to end a Criteria block. See 3.9, “Criteria Block” on page 3-26 for more information.

```
POPULATE;  
EPOPULATE;
```

Even if you do not wish to populate an Info/Man record when a Criteria block is true, you must include a Populate block with that Criteria block. This is the way to code a null Populate block. See 3.10, “Populate Block” on page 3-29 for more information.

```
CFAIL ADD0022;
```

This CFAIL statement tells the interface to execute the processing specified in Label block ADD0022 if the Criteria block is false. This would be the case in this example if the Info/Man field BLG6STAT (index S0BEE) did not contain the value OPEN. See 3.8.3, “CPASS/CFAIL Statements” on page 3-20 for more information.

```
ELABEL;
```

This statement ends Label block ADD0020. For more information see 3.8.2, “ELABEL Statement” on page 3-20

```
LABEL ADD0022;      /* STATUS NOT OPEN - CHECK FOR INITIAL */
```

This statement starts Label block ADD0022. See 3.8.1, “LABEL Statement” on page 3-19 for more information.

```
CRITERIA;  
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('INITIAL');  
  IFNO;  
    RC (8);  
    MSG ('INVALID STATUS - MUST BE INITIAL OR OPEN');  
  EIF;  
ECRITERIA;
```

This Criteria block tells the interface to query field BLG6STAT (index S0BEE) in the Info/Man Change record associated with the CCID for this ADD action.

- If the value in the field is INITIAL, the criteria block is considered true, and processing continues with the POPULATE statement.
- If the value in the field is not INITIAL, the interface executes the IFNO statement, passing a return code of 8 back to Endeavor and issuing the message INVALID STATUS - MUST BE INITIAL OR OPEN.

```
POPULATE;  
EPOPULATE;
```

Even if you do not wish to populate an Info/Man record when a Criteria block is true, you must include a Populate block with that Criteria block. This is the way to code a null Populate block. See 3.10, “Populate Block” on page 3-29 for more information.

```
ELABEL;
```

This statement ends the ADD0022 Label block. More information: 3.8.2, “ELABEL Statement” on page 3-20

```
EUSE;
```

This statement ends the Use block for the before-exit point for stand-alone ADD actions. See 3.7.5, “EUSE Statement” on page 3-18 for more information.

```
USE (ADD, AFTER, STANDALONE);
```

This USE statement starts another Use block. The processing in this Use block is to be invoked for the after-exit point for stand-alone ADD actions. See 3.7.1, “USE Statement” on page 3-14 for more information.

```
FOUND (ADD0010);  
NOTFOUND (RETURN);
```

These NOTFOUND/FOUND statements indicate that:

- If there is a record in Info/Man for the CCID associated with an ADD action, the interface is to execute the processing specified in Label block ADD0010.
- If there is no Change record in Info/Man for the CCID associated with an ADD action, the interface is to return to Endeavor.

See 3.7.3, “FOUND Statement” on page 3-17 for more information.

```
LABEL ADD0010;
```

This statement starts Label block ADD0010, referenced in the FOUND statement earlier in the Use block. See 3.8, “Label Block” on page 3-19 for more information.

```
CRITERIA;  
ECRITERIA;
```

Grouping the CRITERIA and ECRITERIA statements this way indicates that there are no criteria to be satisfied after a record is found. This tells the interface to execute the processing specified in the Populate block. See 3.9, “Criteria Block” on page 3-26 for more information.

```
POPULATE;  
  FIELD PANEL(BLG0C010) INDEX(S0E01)  
    TEXT ('ACTION PROCESSING COMPLETED RC = =ACTRC')  
    TEXT ('ELEMENT: =EV2ENV / =EV2SYS / =EV2SUB ') -  
      (' / =EV2TYP / =EV2STAGE / =EV2ELMFN')  
    TEXT ('ACTION: =ACTION USER: =USER DATE: =DATE ') -  
      ('TIME: =TIME');  
EPOPULATE;
```

This POPULATE statement tells the Info/Man API to populate field BLG0C010 (index S0E01) with a three-line message.

There is one TEXT statement for each line. Lines are continued by placing a hyphen as the last character in the first line. Also note that there is only one semicolon for all three TEXT statements, located at the end of the last one.

The message is built using information from Endeavor control blocks

- Line 1 uses the ACTRC field from the \$ENVDS block.
- Line 2 uses the EV2ENV, EV2SYS, EV2SUB, EV2TYP, EV2STAGE, and EV2ELMFN fields from the \$ENVDS block.
- Line 3 uses the ACTION, USER, DATE, and TIME fields from the \$ENVDS block.

```
ELABEL;
```

This statement ends Label block ADD0010. See 3.8.2, “ELABEL Statement” on page 3-20 for more information.

```
EUSE;
```

This statement ends the Use block for the after-exit point for stand-alone ADD actions. See 3.7.5, “EUSE Statement” on page 3-18 for more information.

This concludes the annotated portion of the syntax example. Sample syntax for the remaining stand-alone actions is shown on the following table.

```
USE(ADD, BEFORE, STANDALONE);
```

```
NOTFOUND (CREATE,ADD0010);
FOUND    (ADD0020);
```

```
LABEL ADD0010;          /* CHANGE RECORD NOT FOUND: CREATE ONE */
  CRITERIA;
  ECRITERIA;
  POPULATE;
    FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=USER);
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
    FIELD PANEL(BLG6DSAB) INDEX(S0E0F)
      VALUE('API CREATE CHG RECORD');
  EPOPULATE;
  RC(0);
ELABEL;
```

```
LABEL ADD0020;      /* CHANGE RECORD FOUND */
RC (0);
CRITERIA;
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
ECRITERIA;
  POPULATE;
  EPOPULATE;
CFAIL ADD0022;
ELABEL;

LABEL ADD0022;      /* STATUS NOT OPEN - CHECK FOR INITIAL */
CRITERIA;
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('INITIAL');
  IFNO;
  RC (8);
  MSG ('INVALID STATUS - MUST BE INITIAL OR OPEN');
  EIF;
ECRITERIA;
  POPULATE;
  EPOPULATE;
ELABEL;
EUSE;

USE (ADD, AFTER, STANDALONE);

FOUND (ADD0010);
NOTFOUND (RETURN);

LABEL ADD0010;
CRITERIA;
ECRITERIA;
  POPULATE;
  FIELD PANEL(BLG0C010) INDEX(S0E01)
    TEXT ('ACTION PROCESSING COMPLETED RC = =ACTRC')
    TEXT ('ELEMENT: =EV2ENV / =EV2SYS / =EV2SUB ') -
    (' / =EV2TYP / =EV2STAGE / =EV2ELMFN')
    TEXT ('ACTION: =ACTION USER: =USER DATE: =DATE ') -
    ('TIME: =TIME');
  EPOPULATE;
ELABEL;
EUSE;

USE (UPDATE, BEFORE, STANDALONE);

NOTFOUND (CREATE,UPT0010);
FOUND (UPT0020);

LABEL UPT0010;      /* CHANGE RECORD NOT FOUND: CREATE ONE */
CRITERIA;
ECRITERIA;
  POPULATE;
  FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=USER);
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
  FIELD PANEL(BLG6DSAB) INDEX(S0E0F)
    VALUE('API CREATE CHG RECORD');
  EPOPULATE;
RC(0);
ELABEL;
```

```

LABEL UPT0020;      /* CHANGE RECORD FOUND */
  RC (0);
  CRITERIA;
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
  ECRITERIA;
  POPULATE;
  EPOPULATE;
  CFAIL UPT0022;
  ELABEL;

LABEL UPT0022;      /* STATUS NOT OPEN - CHECK FOR INITIAL */
  CRITERIA;
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('INITIAL');
  IFNO;
  RC (8);
  MSG ('INVALID STATUS - MUST BE INITIAL OR OPEN');
  EIF;
  ECRITERIA;
  POPULATE;
  EPOPULATE;
  ELABEL;

EUSE;

USE (UPDATE, AFTER, STANDALONE);

FOUND (UPT0010);
NOTFOUND (RETURN);

LABEL UPT0010;
  CRITERIA;
  ECRITERIA;
  POPULATE;
  FIELD PANEL(BLG0C010) INDEX(S0E01)
    TEXT ('ACTION PROCESSING COMPLETED RC = =ACTRC')
    TEXT ('ELEMENT: =EV2ENV / =EV2SYS / =EV2SUB ') -
      (' / =EV2TYP / =EV2STAGE / =EV2ELMFN')
    TEXT ('ACTION: =ACTION USER: =USER DATE: =DATE ') -
      ('TIME: =TIME');
  EPOPULATE;
  ELABEL;
  EUSE;

USE (RESTORE,BEFORE, STANDALONE);

NOTFOUND (CREATE,RST0010);
FOUND (RST0020);

LABEL RST0010;      /* CHANGE RECORD NOT FOUND: CREATE ONE */
  CRITERIA;
  ECRITERIA;
  POPULATE;
  FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=USER);
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
  FIELD PANEL(BLG6DSAB) INDEX(S0E0F)
    VALUE('API CREATE CHG RECORD');
  EPOPULATE;
  RC (0);
  ELABEL;

```

```
LABEL RST0020;      /* CHANGE RECORD FOUND */
  RC (0);
  CRITERIA;
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
  ECRITERIA;
  POPULATE;
  EPOPULATE;
  CFAIL RST0022;
  ELABEL;

LABEL RST0022;      /* STATUS NOT OPEN - CHECK FOR INITIAL */
  CRITERIA;
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('INITIAL');
  IFNO;
  RC (8);
  MSG ('INVALID STATUS - MUST BE INITIAL OR OPEN');
  EIF;
  ECRITERIA;
  POPULATE;
  EPOPULATE;
  ELABEL;

EUSE;

USE (RESTORE,AFTER, STANDALONE);

FOUND (RST0010);
NOTFOUND (RETURN);

LABEL RST0010;
  CRITERIA;
  ECRITERIA;
  POPULATE;
  FIELD PANEL(BLG0C010) INDEX(S0E01)
  TEXT ('ACTION PROCESSING COMPLETED RC = =ACTRC')
  TEXT ('SOURCE ELEMENT: =EVIENV / =EV1SYS / =EV1SUB ') -
  (' / =EV1TYP / =EV1STG# / =EV1ELMFN')
  TEXT ('TARGET ELEMENT: =EV2ENV / =EV2SYS / =EV2SUB ') -
  (' / =EV2TYP / =EV2STAGE / =EV2ELMFN')
  TEXT ('ACTION: =ACTION USER: =USER DATE: =DATE ') -
  ('TIME: =TIME');
  EPOPULATE;
  ELABEL;
  EUSE;

USE (GENERATE, BEFORE, STANDALONE);

NOTFOUND (CREATE,GEN0010);
FOUND (GEN0020);

LABEL GEN0010;      /* CHANGE RECORD NOT FOUND: CREATE ONE */
  CRITERIA;
  ECRITERIA;
  POPULATE;
  FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=USER);
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
  FIELD PANEL(BLG6DSAB) INDEX(S0E0F)
  VALUE('API CREATE CHG RECORD');
  EPOPULATE;
  RC(0);
  ELABEL;
```

```

LABEL GEN0020;          /* CHANGE RECORD FOUND */
RC (0);
CRITERIA;
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
ECRITERIA;
  POPULATE;
  EPOPULATE;
CFAIL GEN0022;
ELABEL;

LABEL GEN0022;          /* STATUS NOT OPEN - CHECK FOR INITIAL */
CRITERIA;
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('INITIAL');
  IFNO;
  RC (8);
  MSG ('INVALID STATUS - MUST BE INITIAL OR OPEN');
  EIF;
ECRITERIA;
  POPULATE;
  EPOPULATE;
ELABEL;

EUSE;

USE (GENERATE ,AFTER, STANDALONE);

FOUND (GEN0010);
NOTFOUND (RETURN);

LABEL GEN0010;
CRITERIA;
ECRITERIA;
  POPULATE;
  FIELD PANEL(BLG0C010) INDEX(S0E01)
  TEXT ('ACTION PROCESSING COMPLETED RC = =ACTRC')
  TEXT ('ELEMENT: =EV1ENV / =EV1SYS / =EV1SUB ') -
  (' / =EV1TYP / =EV1STAGE / =EV1ELMFN')
  TEXT ('ACTION: =ACTION USER: =USER DATE: =DATE ') -
  ('TIME: =TIME');
  EPOPULATE;
ELABEL;
EUSE;

USE (RETRIEVE, BEFORE, STANDALONE);

NOTFOUND (CREATE,RET0010);
FOUND (RET0020);

LABEL RET0010;          /* CHANGE RECORD NOT FOUND: CREATE ONE */
CRITERIA;
ECRITERIA;
  POPULATE;
  FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=USER);
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
  FIELD PANEL(BLG6DSAB) INDEX(S0E0F)
  VALUE('API CREATE CHG RECORD');
  EPOPULATE;
RC(0);
ELABEL;

```

```
LABEL RET0020;      /* CHANGE RECORD FOUND */
  RC (0);
  CRITERIA;
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
  ECRITERIA;
  POPULATE;
  EPOPULATE;
  CFAIL RET0022;
  ELABEL;

LABEL RET0022;      /* STATUS NOT OPEN - CHECK FOR INITIAL */
  CRITERIA;
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('INITIAL');
  IFNO;
  RC (8);
  MSG ('INVALID STATUS - MUST BE INITIAL OR OPEN');
  EIF;
  ECRITERIA;
  POPULATE;
  EPOPULATE;
  ELABEL;

EUSE;

USE (RETRIEVE, AFTER, STANDALONE);

FOUND (RET0010);
NOTFOUND (RETURN);

LABEL RET0010;
  CRITERIA;
  ECRITERIA;
  POPULATE;
  FIELD PANEL(BLG0C010) INDEX(S0E01)
    TEXT ('ACTION PROCESSING COMPLETED RC = =ACTRC')
    TEXT ('ELEMENT: =EVIENV / =EVISYS / =EVISUB ') -
    ('/ =EV1TYP / =EV1STAGE / =EV1ELMFN')
    TEXT ('ACTION: =ACTION USER: =USER DATE: =DATE ') -
    ('TIME: =TIME');
  EPOPULATE;
  ELABEL;
  EUSE;

USE (ARCHIVE, BEFORE, STANDALONE);

NOTFOUND (CREATE,ARC0010);
FOUND (ARC0020);

LABEL ARC0010;      /* CHANGE RECORD NOT FOUND: CREATE ONE */
  CRITERIA;
  ECRITERIA;
  POPULATE;
  FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=USER);
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
  FIELD PANEL(BLG6DSAB) INDEX(S0E0F)
    VALUE('API CREATE CHG RECORD');
  EPOPULATE;
  RC(0);
  ELABEL;
```

```
LABEL ARC0020;          /* CHANGE RECORD FOUND */
RC (0);
CRITERIA;
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
ECRITERIA;
  POPULATE;
  EPOPULATE;
CFAIL ARC0022;
ELABEL;

LABEL ARC0022;          /* STATUS NOT OPEN - CHECK FOR INITIAL */
CRITERIA;
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('INITIAL');
  IFNO;
  RC (8);
  MSG ('INVALID STATUS - MUST BE INITIAL OR OPEN');
  EIF;
ECRITERIA;
  POPULATE;
  EPOPULATE;
ELABEL;

EUSE;

USE (ARCHIVE, AFTER, STANDALONE);

FOUND (ARC0010);
NOTFOUND (RETURN);

LABEL ARC0010;
CRITERIA;
ECRITERIA;
  POPULATE;
  FIELD PANEL(BLG0C010) INDEX(S0E01)
  TEXT ('ACTION PROCESSING COMPLETED RC = =ACTRC')
  TEXT ('ELEMENT: =EV1ENV / =EV1SYS / =EV1SUB ') -
  (' / =EV1TYP / =EV1STAGE / =EV1ELMFN')
  TEXT ('TARGET: =FI2DSN =FI2MBR')
  TEXT ('ACTION: =ACTION USER: =USER DATE: =DATE ') -
  ('TIME: =TIME');
  EPOPULATE;
ELABEL;
EUSE;

USE (DELETE, BEFORE, STANDALONE);

NOTFOUND (CREATE,DEL0010);
FOUND (DEL0020);
```

```
LABEL DEL0010;      /* CHANGE RECORD NOT FOUND: CREATE ONE */
  CRITERIA;
  ECRITERIA;
  POPULATE;
    FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=USER);
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
    FIELD PANEL(BLG6DSAB) INDEX(S0E0F)
      VALUE('API CREATE CHG RECORD');
  EPOPULATE;
  RC(0);
ELABEL;

LABEL DEL0020;      /* CHANGE RECORD FOUND */
  RC (0);
  CRITERIA;
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
  ECRITERIA;
  POPULATE;
  EPOPULATE;
  CFAIL DEL0022;
ELABEL;

LABEL DEL0022;      /* STATUS NOT OPEN - CHECK FOR INITIAL */
  CRITERIA;
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('INITIAL');
    IFNO;
    RC (8);
    MSG ('INVALID STATUS - MUST BE INITIAL OR OPEN');
  EIF;
  ECRITERIA;
  POPULATE;
  EPOPULATE;
ELABEL;

EUSE;

USE (DELETE, AFTER, STANDALONE);

FOUND (DEL0010);
NOTFOUND (RETURN);

LABEL DEL0010;
  CRITERIA;
  ECRITERIA;
  POPULATE;
    FIELD PANEL(BLG0C010) INDEX(S0E01)
      TEXT ('ACTION PROCESSING COMPLETED RC = =ACTRC')
      TEXT ('ELEMENT: =EV1ENV / =EV1SYS / =EV1SUB ') -
      (' / =EV1TYP / =EV1STAGE / =EV1ELMFN')
      TEXT ('ACTION: =ACTION USER: =USER DATE: =DATE ') -
      ('TIME: =TIME');
  EPOPULATE;
ELABEL;
EUSE;

USE (MOVE, BEFORE, STANDALONE);

NOTFOUND (CREATE,MOV0010);
FOUND (MOV0020);
```

```

LABEL MOV0010;          /* CHANGE RECORD NOT FOUND: CREATE ONE */
CRITERIA;
ECRITERIA;
POPULATE;
    FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=USER);
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
    FIELD PANEL(BLG6DSAB) INDEX(S0E0F)
        VALUE('API CREATE CHG RECORD');
EPOPULATE;
RC(0);
ELABEL;

LABEL MOV0020;          /* CHANGE RECORD FOUND */
RC(0);
CRITERIA;
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
ECRITERIA;
POPULATE;
EPOPULATE;
CFAIL MOV0022;
ELABEL;

LABEL MOV0022;          /* STATUS NOT OPEN - CHECK FOR INITIAL */
CRITERIA;
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('INITIAL');
    IFNO;
    RC(8);
    MSG('INVALID STATUS - MUST BE INITIAL OR OPEN');
    EIF;
ECRITERIA;
POPULATE;
EPOPULATE;
ELABEL;

EUSE;

USE(MOVE, AFTER, STANDALONE);

FOUND(MOV0010);
NOTFOUND(RETURN);

LABEL MOV0010;
CRITERIA;
ECRITERIA;
POPULATE;
    FIELD PANEL(BLG0C010) INDEX(S0E01)
        TEXT('ACTION PROCESSING COMPLETED RC = =ACTRC')
        TEXT('SOURCE ELEMENT: =EV1ENV / =EV1SYS / =EV1SUB ') -
            (' / =EV1TYP / =EV1STAGE / =EV1ELMFN')
        TEXT('TARGET ELEMENT: =EV2ENV / =EV2SYS / =EV2SUB ') -
            (' / =EV2TYP / =EV2STAGE / =EV2ELMFN')
        TEXT('ACTION: =ACTION USER: =USER DATE: =DATE ') -
            ('TIME: =TIME');
EPOPULATE;
ELABEL;
EUSE;

USE(TRANSFER, BEFORE, STANDALONE);

NOTFOUND(CREATE, TRA0010);
FOUND(TRA0020);

```

```
LABEL TRA0010;      /* CHANGE RECORD NOT FOUND: CREATE ONE */
  CRITERIA;
  ECRITERIA;
  POPULATE;
    FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=USER);
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
    FIELD PANEL(BLG6DSAB) INDEX(S0E0F)
      VALUE('API CREATE CHG RECORD');
  EPOPULATE;
  RC(0);
ELABEL;

LABEL TRA0020;      /* CHANGE RECORD FOUND */
  RC (0);
  CRITERIA;
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
  ECRITERIA;
  POPULATE;
  EPOPULATE;
  CFAIL TRA0022;
ELABEL;

LABEL TRA0022;      /* STATUS NOT OPEN - CHECK FOR INITIAL */
  CRITERIA;
    FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('INITIAL');
    IFNO;
    RC (8);
    MSG ('INVALID STATUS - MUST BE INITIAL OR OPEN');
  EIF;
  ECRITERIA;
  POPULATE;
  EPOPULATE;
ELABEL;

EUSE;

USE (TRANSFER, AFTER, STANDALONE);

FOUND (TRA0010);
NOTFOUND (RETURN);

LABEL TRA0010;
  CRITERIA;
  ECRITERIA;
  POPULATE;
    FIELD PANEL(BLG0C010) INDEX(S0E01)
      TEXT ('ACTION PROCESSING COMPLETED RC = =ACTRC')
      TEXT ('SOURCE ELEMENT: =EV1ENV / =EV1SYS / =EV1SUB ') -
        (' / =EV1TYP / =EV1STAGE / =EV1ELMFN')
      TEXT ('TARGET ELEMENT: =EV2ENV / =EV2SYS / =EV2SUB ') -
        (' / =EV2TYP / =EV2STAGE / =EV2ELMFN')
      TEXT ('ACTION: =ACTION USER: =USER DATE: =DATE ') -
        ('TIME: =TIME');
  EPOPULATE;
ELABEL;
EUSE;

ETABLE;
```

Chapter 6. Package and Package Action Table Syntax Example

6.1 Syntax Example

This chapter contains sample table syntax for packages and package actions. The sample is for your use when building the E/INFO table for your site. The sample described is located in member EISCRIP3 in iprfx.igual.SOURCE. Several lines are annotated at the beginning of the sample, for your reference.

Info/Man Interactive Facility (IIF) users can find a syntax template for packages and package actions in member EISCRXX3 in iprfx.igual.SOURCE.

```
TABLE NAME (ENDEV03);
```

This statement starts the table. The name of the table is ENDEV03. There must be an ETABLE statement at the end of the table. See 3.8.1, “LABEL Statement” on page 3-19 for more information.

```
CTRL;
  IPKG (TS0B06I);          /* INQUIRY CHANGE TABLE FOR PACKAGES      */
  IPKGA(TS0B07I);         /* INQUIRY ACTIVITY TABLE FOR PKG ACTIONS */
  RPKG (TS0B06R);         /* RETRIEVE CHANGE TABLE FOR PACKAGES     */
  RPKGA(TS0B07R);         /* RETRIEVE ACTIVITY TABLE FOR PKG ACTIONS */
  CPKG (TS0B06C);         /* CREATE CHANGE TABLE FOR PACKAGES       */
  CPKGA(TS0B07C);         /* CREATE ACTIVITY TABLE FOR PKG ACTIONS  */
  UPKG (TS0B06U);         /* UPDATE CHANGE TABLE FOR PACKAGES       */
  UPKGA(TS0B07U);         /* UPDATE ACTIVITY TABLE FOR PKG ACTIONS  */
  AACT (TS0B06A);         /* ADD CHANGE ACTIVITY TABLE FOR PKG ACTION*/
ECTRL;
```

This Control block indicates that Endeavor may request the Info/Man API to:

- Inquire about package (IPKG) and package action (IPKGA) records.
- Retrieve package (RPKG) and package action (RPKGA) records.
- Create a package (CPKG) and package action (CPKGA) records.
- Update package (UPKG) and package action (UPKGA) records.
- Add change activity (AACT) records for package actions.

See 3.6, “Control Block” on page 3-9 for more information.

```
/*****
 * EXIT POINTS: CREATE/AFTER  MODIFY/ AFTER  *
 *****/
USE ((CREATE, AFTER),
     (MODIFY, AFTER));
```

This USE statement indicates that the processing in this Use block applies to the after-exit point for the CREATE PACKAGE and MODIFY PACKAGE actions. More information: 3.7, “Use Blocks” on page 3-13

```
NOTFOUND (CREATE,CMD0010);
FOUND    (CMD0020);
```

These NOTFOUND/FOUND statements indicate that

- If there is no Change record in Info/Man for a package, the interface is to create a record, using the processing specified in Label block CMD0010.
- If there is a record in Info/Man for the package, the interface is to execute the processing specified in Label block CMD0020.

See 3.7.3, “FOUND Statement” on page 3-17 for more information.

```
LABEL CMD0010;          /* CHANGE RECORD NOT FOUND: CREATE ONE */
```

This LABEL statement starts the Label block referred to by the NOTFOUND statement earlier in the syntax. See 3.8, “Label Block” on page 3-19 for more information.

```
CRITERIA;
ECRITERIA;
```

Even if there are no criteria for populating an Info/Man record, there must be a Criteria block within a Label block, if that block also contains a POPULATE statement. When there are no criteria, code the Criteria block this way. See 3.9, “Criteria Block” on page 3-26 for more information.

```
POPULATE;
  /******
  * PECBUSER: CURRENT USER ID          *
  *****/
  FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=PECBUSER);
  FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
  FIELD PANEL(BLG0C010) INDEX(S0E01)
    VALUE('E/INFO CREATED CHANGE RECORD');
EPOPULATE;
```

This POPULATE block tells the Info/Man API what information to put in the Change record that it creates. In this example the API populates three fields

- The BLG6REQN field (index S0B59) with the value in the PECBUSER field of the Endeavor package exit control block (\$PECBDS).
- The BLG6STAT field (index S0BEE) with the value OPEN.
- The BLG0C010 field (index S0E01) with the value E/INFO CREATED CHANGE RECORD.

See 3.9, “Criteria Block” on page 3-26 for more information.

```
CPASS CMD0020;
```

The CPASS statement tells the interface to execute the processing specified in Label block CMD0020 if the API populates the new record successfully. See 3.8.3, “CPASS/CFail Statements” on page 3-20 for more information.

```
ELABEL;
```

This statement ends Label block CMD0010. See 3.8.2, “ELABEL Statement” on page 3-20 for more information.

```
LABEL CMD0020;
```

This statement starts the label block referenced in the CPASS statement in the preceding Label block. See 3.8, “Label Block” on page 3-19 for more information.

```
RC (0);
```

This statement tells the API to send a return code of zero back to Endeavor after successfully executing the processing in this Label block. See 3.8.6, “RC Statements” on page 3-24 for more information.

```
CRITERIA;  
ECRITERIA;
```

Grouping the CRITERIA and ECRITERIA statements this way indicates that there are no criteria to be satisfied. This tells the interface to execute the processing specified in the Populate block. See 3.9, “Criteria Block” on page 3-26 for more information.

```
POPULATE;
```

```

/*****
* PREQCOMM: PACKAGE COMMENT *
* PREQEWS: EXECUTION WINDOW START DATE *
* PREQEWST: EXECUTION WINDOW START TIME *
* PREQEWED: EXECUTION WINDOW END DATE *
* PREQEWET: EXECUTION WINDOW END TIME *
* PECBUSER: CURRENT USER ID *
* PECBFNM: PACKAGE FUNCTION *
* PECBANM: BEFORE OR AFTER *
* PECBNDR: NDVR HIGH RETURN CODE *
* PHDRSTAT: PACKAGE STATUS *
* PHDRCRD: PACKAGE CREATION DATE *
* PHDRCRT: PACKAGE CREATION TIME *
*****/
```

This POPULATE statement is followed by a multi-line comment expanding the names of the Endeavor control block fields used to populate Info/Man fields in the following FIELD and TEXT statements.

See 3.10, “Populate Block” on page 3-29 for more information.

```

FIELD PANEL(BLG6DSAB) INDEX(S0E0F) VALUE(=PREQCOMM);
FIELD PANEL(BLG6SCHED) INDEX(S0C41) VALUE(=PREQEWS);
FIELD PANEL(BLG6SCHT) INDEX(S0C6E) VALUE(=PREQEWST);
FIELD PANEL(BLG6TARD) INDEX(S0C42) VALUE(=PREQEWED);
FIELD PANEL(BLG6TART) INDEX(S0C6F) VALUE(=PREQEWET);
FIELD PANEL(BLG600CN) INDEX(S0B5C) VALUE(=PECBUSER);
FIELD PANEL(BLG6PHAC) INDEX(S0C0B) VALUE(=PECBFNM);
```

These FIELD statements specify the Info/Man fields that are to contain Endeavor information, and the values that the API is to use to populate these fields. The comment earlier in the Populate block identifies the information referred to by the equate value.

See 3.11, “FIELD Statements” on page 3-31 for more information.

```
FIELD PANEL(BLG0C010) INDEX(S0E01)
  TEXT ('PACKAGE PROCESSING COMPLETED RC = =PECBNDRC')
  TEXT ('INFO ACCESSED FROM PACKAGE EXIT =PECBBANM / =PECBFNNM')
  TEXT ('PKG STATUS: =PHDRSTAT USER: =PECBUSER')
  TEXT ('CREATE DATE: =PHDRCRD TIME: =PHDRCRT');
```

These TEXT statements tell the interface to populate field BLGC010 (index S0E01) with a four-line message.

There is one TEXT statement for each line. Lines are continued by placing a hyphen as the last character in the first line. Also note that there is only one semicolon for all three TEXT statements, located at the end of the last statement.

The message is built using information from Endeavor control blocks.

- Line 1 uses the PECBNDRC field from the \$PECBDS block.
- Line 2 uses the PECBBANM and PECBFNNM fields from the \$PECBDS block.
- Line 3 uses the PHDRSTAT field from the PHDRDS block and the PECBUSER field from the \$PECBDS block.
- Line 4 uses the PHDRCRD and PHDRCRT fields from the PHDRDS block.

```
EPOPULATE;
```

This statement ends the Populate block.

See 3.10, “Populate Block” on page 3-29 for more information.

```
ELABEL;
```

This statement ends Label block CMD0020.

See 3.8.2, “ELABEL Statement” on page 3-20 for more information.

```
EUSE;
```

This statement ends the Use block for the after-exit point for the CREATE PACKAGE and MODIFY PACKAGE actions.

See 3.7.5, “EUSE Statement” on page 3-18 for more information.

```

USE (CAST, AFTER);

FOUND (CST0020);
NOTFOUND (CREATE, CST0010);

LABEL CST0010; /* CHANGE RECORD NOT FOUND: CREATE ONE. */
CRITERIA;
ECRITERIA;
POPULATE;

/******
 * PECBUSER: CURRENT USER ID *
*****/
FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=PECBUSER);
FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
EPOPULATE;
CPASS CST0020;
ELABEL;

LABEL CST0020;
RC (0);
CRITERIA;
ECRITERIA;
POPULATE;

/******
 * PREQCOMM: PACKAGE COMMENT *
 * PREQEWS: EXECUTION WINDOW START DATE *
 * PREQEWST: EXECUTION WINDOW START TIME *
 * PREQEWED: EXECUTION WINDOW END DATE *
 * PREQEWET: EXECUTION WINDOW END TIME *
 * PECBUSER: CURRENT USER ID *
 * PECBFNM: PACKAGE FUNCTION *
 * PECBBANM: BEFORE OR AFTER *
 * PECBNDRC: NDVR HIGH RETURN CODE *
 * PHDRSTAT: PACKAGE STATUS *
 * PHDRCRD: PACKAGE CREATION DATE *
 * PHDRCRT: PACKAGE CREATION TIME *
 * PHDRCD: PACKAGE CAST DATE *
 * PHDRCT: PACKAGE CAST TIME *
*****/
FIELD PANEL(BLG6DSAB) INDEX(S0E0F) VALUE(=PREQCOMM);
FIELD PANEL(BLG6SCHD) INDEX(S0C41) VALUE(=PREQEWS);
FIELD PANEL(BLG6SCHT) INDEX(S0C6E) VALUE(=PREQEWST);
FIELD PANEL(BLG6TARD) INDEX(S0C42) VALUE(=PREQEWED);
FIELD PANEL(BLG6TART) INDEX(S0C6F) VALUE(=PREQEWET);
FIELD PANEL(BLG600CN) INDEX(S0B5C) VALUE(=PECBUSER);
FIELD PANEL(BLG6PHAC) INDEX(S0C0B) VALUE(=PECBFNM);

```

```

FIELD PANEL(BLG0C010) INDEX(S0E01)
TEXT ('PACKAGE PROCESSING COMPLETED RC = =PECBNDRC')
TEXT ('INFO ACCESSED FROM PACKAGE EXIT =PECBBANM / =PECBFNNM')
TEXT ('PKG STATUS: =PHDRSTAT USER: =PECBUSER')
TEXT ('CREATE DATE: =PHDRCRD TIME: =PHDRCRT')
TEXT ('CAST DATE: =PHDRCD TIME: =PHDRCT');
EPOPULATE;
ELABEL;

EUSE;

/*****
* EXIT POINT: CREATE/ACTIVITY. INVOKED AFTER *
* THE CAST AFTER EXIT. THIS INFO EXIT WILL *
* CREATE AN ACTIVITY RECORD PER PACKAGE ACTION *
* BY CCID. *
*****/

USE (CREATE, ACTIVITY);

LABEL CRT0010; /* MAINLINE - ALL PKG ACTION POINTS. */
CRITERIA;
ECRITERIA;
POPULATE;

/******
* PECBUSER: CURRENT USER ID *
*****/
FIELD PANEL(BLG6REQN) INDEX(S0B59) VALUE(=PECBUSER);
FIELD PANEL(BLG6STAT) INDEX(S0BEE) VALUE('OPEN');
FIELD PANEL(BLG6DSAB) INDEX(S0E0F)
TEXT ('CCID: =PACTCCID IN PACKAGE: =PECBPKID');
EPOPULATE;
CPASS CRT0020;
ELABEL;

LABEL CRT0020; /* LOOKING FOR ACTION: ADD */
CRITERIA;
FIELD (=PACTACTN) VALUE('ADD ');
ECRITERIA;
POPULATE;
EPOPULATE;

CPASS GRP0010;
CFAIL CRT0025;
ELABEL;

```

6.1 Syntax Example

```
LABEL CRT0025;          /* LOOKING FOR ACTION: UPDATE          */
  CRITERIA;
    FIELD (=PACTACTN) VALUE('UPDATE ');
  ECRITERIA;
    POPULATE;
    EPOPULATE;

  CPASS GRP0010;
  CFAIL CRT0030;
ELABEL;

LABEL CRT0030;          /* LOOKING FOR ACTION: RESTORE          */
  CRITERIA;
    FIELD (=PACTACTN) VALUE('RESTORE ');
  ECRITERIA;
    POPULATE;
    EPOPULATE;

  CPASS GRP0010;
  CFAIL CRT0035;
ELABEL;

LABEL CRT0035;          /* LOOKING FOR ACTION: GENERATE          */
  CRITERIA;
    FIELD (=PACTACTN) VALUE('GENERATE');
  ECRITERIA;
    POPULATE;
    EPOPULATE;

  CPASS GRP0020;
  CFAIL CRT0040;
ELABEL;

LABEL CRT0040;          /* LOOKING FOR ACTION: DELETE          */
  CRITERIA;
    FIELD (=PACTACTN) VALUE('DELETE ');
  ECRITERIA;
    POPULATE;
    EPOPULATE;

  CPASS GRP0020;
  CFAIL CRT0045;
ELABEL;
```

```
LABEL CRT0045;      /* LOOKING FOR ACTION: RETRIEVE      */
CRITERIA;
  FIELD (=PACTACTN) VALUE('RETRIEVE');
ECRITERIA;
  POPULATE;
  EPOPULATE;

CPASS GRP0030;
CFAIL CRT0050;
ELABEL;

LABEL CRT0050;      /* LOOKING FOR ACTION: ARCHIVE      */
CRITERIA;
  FIELD (=PACTACTN) VALUE('ARCHIVE ');
ECRITERIA;
  POPULATE;
  EPOPULATE;

CPASS GRP0020;
CFAIL CRT0055;
ELABEL;

LABEL CRT0055;      /* LOOKING FOR ACTION: MOVE      */
CRITERIA;
  FIELD (=PACTACTN) VALUE('MOVE  ');
ECRITERIA;
  POPULATE;
  EPOPULATE;

CPASS GRP0040;
CFAIL CRT0060;
ELABEL;

LABEL CRT0060;      /* LOOKING FOR ACTION: TRANSFER  */
CRITERIA;
  FIELD (=PACTACTN) VALUE('TRANSFER');
ECRITERIA;
  POPULATE;
  EPOPULATE;

CPASS GRP0040;
ELABEL;
```

6.1 Syntax Example

```
LABEL GRP0010;      /* FOR PACKAGE ACTION: ADD          */
RC (0);            /*                                UPDATE      */
CRITERIA;         /*                                RESTORE     */
ECRITERIA;
POPULATE;

/******
 * PACTACTN: ACTION NAME          *
 * PACTCOMM: ACTION COMMENT      *
 * PACTSDSN: DATA SET NAME      *
 * PACTSMBR: MEMBER              *
 * PACTTENV: ENVIRONMENT NAME    *
 * PACTTSYS: SYSTEM NAME        *
 * PACTTSBS: SUBSYSTEM NAME     *
 * PACTTYP: TYPE                 *
 * PACTTSTG: STAGE NAME         *
 * PACTTELM: ELEMENT NAME       *
*****/
FIELD PANEL(BLG0C030) INDEX(S0E01)
TEXT ('ACTION: =PACTACTN =PACTCOMM')
TEXT ('ELEMENT: =PACTTENV / =PACTTSYS / =PACTTSBS / ') -
('=PACTTYP / =PACTTSTG / =PACTTELMFN ');
EPOPULATE;
CPASS GRP0050;
ELABEL;

LABEL GRP0020;      /* FOR PACKAGE ACTIONS: GENERATE          */
RC (0);            /*                                DELETE     */
CRITERIA;         /*                                ARCHIVE    */
ECRITERIA;
POPULATE;

/******
 * PACTACTN: ACTION NAME          *
 * PACTCOMM: ACTION COMMENT      *
 * PACTSENV: ENVIRONMENT NAME    *
 * PACTSSYS: SYSTEM NAME        *
 * PACTSSBS: SUBSYSTEM NAME     *
 * PACTSTYP: TYPE                 *
 * PACTSSTG: STAGE NAME         *
 * PACTSELM: ELEMENT NAME       *
*****/
FIELD PANEL(BLG0C030) INDEX(S0E01)
TEXT ('ACTION: =PACTACTN =PACTCOMM')
TEXT ('ELEMENT: =PACTSENV / =PACTSSYS / =PACTSSBS / ') -
('=PACTSTYP / =PACTSSTG / =PACTSELMFN ');
EPOPULATE;
ELABEL;
```

```

LABEL GRP0030;          /* FOR PACKAGE ACTIONS: RETRIEVE      */
RC (0);                /*                                          */
CRITERIA;
ECRITERIA;
POPULATE;

/* ***** */
* PACTACTN: ACTION NAME *
* PACTCOMM: ACTION COMMENT *
* PACTTDSN: DATA SET NAME *
* PACTTMBR: MEMBER *
* PACTSENV: ENVIRONMENT NAME *
* PACTSSYS: SYSTEM NAME *
* PACTSSBS: SUBSYSTEM NAME *
* PACTSTYP: TYPE *
* PACTSSTG: STAGE NAME *
* PACTSELM: ELEMENT NAME *
/* ***** */
FIELD PANEL(BLG0C030) INDEX(S0E01)
TEXT ('ACTION: =PACTACTN =PACTCOMM')
TEXT ('ELEMENT: =PACTSENV / =PACTSSYS / =PACTSSBS / ') -
('=PACTSTYP / =PACTSSTG / =PACTSELMFN ');
EPOPULATE;
CPASS GRP0070;
ELABEL;

LABEL GRP0040;          /* FOR PACKAGE ACTIONS: MOVE          */
RC (0);                /* TRANSFER                            */
CRITERIA;
ECRITERIA;
POPULATE;

/* ***** */
* PACTACTN: ACTION NAME *
* PACTCOMM: ACTION COMMENT *
* PACTSENV: ENVIRONMENT NAME *
* PACTSSYS: SYSTEM NAME *
* PACTSSBS: SUBSYSTEM NAME *
* PACTSTYP: TYPE *
* PACTSSTG: STAGE NAME *
* PACTSELM: ELEMENT NAME *
* PACTTENV: ENVIRONMENT NAME *
* PACTTSYS: SYSTEM NAME *
* PACTTSBS: SUBSYSTEM NAME *
* PACTTTYP: TYPE *
* PACTTSTG: STAGE NAME *
* PACTTELM: ELEMENT NAME *
/* ***** */

```

```
FIELD PANEL(BLG0C030) INDEX(S0E01)
TEXT ('ACTION: =PACTACTN =PACTCOMM')
TEXT ('SOURCE: =PACTSENV / =PACTSSYS / =PACTSSBS / ') -
      ('=PACTSTYP / =PACTSSTG / =PACTSELMFN ')
TEXT ('TARGET: =PACTTENV / =PACTTSYS / =PACTTSBS / ') -
      ('=PACTTTYP / =PACTTSTG / =PACTTELMFN ');
EPOPULATE;
ELABEL;

LABEL GRP0050;          /* ADD PATH INFO FOR ADD, UPDATE */
RC (0);                /* */
CRITERIA;
FIELD (=PACTSFLAG) VALUE('P');
ECRITERIA;
POPULATE;
FIELD PANEL(BLG0C030) INDEX(S0E01)
TEXT ('SOURCE: =PACTSPANM =PACTSFINM ');
EPOPULATE;

CFAIL GRP0060;
ELABEL;

LABEL GRP0060;          /* ADD DATA SET INFO FOR ADD, UPDATE */
RC (0);                /* RESTORE */
CRITERIA;
FIELD (=PACTSDSN) VALUE(NONBLANK);
ECRITERIA;
POPULATE;
FIELD PANEL(BLG0C030) INDEX(S0E01)
TEXT ('SOURCE: =PACTSDSN =PACTSMBR ');
EPOPULATE;
ELABEL;

LABEL GRP0070;          /* ADD PATH INFO FOR RETRIEVE */
RC (0);                /* */
CRITERIA;
FIELD (=PACTTFLAG) VALUE('P');
ECRITERIA;
POPULATE;
FIELD PANEL(BLG0C030) INDEX(S0E01)
TEXT ('TARGET: =PACTTPANM =PACTTFINM ');
EPOPULATE;

CFAIL GRP0080;
ELABEL;
```

```

LABEL GRP0080;          /* ADD DATA SET INFO FOR  RETRIEVE      */
RC (0);                /*                                RESTORE      */
CRITERIA;
  FIELD (=PACTTDSN) VALUE(NONBLANK);
ECRITERIA;
  POPULATE;
  FIELD PANEL(BLG0C030) INDEX(S0E01)
  TEXT ('TARGET: =PACTTDSN =PACTTMBR ');
EPOPULATE;
ELABEL;
EUSE;

/*****
* EXIT POINTS: REVIEW/AFTER EXECUTE/AFTER      *
*              SHIP/AFTER   CONFIRM/AFTER      *
*              BACKOUT/AFTER BACKIN/AFTER      *
*              COMMIT/AFTER                    *
*****/

USE ((REVIEW, AFTER),
     (EXECUTE, AFTER),
     (SHIP, AFTER),
     (CONFIRM, AFTER),
     (BACKOUT,AFTER),
     (BACKIN,AFTER),
     (COMMIT,AFTER));

LABEL GRP0010;          /* MAINLINE - ALL POINTS      */
RC(0);
CRITERIA;
ECRITERIA;
POPULATE;

/*****
* PREQCOMM:  PACKAGE COMMENT                    *
* PREQWSD:   EXECUTION WINDOW START DATE      *
* PREQWST:   EXECUTION WINDOW START TIME      *
* PREQWED:   EXECUTION WINDOW END DATE        *
* PREQWET:   EXECUTION WINDOW END TIME        *
* PECBUSER:  CURRENT USER ID                  *
* PECBFNM:   PACKAGE FUNCTION                 *
* PECBBANM:  BEFORE OR AFTER                  *
* PECBNDRC:  NDVR HIGH RETURN CODE            *
* PHDRSTAT:  PACKAGE STATUS                   *
*****/

```

```

FIELD PANEL(BLG6DSAB) INDEX(S0E0F) VALUE(=PREQCOMM);
FIELD PANEL(BLG6SCHD) INDEX(S0C41) VALUE(=PREQWSD);
FIELD PANEL(BLG6SCHT) INDEX(S0C6E) VALUE(=PREQWST);
FIELD PANEL(BLG6TARD) INDEX(S0C42) VALUE(=PREQWED);
FIELD PANEL(BLG6TART) INDEX(S0C6F) VALUE(=PREQWET);
FIELD PANEL(BLG600CN) INDEX(S0B5C) VALUE(=PECBUSER);
FIELD PANEL(BLG6PHAC) INDEX(S0C0B) VALUE(=PECBFNNM);
FIELD PANEL(BLG0C010) INDEX(S0E01)
  TEXT ('PACKAGE PROCESSING COMPLETED RC = =PECBNDRC')
  TEXT ('INFO ACCESSED FROM PACKAGE EXIT =PECBBANM / =PECBFNNM')
  TEXT ('PKG STATUS: =PHDRSTAT  USER: =PECBUSER');
EPOPULATE;

CPASS EXE0010;
ELABEL;

LABEL EXE0010;          /* EXECUTE AFTER - LOG SECTION          */
CRITERIA;
  FIELD (=PECBFNNM) VALUE ('EXECUTE ');
ECRITERIA;
  POPULATE;

  /******
  * PHDRXD:   EXECUTION DATE           *
  * PHDRXT:   EXECUTION TIME           *
  * PHDREXD:  END EXECUTION DATE       *
  * PHDREEXT: END EXECUTION TIME       *
  *****/
FIELD PANEL(BLG0C010) INDEX(S0E01)
  TEXT ('EXECUTION:   =PHDRXD  =PHDRXT')
  TEXT ('END EXECUTION: =PHDREXD  =PHDREEXT');
EPOPULATE;

CFAIL BK00010;
ELABEL;

LABEL BK00010;          /* BACKOUT AFTER - LOG SECTION          */
CRITERIA;
  FIELD (=PECBFNNM) VALUE ('BACKOUT ');
ECRITERIA;
  POPULATE;

  /******
  * PHDRBOST: BACKOUT STATUS           *
  * PHDRBOD:  BACKOUT DATE             *
  * PHDRBOT:  BACKOUT TIME             *
  *****/
FIELD PANEL(BLG0C010) INDEX(S0E01)
  TEXT ('BACKOUT:     =PHDRBOD  =PHDRBOT  STATUS: =PHDRBOST');
EPOPULATE;

CFAIL BKI0010;
ELABEL;

LABEL BKI0010;          /* BACKIN AFTER - LOG SECTION          */
CRITERIA;
  FIELD (=PECBFNNM) VALUE ('BACKIN ');
ECRITERIA;
  POPULATE;

  /******

```

```

                * PHDRBID:   BACKIN DATE           *
                * PHDRBIT:   BACKIN TIME           *
                *****/
FIELD PANEL(BLG0C010) INDEX(S0E01)
TEXT ('BACKIN:           =PHDRBID =PHDRBIT');
EPOPULATE;

CFAIL COM0010; ELABEL;

LABEL COM0010;      /* COMMIT AFTER - LOG SECTION      */
CRITERIA;
FIELD (=PECBFNM) VALUE ('COMMIT ');
ECRITERIA;
POPULATE;

                /******
                * PHDRCMD:   COMMIT DATE           *
                * PHDRCMT:   COMMIT TIME           *
                *****/
FIELD PANEL(BLG0C010) INDEX(S0E01)
TEXT ('COMMIT:           =PHDRCMD =PHDRCMT');
EPOPULATE;

CFAIL SHP0010;
ELABEL;

LABEL SHP0010;      /* SHIP XMIT AFTER - LOG SECTION      */
CRITERIA;
FIELD (=PECBSFNM) VALUE ('XMIT ');
ECRITERIA;
POPULATE;

                /******
                * PREQDEST:  SHIP DESTINATION       *
                * PREQSTYP:  SHIPMENT TYPE          *
                * PREQSCMP:  SHIP COMPLEMENTS (Y/N) *
                *****/
FIELD PANEL(BLG0C010) INDEX(S0E01)
TEXT ('SHIP:  DESTINATION =PREQDEST TYPE =PREQSTYP ') -
('COMPLEMENTS =PREQSCMP');
EPOPULATE;

CFAIL CFM0010;
ELABEL;

LABEL CFM0010;      /* SHIP CONFIRM AFTER - LOG SECTION      */
CRITERIA;
FIELD (=PECBSFNM) VALUE ('CONFIRM ');
ECRITERIA;
POPULATE;

                /******
                * PREQDEST:  SHIP DESTINATION       *
                * PREQSCNF:  SHIP CONFIRMATIN TYPE   *
                * PREQSRES:  SHIP CONFIRMATION RESULTS *
                * PREQSRCV:  SHIP CONFIRM RC VALUE    *
                *****/

```

6.1 Syntax Example

```
FIELD PANEL(BLG0C010) INDEX(S0E01)
TEXT ('SHIP:  DESTINATION =PREQDEST  CONFIRM TYPE =PREQSCNF') -
('          CONFIRMATION RESULTS =PREQSRES =PREQSRCV');
EPOPULATE;

ELABEL;

EUSE;

/*****
* EXIT POINTS: ADD/BEFORE      UPDATE/BEFORE      *
*                RESTORE/BEFORE RETRIEVE/BEFORE  *
*                ARCHIVE/BEFORE GENERATE/BEFORE   *
*                DELETE/BEFORE  MOVE/BEFORE       *
*                TRANSFER/BEFORE                   *
*****/

USE ((ADD, BEFORE),
    (UPDATE, BEFORE),
    (RESTORE, BEFORE),
    (RETRIEVE, BEFORE),
    (ARCHIVE, BEFORE),
    (GENERATE, BEFORE),
    (DELETE, BEFORE),
    (MOVE, BEFORE),
    (TRANSFER, BEFORE));

NOTFOUND (RETURN);
FOUND (RETURN);      /* GREAT -- JUST RETURN      */

EUSE;

/*****
* ON PACKAGE ACTIONS - JUST BEFORE EXECUTION.  *
* PURPOSE:  RECORD ACTION RETURN CODE ON      *
*           THE CCID ACTIVITY RECORD.         *
*****/

USE ((ADD, AFTER),
    (UPDATE, AFTER),
    (RESTORE, AFTER),
    (RETRIEVE, AFTER),
    (ARCHIVE, AFTER),
    (GENERATE, AFTER),
    (DELETE, AFTER),
    (MOVE, AFTER),
    (TRANSFER, AFTER));

NOTFOUND (ERR0010);
FOUND (GRP0010);
```

```
LABEL GRP0010;          /* ALL PKG ACTION POINTS          */
RC(0);
CRITERIA;
ECRITERIA;
POPULATE;
  FIELD PANEL(BLG0C030) INDEX(S0E01)
  TEXT ('ACTION: =ACTION =DATE =TIME USER: =USER RC: =ACTRC')
  TEXT ('COMMENT: =COMM');
EPOPULATE;

ELABEL;

LABEL ERR0010;          /* TELL USER          */
MSG ('ACTIVITY CCID =CCID RECORD NOT FOUND');
ELABEL;

EUSE;

/*****
* EXIT POINT: DELETE AFTER          *
* PURPOSE: DELETE CHANGE AND ACTIVITY PKG          *
*           RECORDS, THEN RETURN.          *
*****/

USE (PDELETE, AFTER);

NOTFOUND (RETURN);
FOUND (DELETE,RETURN);

EUSE;

ETABLE;
```


Chapter 7. Displaying Info/Man Information

7.1 Displaying Information Stored in Info/Man

The Endeavor Info/Man interface allows users, in foreground, to:

- Produce selection lists of CCIDs when requesting stand-alone actions.
- Display the information stored in Info/Man about a CCID.
- Display the information stored in Info/Man about an Endeavor package.

The Info/Man record display panel is the same for packages and for CCIDs. The information displayed is basic and some fields may be blank when they are not used in a site's Info/Man policy.

7.2 Listing and Displaying CCID Information

When the Info/Man interface is active, users can produce a list of available CCIDs by wildcarding the CCID field. The CCID list can be produced when requesting the following stand-alone actions in foreground and when generating SCL for execution in batch.

- ADD/UPDATE
- MOVE
- RETRIEVE
- GENERATE
- DELETE
- TRANSFER
- ARCHIVE

The interface searches the Change and Problem records in Info/Man and returns a list of CCIDs that have an Info/Man type of ACTION, an Info/Man status other than CLOSED, and meet the wildcard criteria.

Note: Wildcarding CCIDs is not allowed when building packages.

When a CCID value is fully specified, a list is **not** produced. If no Info/Man records meet the wildcard criteria, a CCID prompt panel appears. The user can type either a fully specified CCID or a wildcarded CCID on this panel. This search process can be repeated multiple times if necessary.

From the CCID list, you can either view Info/Man correlation information for one or more CCIDs on the list, or select one of the CCIDs for the action being requested.

CAUTION:

If you use the Info/Man interface, make sure to set up all your Endeavor systems to require CCIDs. See the *Administration Guide* for instructions.

When using the Info/Man interface, Endeavor CCIDs must start with an alpha character. In addition, Info/man stores only the first eight characters of the Endeavor CCID.

7.2.1 CCID List Panel

The CCID List panel appears when you specify a full or partial wildcard in the CCID field on an action request panel, as shown below.

```

----- ENDEVOR FOR OS/390 CCID SELECTION LIST ----- ROW 1 TO 18 OF 18
COMMAND ==> SCROLL ==> PAGE
ENVIRONMENT:          SYSTEM:          SUBSYSTEM:
TYPE:                STAGE:          ELEMENT:
'S' - SELECT A CCID FROM INFO 'B' - BROWSE INFO/MAN RECORD
HIT PFKEY3 (END) TO RETURN TO PROMPT PANEL. FROM THE PROMPT PANEL
EITHER USE THE PREVIOUS CCID OR REDRIVE CCID LIST.
RNID/CCID  RECORD DESCRIPTION
00000033  ENDEVOR INTERFACE&COLON. ACTION LEVEL
00000034  SEE FREE FORM TEXT FOR ACTION AUDIT TRAIL
00000035  ACTION LEVEL INTERFACE RECORD -
00000047  TEST RECORD FOR E/OS/390 INTERFACE - CHANGE ACT

```

The Endeavor location fields on this panel display the current environment, system, subsystem, type, and stage, and the element against which the action is being requested. Other panel components are described below.

Panel Component	Description
S--SELECT	Type this option next to the desired CCID to choose that CCID from this list.
B--BROWSE	Use this option to tell Endeavor to display the Info/Man record associated with this CCID.
RNID/CCID	This field displays Info/Man record IDs (RNIDs).
RECORD DESCRIPTION	This field displays the description associated with the Info/Man RNID.

7.2.2 Action Prompt Panel

When no CCIDs on the Info/Man side meet the wildcard criteria and the system named on the panel has been defined to require CCIDs, the Action Prompt panel is displayed, allowing the user to enter a new CCID value.

The CCID provided on the prompt panel can also be wildcarded. If CCIDs on the Info/Man side meet this wildcard criteria, a list selection list appears.

After a user selects a CCID, the interface returns the original action panel.

7.2.3 CCID Correlation Panel

When you type **B** (Browse) next to a CCID on the CCID List panel, and Info/Man contains information about that CCID, the CCID correlation panel appears when you press ENTER. This panel is shown below.

```
----- ACTION PROMPT ----- CCID REQUIRED
COMMAND ==>
  Specification Required:
    CCID:   Y (Y/N)
    COMMENT: Y (Y/N)
  Action: ADD           Element: CITB000X
  Environment: DEV      System: NDVR370      Subsystem: INFO
  Type:   ISPP          Stage: DEVSTG1
  CCID   ==>
  COMMENT ==> TESTING INFO
```

From this panel you can:

- Press PF3 to return to Endeavor.
- Press PF1 to access help for this panel.

7.3 Displaying Package Information

When package information exists in the Info/Man database, an additional option, **CI - Display Correlation Information**, appears on the following panels:

- Package Display.
- Create/Modify Package.
- Cast Package.
- Review Package.
- Execute/Submit Package.
- Backout Package.
- Commit Package. On this panel the option appears as **I - Display Correlation Information**.

If your site does not use Info/Man, or Info/Man correlation information does not exist, this option does not appear on the panel.

The Display Package panel below is for a site with Info/Man enabled.

```

DISPLAY _____ PACKAGE DISPLAY _____
OPTION ==>
  blank - Display Action Summary      B - Display Backout Information
  A - Display Approvers                S - Display SCL
  R - Display Cast Report              CI - Display Correlation Information
PACKAGE ID: EITEST1                   STATUS: APPROVED
DESCRIPTION: *
PACKAGE TYPE: STANDARD
SHARABLE PACKAGE: N
BACKOUT ENABLED: N
EXECUTION WINDOW FROM: 15JUN00 00:00  TO: 30JUN00 00:00
                        USER ID DATE   TIME
CREATED:                DA1DM47 15JUN00 13:41
LAST UPDATED:
CAST:                   DA1DM47 16JUN00 12:19
APPROVED/DENIED:        16JUN00 12:19
EXECUTED:
BACKED OUT:
BACKED IN:
COMMITTED:
                                ENDEVOR RC:

```

The **CI** option behaves as follows.

- The option appears on the appropriate panels if Info/Man is installed, regardless of whether Info/Man is storing any package information.
- If no correlation exists, the option and the text are shown in normal intensity.
- If correlation information exists, the option is shown in high intensity when certain conditions are met. These conditions are described below.
- If the user selects the **CI** option when no correlation information exists, the ISPF short message **INVALID OPTION** appears after you press ENTER.

The **CI** option activates in either of two ways.

- If the user creates the parent record in Info/Man using the Endeavor scripts, then **CI** turns to high intensity when the package is cast.
- If the user creates the parent record directly in Info/Man, then **CI** turns to high intensity at the first exit point invoked from Endeavor after the Cast action.

This means that the **CI** option may stay at low intensity, and therefore unavailable, even when package information exists in Info/Man. For example, if a user creates parent records for packages directly in Info/Man, and builds an E/INFO table with only a (CREATE,AFTER) USE statement, the **CI** option will not turn to high intensity.

When you type **CI** then press ENTER, this correlation panel appears.

```

----- ENDEVOR/INFORMATION MANAGEMENT CORRELATION -----
OPTION ==>
Record Identifier: 00000029          Assignee: QAGROUP
Co-requisites: EITEST1             Coordinator: B. SMITH
Description: ENDEVOR/INFO PACKAGE TEST PACKAGE=EITEST1
Entered Date: 05/23/00             Approval Status: PENDING
Entered Time: 11:14                Change Status: OPEN
Last Altered Date: 06/01/00        Change Type: PACKAGE
Last Altered Time: 13:16           Entry Priv Class: MASTER
Planned Start Date: 06/15/00       Owing Priv Class:
Required Date: 07/01/00            Current Phase: EXECUTE
Completion Date:                    Current Priority: 10

```

This panel is display-only. Fields cannot be modified from the Package Correlation panel.

Press PF3 to return to the package panel from which you accessed the correlation information.

Chapter 8. Interface Reports

8.1 About the Interface Reports

The Endeavor Info/Man interface reports are designed to show the status of records in Info/Man and to show their relationship to Endeavor information. The reports provide an audit trail for packages, package actions, and CCIDs.

The reports are built using the Info/Man Report Format Table (RFT) facility. The RFT streams that produce each report are listed below

This RFT stream	Produces this interface report
EINTACH0	Action Change Record Summary.
EINTACH1	Action Change Record Detail.
EINTAPB0	Action Problem Record Summary.
EINTAPB1	Action Problem Record Detail.
EINTPKG0	Package Parent Record Summary.
EINTPKG1	Package/Activity Record Summary.
EINTPKG2	Package/Activity Record Detail.

These RFT streams are located in `iprfx.igual.SOURCE`.

8.2 Running the Reports

You can run these reports either in native Info/Man, or in batch.

8.2.1 Running the Reports in Native Info/Man

To run the reports in native Info/Man, type the following on the Info/Man command line, then press ENTER:

```
REPORT,8,RFT_name
```

In this syntax, RFT name can be any of the RFT streams in the table in the preceding section. For example, the following command generates the Action Change Record Summary:

```
REPORT,8,EINTACH0
```

Info/Man prompts you for the report destination if the session defaults for print, report, and for customized report destination are not in your Info/Man profile. To update these fields in your profile select option **2** (PROFILE) from the Info/Man Main Menu to set the values, or contact your site Info/man administrator.

8.2.2 Running the Reports in Batch

The user submitting batch report jobs must be defined as a user to Info/Man. There are two additional issues that you must address before running interface reports in batch.

- Contention for the ISPF profile data set.
- Setting defaults in Info/Man for output destination and class for reports.

8.2.2.1 Contention for the ISPF Profile Data Set

Info/Man requires access to the ISPF profile data set when executing report requests in batch. This means that contention occurs when you use the same ISPF profile data set to run reports in batch and to have an active ISPF/TSO session. To resolve this problem, copy your interactive ISPF profile data set and rename it userid.igual.ISPPROF. Use the renamed profile for use when logging on in batch.

8.2.2.2 Setting Defaults in Info/Man for Output Destination and Class for Reports

Info/Man stores user profile information, including default output destination and class information for reports, in the BLG0PROF member in the ISPF profile data set.

These defaults can be set in either of two ways:

- Each user can set their own options by invoking an interactive Info/Man session and updating their own BLG0PROF profile member.
- The Info/Man administrator can set up a default profile member for all users.

To run the reports in batch, modify JCL member BC1JEI35 in the iprfx.igual.JCLLIB data set. The default input stream is all reports. You can select the reports you want to run by modifying the input stream.

BC1TEI35 JCL is shown below.

```
// (JOB CARD)
// *
// *****
// * THE PURPOSE OF THIS JOB IS TO RUN CA-ENDEAVOR/INFO INTERFACE REPORTS *
// * IN BATCH, OUTSIDE OF AN INTERACTIVE SESSION. *
// * *
// * NOTE: IN THE INPUT STREAM IS A VARIABLE CALLED USERID. THIS *
// * MUST BE FILLED IN WITH A TSO USERID. *
// * *
// * NOTE: IN THE INPUT STREAM IS A ENTRY FOR THE BLGSESXX PARM. *
// * THIS IS THE SAME PARAMETER USED WITH THE INTERFACE *
// * AS WELL AS NATIVE INFO. REPLACE THE 'XX' WITH THE *
// * BLGSES SUFFIX USED IN THE BC1TEI90 START UP BLOCK. *
// * *
// * NOTE: THE INPUT STREAM IS SET UP TO RUN ALL POSSIBLE REPORTS. *
// * USER SHOULD DELETE THOSE THEY DO NOT WANT TO RUN. *
// *****
// *
// * SAMPLE JCL TO ALLOW USER TO RUN INFO/MAN REPORTS FROM
// * CA-ENDEAVOR IN BATCH MODE.
// *
// * FOR MORE INFORMATION ON HOW TO RUN INFO/MAN IN BATCH,
// * PLEASE REFER TO THE IBM MANUAL 'PLANNING AND INSTALLING THE
// * INFORMATION/FAMILY', THE CHAPTER ENTITLED INVOKING THE "INFO-
// * MAN LICENSED PROGRAMS" HAS MORE INFORMATION ON OPTIONS AND
// * DDNAMES.
// *
// *****
```

```
//INFORPTS EXEC PGM=IKJEFT01,DYNAMNBR=25,REGION=2048K
//STEPLIB DD DSN=uprfx.uqua1.LOADLIB,DISP=SHR
// DD DSN=uprfx.uqua1.LOAD,DISP=SHR <==INFOMAN LOAD LIB.
//*
//ISPPROF DD DSN=uprfx.uqua1.PROF,DISP=SHR
//*
//SYSPROC DD DISP=SHR,DSN=uprfx.uqua1.ISRCLIB
//*
//ISPMLIB DD DISP=SHR,DSN=uprfx.uqua1.ISPMLIB
//*
//ISPPLIB DD DISP=SHR,DSN=uprfx.uqua1.ISPPLIB
//*
//ISPSLIB DD DISP=SHR,DSN=uprfx.uqua1.ISPSLIB
//*
//ISPTLIB DD DISP=SHR,DSN=uprfx.uqua1.ISPTLIB
//*
//BLGRPTS DD SYSOUT=*
//*
//SYSTSPRT DD SYSOUT=*
//SYSTSIN DD *
PROFILE PREFIX(USERID)
ISPSTART PGM(BLGINIT) PARM(IRC(REPORT,8,EINTACH0,QUIT) SESS(XX))
ISPSTART PGM(BLGINIT) PARM(IRC(REPORT,8,EINTACH1,QUIT) SESS(XX))
ISPSTART PGM(BLGINIT) PARM(IRC(REPORT,8,EINTAPB0,QUIT) SESS(XX))
ISPSTART PGM(BLGINIT) PARM(IRC(REPORT,8,EINTAPB1,QUIT) SESS(XX))
ISPSTART PGM(BLGINIT) PARM(IRC(REPORT,8,EINTPKG0,QUIT) SESS(XX))
ISPSTART PGM(BLGINIT) PARM(IRC(REPORT,8,EINTPKG1,QUIT) SESS(XX))
ISPSTART PGM(BLGINIT) PARM(IRC(REPORT,8,EINTPKG2,QUIT) SESS(XX))
/*
```

8.3 Customizing the Reports

You can customize the standard reports, or create your own reports using Info/Man's RFT facility.

For information on the Info/Man RFT facility, see your Info/Man administrator.

8.4 Action Change Record Summary

The Action Change Record Summary report lists the Info/Man Change records recorded as type ACTION. The list is sorted by CCID.

Use this report as an inventory list of CCIDs, and to monitor the status of each CCID.

CCID	DESCRIPTION OF INFO CHANGE RECORD	LAST MOD USERID	LAST MOD DATE	LAST MOD TIME	CHANGE STATUS	APPROVAL STATUS
00000035	ACTION LEVEL INTERFACE RECORD -	DA1CC47	01/16/00	12:07	CLOSED	REJECTED
00000034	SEE FREE FORM TEXT FOR ACTION AUDIT TRAI	DA1CC47	02/07/00	14:06	INITIAL	PENDING
C0000001	USER VALUE ON CREATE MUST START WITH CHA	DA1CC47	01/16/00	12:13	OPEN	
DA1SJ27	API CREATE CHG RECORD	DA1J033	02/23/00	10:33	OPEN	
DA2SRJ	API CREATE CHG RECORD	DA1J033	02/23/00	10:21	OPEN	

The table below describes the report fields.

Field	Description
CCID	The Endeavor CCID for the action, as recorded in the Info/Man RNID field
Description of Info/Man Change Record	The description of the action from the Populate block of the batch utility, recorded in the freeform text field in the Info/Man record.
Last Mod Userid	The ID of the user who last modified the Info/Man record.
Last Mod Date	The date of the last modification.
Last Mod Time	The time of the last modification.
Change Status	Change status of the action.
Approval Status	Approval status of the action.

8.5 Action Change Record Detail

The Action Change Record Detail report prints one page for each Info/Man Change record listed on the Action Change Detail Summary report. In addition to repeating the information in the summary report, this report prints any freeform text associated with the Change record. There is one freeform text entry for each action associated with the CCID.

Use this report as an audit trail of the actions associated with a particular CCID.

CCID	DESCRIPTION OF INFO CHANGE RECORD	LAST MOD USERID	LAST MOD DATE	LAST MOD TIME	CHANGE STATUS	APPROVAL STATUS
C0000001 DA1SJ27	USER VALUE ON CREATE MUST START WITH CHA API CREATE CHG RECORD FREE FORM TEXT	DA1CC47 DA1J033	01/16/00 02/23/00	12:13 10:33	OPEN OPEN	
	ACTION PROCESSING COMPLETED RC = 0 ELEMENT: QA / QA36 / B9212C / ASMPROGS / QASTG1 / INFPROG4 ACTION: ADD USER: DA1SJ27 DATE: 02/22/00 TIME: 16:01 ACTION PROCESSING COMPLETED RC = 4 ELEMENT: QA / QA36 / B9212C / ASMPROGS / QASTG1 / BATCELE4 ACTION: UPDATE USER: DA1SJ27 DATE: 02/23/00 TIME: 10:21 ACTION PROCESSING COMPLETED RC = 4 ELEMENT: QA / QA36 / B9212C / ASMPROGS / QASTG1 / BATCELE4 ACTION: UPDATE USER: DA1SJ27 DATE: 02/23/00 TIME: 10:33					

8.6 Action Problem Record Summary/Detail

The Action Level Interface Problem Record Summary and Detail reports contain information similar to that in the Change record reports.

Use these reports the same way that you would use the Action Change Record reports: the Summary report as an inventory list for CCIDs, and the Detail report as an audit trail for actions associated with CCIDs.

8.6.1 Summary Report

CCID	DESCRIPTION OF INFO PROBLEM RECORD	LAST MOD USERID	LAST MOD DATE	LAST MOD TIME	PROBLEM STATUS	DATE ENTERED	ASSIGNEE NAME
00000010	TEST PROBLEM NUMBER	DA1CC47	08/12/99	19:28	INITIAL	01/08/95	

8.6.2 Detail Report

CCID	DESCRIPTION OF INFO PROBLEM RECORD	LAST MOD USERID	LAST MOD DATE	LAST MOD TIME	PROBLEM STATUS	DATE ENTERED	ASSIGNEE NAME
00000010	TEST PROBLEM NUMBER FREE FORM TEXT	DA1CC47	08/12/99	19:28	INITIAL	01/08/95	
THIS IS FREE FORM DATA TO SEE IF THE REPORT SHELL WORKED.							

The table below describes the report fields.

Field	Description
CCID	The Endeavor CCID. For action problem records, this is the same as the Info/Man RNID.
Description	The description of the action from the Populate block of the batch utility, recorded in the free form text field in the Info/Man record.
Last Mod Userid	The ID of the user who last modified the Info/Man record.
Last Mod Date	The date of the last modification.
Last Mod Time	The time of the last modification.
Problem Status	The status of the problem.
Date entered	The date this problem record was created in Info/Man.
Assignee name	The person who has been assigned the problem.
Free form text	Appears on the detail report only. Displays information about this action.

8.7 Package Parent Record Summary

The Package Level Interface Parent Record Summary report lists Info/Man Change records with a type of PACKAGE.

DATE 02/24/00		***		ENDEAVOR FOR OS/390 - INFO/MAN PACKAGE LEVEL PARENT RECORD SUMMARY REPORT						***		PAGE 1	
TIME 14:13:00													

PACKAGE LEVEL INTERFACE RECORDS													

CURR STAT	INFO RNID	CA-Endevor PACKAGE	INFO/MAN UPDATE	USER	PACKAGE START	EXECUTION END	WINDOW START	WINDOW END	APPROVER STATUS	DESCRIPTION			
----	----	-----	-----	-----	-----	-----	-----	-----	-----	-----			
OPEN	00000029	EITEST1	09/04/99	DA1DM47	02/15/99	10:00	02/25/00	10:00	PENDING	E/INFO PKG TEST PACKAGE=E			
OPEN	00000036		10/20/99	DA1CC47	10/20/99	10:00	02/25/00	10:00	PENDING	PACKAGE:EPKG0002			
OPEN	00000039	EPCK0003	08/19/99	DA1CC47	02/15/99	10:00	02/25/00	10:00	PENDING	PACKAGE:EPKG0003			
OPEN	00000055	OLEN06	10/23/99	DA1CC47	10/23/99	00:00	12/31/99	00:00		E/INFO CREATED CHANGE RECORD			
OPEN	00000056	OLEN07	10/23/99	DA1CC47	10/23/99	00:00	12/31/99	00:00		E/INFO CREATED CHANGE RECORD			

The table below describes the report fields.

Field	Description
Curr Stat	Current status of the package change record.
Info/Man RNID	The Info/Man RNID associated with this change record. The RNID is cross referenced with the correlation field in the package header in Endeavor.
Endevor package	The ID of the package in Endeavor. The Endeavor package ID is cross referenced with the Info/Man corequisite field.
INFO/MAN update	The date when this Info/Man record was last updated.
User	The ID of the user who last updated the record.
Package execution window - start	The date and time of the start of the package execution window.
Package execution window - end	The date and time of the end of the package execution window.
Approval status	User-defined status of the approval process.
Description	Description of the package change record.

8.8 Package/Activity Record Summary

The Package/Activity Record Summary report lists package parent record summary information along with summary information for any associated package actions. The report is sorted by the package parent record Info/Man RNID.

Curr Stat	Info RNID	CA-Endevor PACKAGE	Info/Man Update	User	Package Execution Start	Window End	Approval Status	Last Package Function
OPEN	00000036	PACKAGEI	10/20/99	DA1CC47	10/20/99 10:00	02/25/00 10:00	PENDING	CAST
DESCRIPTION								
PACKAGE:EPKG0002								
ACTIVITY RECORDS: ONE PER UNIQUE CA-Endevor CCID IN PACKAGE								
RNID	CCID	Last Update Date	User	Activity Type	Description			
00000037	CCID	02/07/00	DA1CC47	PKG/ACT	CCID:CCID IN PACKAGE:EPKG0002			
00000038	TEST	02/07/00	DA1CC47	PKG/ACT	CCID:TEST IN PACKAGE:EPKG0002			

8.8.1 Package Information

Field	Description
Curr Stat	Current status of the package change record.
INFO RNID	The Info/Man RNID associated with this change record. The RNID is cross referenced with the CORRELATION field in the package header in Endeavor.
Endevor package	The ID of the package in Endeavor. The Endeavor package ID is cross referenced with the Info/Man COREQUISITE field.
INFO/MAN update	The date when this Info/Man record was last updated.
User	The ID of the user who last updated the record.
Package execution window - start	The date and time when the package execution window starts.
Package execution window - end	The date and time of the end of the package execution window.
Approval status	User-defined status of the approval process.
Package status	Current status of the package in Endeavor.
Last package function	Last package function executed in Endeavor.
Description	Description of the package.

8.8.2 Activity Information

Field	Description
RNID	RNID associated with this activity record.
CCID	Endevor CCID associated with this activity record.
Last update - date	Date of the last update.
Last update - user	ID of the user who last updated this record.
Activity type	Always PKG/ACT for activity records.
Description	Description of the activity record.

8.9 Package/Activity Record Detail--Package Detail

The Package/Activity Record Detail report provides detail information for both the package parent record and any associated activity records. Detail information includes the summary information plus any freeform text associated with the parent package record or the activity records.

Page 1 of the report contains the information for the parent package record. The report starts a new page for each activity record.

DATE 02/24/00 *** ENDEVOR FOR OS/390 - INFO/MAN PACKAGE LEVEL ACTIVITY RECORD DETAIL REPORT *** PAGE 55							
TIME 14:15:17							

PACKAGE LEVEL INTERFACE RECORD							

CURR STAT	INFO RNID	CA-Endevor PACKAGE	INFO/MAN UPDATE	USER	PACKAGE EXECUTION START	WINDOW END	APPROVER STATUS LAST PACKAGE FUNCTION
----	----	-----	-----	-----	-----	-----	-----
OPEN	00000146	INFO2	02/23/00	DA1J033	02/23/00 00:00	12/31/99 00:00	EXECUTE
DESCRIPTION							

TEST							
FREE FORM TEXT							
PACKAGE PROCESSING COMPLETED RC = 0							
INFO ACCESSED FROM PACKAGE EXIT AFTER / CREATE							
PKG STATUS: IN-EDIT USER: DA1SJ27							
CREATE DATE: 02/23/00 TIME: 19:27							
PACKAGE PROCESSING COMPLETED RC = 0							
INFO ACCESSED FROM PACKAGE EXIT AFTER / MODIFY							
PKG STATUS: IN-EDIT USER: DA1SJ27							
CREATE DATE: 02/23/00 TIME: 19:27							
PACKAGE PROCESSING COMPLETED RC = 0							
INFO ACCESSED FROM PACKAGE EXIT AFTER / CAST							
PKG STATUS: APPROVED USER: DA1SJ27							
CREATE DATE: 02/23/00 TIME: 19:27							
CAST DATE: 02/23/00 TIME: 19:30							
PACKAGE PROCESSING COMPLETED RC = 0							
INFO ACCESSED FROM PACKAGE EXIT AFTER / EXECUTE							
PKG STATUS: EXECUTED USER: DA1SJ27							
EXECUTION: 02/23/00 19:31							
END EXECUTION: 02/23/00 19:33							

8.10 Package/Activity Detail--Activity Detail

The activity detail portion of the Package Activity Detail report starts a new page for each activity record associated with the parent change record. The information includes the same information as on the summary report, and includes freeform text associated with each activity record.

The freeform text of activity records contains one entry for each action associated with the activity record CCID.

DATE 02/24/00		*** ENDEVOR FOR OS/390 - INFO/MAN PACKAGE LEVEL ACTIVITY RECORD DETAIL REPORT			*** PAGE 56	
TIME 14:15:17						
ACTIVITY RECORDS: ONE PER UNIQUE CA-Endevor CCID IN PACKAGE INFO2						
RNID	CCID	LAST UPDATE DATE	USER	ACTIVITY TYPE	DESCRIPTION	
00000147	00000033	02/23/00	DA1J033	PKG/ACT	CCID: 00000033	IN PACKAGE: INFO2
	FREE FORM TEXT					
	ACTION: ADD SHOULD PASS VALID CCID IN INFO					
	ELEMENT: QA / QA36 / B9212C / ASMPROGS / QASTG1 / PKG5					
	SOURCE: BST.QA36.JCLLIB PROG1					
	ACTION: ADD 02/23/00 19:32 USER: DA1SJ27 RC: 0					
	COMMENT: SHOULD PASS VALID CCID IN INFO					
RNID	CCID	LAST UPDATE DATE	USER	ACTIVITY TYPE	DESCRIPTION	
00000148	DA9SRJ	02/23/00	DA1J033	PKG/ACT	CCID: DA9SRJ	IN PACKAGE: INFO2
	FREE FORM TEXT					
	ACTION: ADD SHOULD PASS CREATES CCID IN INFO					
	ELEMENT: QA / QA36 / B9212C / ASMPROGS / QASTG1 / PKG10					
	SOURCE: BST.QA36.JCLLIB PROG1					
	ACTION: ADD 02/23/00 19:32 USER: DA1SJ27 RC: 0					
	COMMENT: SHOULD PASS CREATES CCID IN INFO					

Appendix A. Information for Advanced Users

A.1 Modified Assisted Entry Panel (BLG6CORQ)

The BLG6CORQ Panel provided by standard Info/Man must be modified to allow it to store the 16-character Endeavor CCID.

Phase 2, Step 3 of the installation procedure documented in 2.3.3, "Step 3. Copy Dictionary Update" on page 2-7 explains how to modify the BLG6CORQ panel if you are working with standard Info/Man.

Info/Man administrators at sites that have already modified their Info/Man panels should do the following:

1. Create a PWORD of RNCC/ with a validation pattern of CCV15.
2. Assign an available prefix index number to this PWORD.
3. Update the BLG6CORQ panel to point to the index number you have assigned

The reports shown later in this section are standard Info/Man reports that provide the information needed to modify the BLG6CORQ Panel to support the interface.

A.1.1 PMF Report for BLG6CORQ Panel

This report is for the use of an Info/Man expert at your site when modifying the BLG6CORQ Panel. The second part of this report follows. The *index* value must be assigned by the Info/Man administrator.

PMF PANEL CONTENT REPORT										
DATE 11/11/96								PANELS: BLG6CORQ		
TIME 18:14:50								DATASET LABEL: RPANEL0		
PANEL NAME: BLG6CORQ										
<pre> +-----+] BLG6CORQ ENDEVOR/INTERFACE XREF FIELD RNCC/]] USE...THIS FIELD IS UPDATED BY THE ENDEVOR FOR OS/390 INTERFACE.]] FORM...AAAAAAAAAAAAAAAA 1 - 16 THE PACKAGE ID.]] THIS FIELD WILL BE USED TO CROSSREFERENCE A PACKAGE ID]] WITH A RNID.]] EXAMPLES: PKG00001.....REPLY.....PKG00001]] P000001200.....REPLY.....P000001200]] REPLY AS ILLUSTRATED] +-----+ </pre>										
ASSISTED ENTRY VALIDATION LINES										
PREFIX	PREFIX	DATA			PATTERN	AUTH				
INDEX	WORD	VALIDATION			TYPE	CODE				
----	----	-----			----	----				
index	RNCC/	CCV15				0000				
PANEL	FLOW	CONTROL	INFORMATION							
CREATE	INQUIRY	RETURN	DIALOG	FORCE	SRC	SWORD	STRUCTURED	COLLECT	PROGRAM	NULL REPLY
TARGET	TARGET	TO CALLER	END	END	END	INDEX	WORD	CALLER	EXIT	TARGET
-----	-----	-----	-----	-----	-----	-----	-----	-----	-----	-----
		YES	NO	NO		OCD4	XIM00NRC01	NO		

A.1.1.1 PMF Report for BLG6CORQ Panel page 2

This is the second part of the PMF report for the BLG6CORQ panel.

PMF PANEL CONTENT REPORT											PAGE 2			
DATE 11/11/96											PANELS: BLG6CORQ			
TIME 18:14:50											DATASET LABEL: RPANEL0			
RESPONSE PROCESSING CONTROL DATA														
CREATE	INQUIRY	REPLY	REPLACE	JOURNAL	JOURNAL	COGNIZE	'OR' OP	FORCE	COLLECT AS	DATA IS				
VALUE	VALUE	MEANING	PREVIOUS	REPLY	SEQ	RESPONSE	ALLOWED	SEARCH ANY	STRING DATA	A DATE				
45	45	WORDS	YES	NO	ORDER	YES	NO	NO	NO	NO				
COMMON CONTROL LINE INFORMATION														
HELP	CHANGE	NAME	PANELS	PANELS	PANEL	REPLY	DIVERT	SERVICE	COMM	DIALOG	IBM	PTF	APAR	DATE LAST
TARGET	ALLOWED	CHANGED	ADDED	DELETED	MODIFIED	IS DATA	ALLOWED	LEVEL	PANEL	SECTION	LEVEL	LEVEL	NUMBER	ALTERED
BLG0AAE0	NO	NO	NO	NO	YES	NO	YES		NO	00	04	00		10/21/92
TOTAL NUMBER OF PANELS PROCESSED: 1														

A.2 Customized Dictionary Entry

In addition to modifying the BLG6CORQ Panel, you also must add an entry to the dictionary data set specified in the BLGSESxx session parameters member.

The second report in this chapter is a PWORD XREF report, and contains the PWORD dictionary entry for the modified Assisted Entry panel.

This report is for the use of the Info/Man expert at a site that has already customized Info/Man. The report provides the Info/Man technician with the information needed to customize the appropriate dictionary entry.

A.2.1 The PWORD XREF Report

The PWORD XREF report is shown below. The *index* value must be assigned by the Info/Man administrator.

DATA SET LABEL: RPANEL0	PANEL RANGE: BLG6CORQ	PAGE 1
PREFIX PREFIX DATA		
INDEX VALIDATION]----- PANEL NAMES REFERENCING PREFIX WORD -----]	
<u>index</u> RNCC/ CCV15	BLG6CORQ	

A.3 Reserved Info/Man Fields

The fields summarized on the following pages are reserved by the interface for its use. Make sure that these fields are not being used to store any other data, because the interface overwrites any data it finds in these fields.

A.3.1 Inquiry PIDT Required Fields

Inquiry against Change records, for packages (IPKG).

This PIDT Field	Is reserved for this information
FIELD PANEL (BLG00000) INDEX(S0B06)	Change type record
FIELD PANEL (BLG6CORQ) INDEX(S0CD4)	Package ID
FIELD PANEL (BLG6PTYP) INDEX(S0C09)	Type: PACKAGE
FIELD PANEL (BLG6ACCN) INDEX(S0CCF)	RNID value

Inquiry against Activity records, for package actions (IPKGA).

This PIDT Field	Is reserved for this information
FIELD PANEL (BLG0F000) INDEX(S0B07)	Activity type record
FIELD PANEL (BLG6RNOR) INDEX(S0CD0)	Parent RNID
FIELD PANEL (BLG6ACNM) INDEX(S0CBC)	CCID
FIELD PANEL (BLG6PTYP) INDEX(S0C09)	Type: PKG/ACT
FIELD PANEL (BLG6ACCN) INDEX(S0CCF)	RNID value

Inquiry against Change records, for actions (IACT).

This PIDT Field	Is reserved for this information
FIELD PANEL (BLG00000) INDEX(S0B06)	Change type record
FIELD PANEL (BLG6PTYP) INDEX(S0C09)	Type: ACTION
FIELD PANEL (BLG6ACCN) INDEX(S0CCF)	RNID value

Inquiry against Problem records, for actions (IACT).

This PIDT Field	Is reserved for this information
FIELD PANEL(BLG00000) INDEX(S0032)	Problem type record
FIELD PANEL(BLG6PTYP) INDEX(S0C09)	Type: ACTION
FIELD PANEL(BLG6ACCN) INDEX(S0CCF)	RNID value

A.3.2 Retrieve PIDT Required Fields

Retrieve against Change records, for packages (RPKG).

This PIDT Field	Is reserved for this information
FIELD PANEL(BLG00000) INDEX(S0B06)	Change type record
FIELD PANEL(BLG6CORQ) INDEX(S0CD4)	Package ID
FIELD PANEL(BLG6PTYP) INDEX(S0C09)	Type: PACKAGE

Retrieve against Activity records, for package actions (RPKGA).

This PIDT Field	Is reserved for this information
FIELD PANEL(BLG1A131) INDEX(S0B07)	Activity type record
FIELD PANEL(BLG6RNOR) INDEX(S0CD0)	Parent RNID
FIELD PANEL (BLG6PTYP) INDEX(S0C09)	Type: PKG/ACT
FIELD PANEL(BLG6ACNM) INDEX(S0CBC)	CCID

Retrieve against Change records, for actions (RACT).

This PIDT Field	Is reserved for this information
FIELD PANEL(BLG00000) INDEX(S0B06)	Change type record
FIELD PANEL(BLG6PTYP) INDEX(S0C09)	Type: ACTION

Retrieve against Problem records, for actions (RACT).

This PIDT Field	Is reserved for this information
FIELD PANEL(BLG00000) INDEX(S0032)	Problem type record
FIELD PANEL(BLG6PTYP) INDEX(S0C09)	Type: ACTION

A.3.3 Create PIDT Required Fields

Create Change records for packages (CPKG).

This PIDT Field	Is reserved for this information
FIELD PANEL(BLG00000) INDEX(S0B06)	Change type record
FIELD PANEL(BLG6CORQ) INDEX(S0Cd4)	Package ID
FIELD PANEL(BLG6PTYP) INDEX(S0C09)	Type: PACKAGE
FIELD PANEL(BLG6URN0) INDEX(S0CCF)	RNID value

Create Activity records for package actions (CPKGA).

This PIDT Field	Is reserved for this information
FIELD PANEL(BLG1A131) INDEX(S0B07)	Activity type record
FIELD PANEL(BLG6RNOR) INDEX(S0CD0)	Parent RNID
FIELD PANEL(BLG6ACNM) INDEX(S0CBC)	CCID
FIELD PANEL (BLG6PTYP) INDEX(S0C09)	Type: PKG/ACT
FIELD PANEL(BLG6URN0) INDEX(S0CCF)	RNID value

Create Change records for actions (CACT).

This PIDT Field	Is reserved for this information
FIELD PANEL(BLG00000) INDEX(S0B06)	Change type record
FIELD PANEL(BLG6PTYP) INDEX(S0C09)	Type: package or action
FIELD PANEL(BLG6URN0) INDEX(S0CCF)	RNID value

Create Problem records for actions (CACT).

This PIDT Field	Is reserved for this information
FIELD PANEL(BLG00000) INDEX(S0032)	Change type record
FIELD PANEL(BLG6PTYP) INDEX(S0C09)	Type: package or action
FIELD PANEL(BLG6URN0) INDEX(S0CCF)	RNID value

A.3.4 Update PIDT Required Fields

Update Change records for packages (UPKG).

This PIDT Field	Is reserved for this information
FIELD PANEL(BLG00000) INDEX(S0B06)	Change type record

Update Activity records for package actions (UPKGA).

This PIDT Field	Is reserved for this information
FIELD PANEL(BLG1A131) INDEX(S0B07)	Activity type record
FIELD PANEL (BLG6PTYP) INDEX(S0C09)	Type: PKG/ACT

Update Change records for actions (UACT).

This PIDT Field	Is reserved for this information
FIELD PANEL(BLG00000) INDEX(S0B06)	Change type record

Update Problem records for actions (UACT).

This PIDT Field	Is reserved for this information
FIELD PANEL(BLG00000) INDEX(S0032)	Problem type record

Appendix B. Installing the Interface Under IIF

B.1 General Installation Information for IIF Sites

The target audience for this chapter are those people who have a strong understanding of Info/Man IIF records and a general knowledge of Endeavor.

B.1.1 Four Installation Phases

The installation of the interface falls into four phases:

- Phase 1: Installing basic Info/Man.
- Phase 2: Modifying basic Info/Man to support the interface.
- Phase 3: Tying the Endeavor and Info/Man sides together.
- Phase 4: Verifying the installation

B.1.2 Who Should Perform the Installation?

The best combination of people to install the interface are the site Info/Man administrator and the site Endeavor administrator working together.

B.2 Phase 1. Installing Basic Info/Man

The requirements in this section are basic Info/Man installation requirements that exist whether or not the interface is installed.

Note: The Endeavor--Info/Man interface requires that the IBM Information/Management product, version 4.2 or higher, is installed at your site.

Info/Man setup involves the following.

Step	Action
1	Define all interface users to a class in Info/Man.
2	Make sure the Information/Management LOAD LIBRARY data is in LINKLST, or that you have its data set name available.
3	Prepare a session parameter (BLGSESxx) module to reflect the database, data dictionary, read panel, and RFT data sets that the interface uses when accessing the data base via the API.
4	Make sure that BLGISPFD, an Info/Man supplied ISPF panel, has been moved into your ISPLIB concatenation.

These steps are discussed next.

B.2.1 Step 1. Define Users to Info/Man Classes

Each person authorized to use the interface must be defined to a class in Info/Man. See the Info/Man documentation for instructions.

B.2.2 Step 2. Prepare a Session Parameter Member

Info/Man requires a BLGSESxx module that points to the database, data dictionary, read panels, and RFT data set that the Interface uses when accessing the database via the API or native Info/Man.

This session parameter is well documented in the Info/Man install manuals. It is required of standard Info/Man. The naming standard is BLGSESxx where xx is the user's choice. This suffix is also required for an Endeavor start-up block assembly. Note that the same BLGSESxx member can be used for the interface as well as standard access to Info/Man.

The suffix used to name the actual BLGSESxx module is required in Phase 3. The rest of this install requires the data set names from the BLGSESxx module.

Note: The interface installation includes 15 modified panels, which must be copied into your read panel data set concatenation. You can either overlay the IBM version of this data set, or provide a new read panel data set ahead of the standard IBM data set. Either way, the result is that the RPANEL0 label points to the data set used in the install.

B.2.3 Step 3. Put Information/Management Load Data Set in LINKLIST

The Info/Man load data set is used to install the interface.

Sometimes this library is part of LINKLIST, and sometimes it is not. If its data set is not in LINKLIST, the person implementing the interface needs to know the name, and must put it into a STEPLIB in the concatenation.

B.2.4 Step 4. Move BLGISPF into ISPLIB

Make sure that BLGISPF, an Info/Man-supplied ISPF panel, has been moved into your ISPLIB concatenation. See your Info/Man administrator if you have any questions about this step.

B.2.5 Phase 1 Summary

In this phase of the implementation you have built a vanilla version of Info/Man IIF. At this point you should have installed Info/Man on your system, assigned users to an Info/Man class, and defined a BLGSESxx session parameter module. For future reference, record the following:

Class name:

BLGSESxx:

Within BLGSESxx, you have defined:

Dictionary DD name:

RFTDS DS name:

RPANEL0 name:

INFOMAN LOAD DS name:

B.3 Phase 2. Modifying Basic Info/Man for the Interface

In this phase, you customize the base Info/Man system set up in Phase 1 by performing the following steps

Step	Action	JCL used
1	Copy interface report shells to the RFT data set identified in Phase 1.	BC1JEI20
2	Copy the Endeavor versions of the Info/Man IIF panels into the RPANEL0 data set identified in Phase 1.	BC1JEX15
3	Install the dictionary updates.	BC1JEI30
4	Create the IIF PIDT and PIPT tables.	BC1JEX45
5	Build a rule set for driving the API using the interface batch utility.	BC1JEX00
6	Assemble or copy the BLGSESxx module into your STEPLIB data set.	BC1JEI05

These steps are discussed next.

B.3.1 Step 1. Copy Interface Report Shells into the RFT Data Set

Copy the interface report shells (RFTs) to your RFT data set. Do this using the BC1JEI20 member in iprfx.igual.JCLLIB. Before submitting this JCL, provide a job card, confirm that iprfx.igual and uprfx.uqual are the correct data set qualifiers, and confirm that tdisk is a valid unit name.

BC1JEI20 JCL is shown next.

```

/* ( COPY JOBCARD )
/******
/*
/* BC1JEI20 - JCL TO COPY THE ENDEVER INTERFACE REPORT SHELLS      *
/*          INTO THE USERS RFTDS DATASET.                          *
/*
/* USER NEEDS TO KNOW THE NAME OF THE RFTDS DATASET POINTED      *
/* TO IN THE BLGSESXX INFO/MAN PARM MEMBER.                        *
/*
/******
//STEP1 EXEC PGM=IEBCOPY
//SYSPRINT DD SYSOUT=*
//SYSUT3 DD UNIT=tdisk,SPACE=(TRK,(1))
//SYSUT4 DD UNIT=tdisk,SPACE=(TRK,(1))
//FROM DD DSN=iprfx.igual.SOURCE,DISP=SHR
//TO DD DSN=uprfx.uqual.SBLMFMT,DISP=SHR <==== RFTDS DATASET
//SYSIN DD *
COPY INDD=((FROM,R)),OUTDD=TO
SELECT MEMBER=(EINTACH0)
SELECT MEMBER=(EINTACH1)
SELECT MEMBER=(EINTAPB0)
SELECT MEMBER=(EINTAPB1)
SELECT MEMBER=(EINTPKG0)
SELECT MEMBER=(EINTPKG1)
SELECT MEMBER=(EINTPKG2)
/*

```

This job copies the following report shells.

This member	Is the RFT Shell for This Report
EINTACH0	Action Change Record Summary.
EINTACH1	Action Change Record Detail.
EINTAPB0	Action Problem Record Summary.
EINTAPB1	Action Problem Record Detail.
EINTPKG0	Package Parent Record Summary.
EINTPKG1	Package/Activity Record Summary.
EINTPKG2	Package/Activity Record Detail.

B.3.2 Step 2. Copy Modified Panels into the RPANEL0 Data Set

The interface requires that a modified version of the Assisted Entry Panel (BLG6CORQ) be available in the RPANEL0 data set pointed to by the BLGSESxx session parameter member. The modification allows the BLG6CORQ panel to store the 16-character Endeavor package ID. This field has been flagged as a reserved field for the interface.

The modified panel and other Endeavor IIF tailored panels are provided with the interface. Copy them into the RPANEL data set using the JCL in member BC1JEX15 in the iprfx.iqual.JCLLIB data set.

This job uses the BLGUT6F utility to copy a PDS member that has been created from a Info/Man panel using BLGUT6.

This panel must reside above the IBM panel of the same name. The person who runs the job to copy the modified panel needs only the data set name. You recorded the RPANEL0 data set name in the worksheet at the end of Phase 1.

BC1JEX15 JCL is shown next. Before submitting this JCL, provide a job card, and confirm that iprfx.iqual and uprfx.uqual are the correct data set qualifiers.

```
//(JOB CARD)
//*****
//*
//* USE THIS JOB TO INSTALL ALL THE MODIFIED PANELS AND SYSTEM TSPS
//* REQUIRED TO USE THE API WITH IIF RECORD FORMATS.
//*
//*
//*
//* JOB TO UPLOAD MEMBERS OF A PDS THAT HAVE BEEN CREATED FROM THE
//* BLGUT6F DOWN UTILITY PROGRAM THAT CONVERTS VSAM PANELS INTO
//* MEMBERS IN A PDS FOR DISTRIBUTION. (LRECL=80. UNUSABLE FORMAT)
//*
//* NOTE STEPLIB NOT NECESSARY IF INFO/MAN DATASET ARE IN LINKLST
//*
//*****
//BLGUT6 EXEC PGM=BLGUT6,REGION=2048K
//STEPLIB DD DSN=UPRFX.UQUAL.LOAD,DISP=SHR <== INFO/MAN LOADLIB
//SYSPRINT DD SYSOUT=*
//BLGPDS DD DISP=SHR,DSN=IPRFX.IQUAL.SOURCE
//BLGPNLS DD DISP=SHR,DSN=UPRFX.UQUAL.ENDPNLS <== RPAPELO
//SYSIN DD *
INCLUDE
BLGAPI10
BLGAPI90
BLGAPI92
BLGAPI95
BLGAPI99
BLGZAACP
BLGZAAUP
BLG6CORQ
BTN6PTY1
BTN0C100
BTN0C101
BTN0S020
BTN0B100
BTN0M100
BTN0L100
/*
```

B.3.3 Step 3. Copy Dictionary Update

The modified Assisted Entry Panel (BLG6CORQ) has an associated dictionary entry. This entry must be copied into the dictionary identified in the Phase 1 Summary. Run JCL job BC1JEI30 to install this dictionary entry.

CAUTION:

Run this job only if you are installing basic Info/Man. If your site has customized Info/Man, refer to Appendix A, "Information for Advanced Users," for instructions on updating the dictionary entry.

BC1JEI30 JCL is shown next. Before submitting this JCL, provide a job card, and confirm that iprfx.igual and uprfx.uqual are the correct data set qualifier.

```

/*  JOBCARD
//*****
//*  COPY PDS MEMBER TO SEQUENTIAL DSN  *
//*****
//COPYMEM EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DSN=iprfx.igual.SOURCE(INFODICT),DISP=SHR
//SYSUT2 DD DSN=&&TEMPDICT,DISP=(,PASS),
//      SPACE=(TRK,(1,2)),
//      DCB=(RECFM=VBA,LRECL=137,BLKSIZE=1370,DSORG=PS)
//SYSIN DD DUMMY
//*****
//*  LOAD INFO DICTIONARY
//*****
//LOADMEM EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//COPY DD DSN=&&TEMPDICT,DISP=(OLD,DELETE)
//SYSIN DD *
REPRO OUTDATASET(uprfx.uqual.DICT) +
INFILE(COPY) +
REPLACE
/*

```

B.3.4 Step 4. Create Computer Associates PIDT and PIPT Tables

Create the Computer Associates PIDT and PIPT tables by executing the InfoMan BLGUT8 utility program using the CA supplied source PIDT tables that reside in iprfx.igual.SOURCE. This step is necessary to pick up the updated data dictionary entry. The tables are required by the interface to support batch and real-time execution, CCID list support and display services.

Use the BC1JEX45 member in iprfx.igual.JCLLIB. This JCL is shown next. Before submitting the JCL, provide a job card and ensure that iprfx.igual and uprfx.uqual are the correct data set qualifiers.

```

/* ( COPY JOBCARD )
//STEP1 EXEC PGM=BLGUT8,REGION=2048K
//STEPLIB DD DSN=UPRFX.UQUAL.LOAD,DISP=SHR <== INFO/MAN LOAD LIB
//BLGPNLS DD DSN=IPRFX.IQUAL.SOURCE(BLG6CORQ),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(BTN6PTY1),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(BTN0C100),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(BTN0C101),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(BTN0S020),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(BTN0B100),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(BTN0M100),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(BTN0L100),DISP=SHR
// DD DSN=UPRFX.UQUAL.SBLMPNLS,DISP=SHR <== INFO/MAN PANELS
//BLGDICT DD DSN=UPRFX.UQUAL.DICT,DISP=SHR <== DICT DATA SET
//BLGRFT DD DSN=UPRFX.UQUAL.SBLMFM,DISP=SHR <== RFTDS
//SYSPRINT DD SYSOUT=*
//SYSIN DD DSN=IPRFX.IQUAL.SOURCE(TS0B06AS),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0B06CS),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0B06IS),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0B06RS),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0B06US),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0B07CS),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0B07IS),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0B07RS),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0B07US),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0032CS),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0032IS),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0032RS),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0032US),DISP=SHR

```

B.3.5 Step 5. Build a Rule Set

In addition to setting up Info/Man to support the interface, you need to build a rule set to drive the Info/Man API. You do this using the batch utility provided by the interface.

Before proceeding, review:

- Chapter 3, “Setting Up the Endeavor/INFO Table — the Batch Utility” on page 3-1 of this manual.
- The samples provided in iprfx.igual.SOURCE. There are three samples provided:

Sample	Description
EISCRXX1	Batch utility input for stand-alone actions, using IIF Info/Man Change records.
EISCRXX2	Batch utility input for stand-alone actions, using IIF Info/Man Problem records.
EISCRXX3	Batch utility input for package functions and package actions under IIF Info/Man.

Next, create a rule set. It is advisable to keep it simple at first. Do this by taking one or two actions or package functions and writing utility syntax input for them. Then

run the batch utility to create an initial rule set. The JCL for executing the utility to build the rule set is in member BC1JEX00 of iprfx.igual.JCLLIB.

A successful run of the utility produces a load module called by a name you specify. This load module name is used in Phase 3. Assemble this load module into a STEPLIB data set.

Remember:

- To change what you use the interface for is a simple run of the utility. It requires no further Info/Man or Endeavor work to enhance the rule set.
- If you point to any PIDT tables other than the defaults, verify that they reside in the RFTDS data set pointed to in your BLGSESxx member.

The JCL for executing the utility to build the rule set is in member BC1JEX00 of iprfx.igual.JCLLIB. See B.3.5.1, “BC1JEX00 JCL.” Before submitting this JCL, provide a job card, confirm that iprfx.igual and uprfx.uqual are the correct data set qualifiers, confirm that tdisk is a valid unit name and, if necessary, change the size of the WORK1 or WORK2 data sets.

B.3.5.1 BC1JEX00 JCL

```

/*(JOB CARD)
/*-----*
/*
/* (C) 2002 COMPUTER ASSOCIATES INTERNATIONAL, INC.
/*
/* NAME: BC1JEX00
/*
/* THE PURPOSE OF THIS JOB TO RUN THE BATCH UTILITY.
/*
/*
/**** NOTE THIS VERSION OF THE BATCH UTILITY JCL IS FOR THE IIF
/**** RECORD SUPPORT. THE SCRIPT DEFAULT INPUT AND THE SOURCE
/**** PIDT DATASETS ARE DIFFERENT THAN THE STANDARD PROBLEM AND
/**** CHANGE RECORDS.
/*
/* INPUT TO THIS JOB IS THE BATCH UTILITY SYNTAX. OUTPUT FROM
/* THIS IS THE E-PIDT TABLE. THIS TABLE NAME MUST BE INCLUDE IN
/* THE ASSEMBLY OF THE BC1TEI90 STARTUP BLOCK.
/*
/* NOTE: NDVRIN POINTS TO THE BATCH SYNTAX THE CUSTOMER WANTS TO
/* USE. THERE ARE THREE SAMPLE SCRIPTS DELIVERED WITH THE
/* PRODUCT.
/*
/* EISCRXX1 ==> USE FOR STAND ALONE ACTIONS - CHANGE RECORDS
/* EISCRXX2 ==> USE FOR STAND ALONE ACTIONS - PROBLEM RECORDS
/* EISCRXX3 ==> USE FOR PACKAGE LEVEL INTERFACE
/*

```

B.3 Phase 2. Modifying Basic Info/Man for the Interface

```
/* REPLACE THE NDVRIN MEMBER NAME IMBMR WITH THE MEMBER YOU CHOSE.*
/*-----*
/******
/* RUN UTILITY TO ASSEMBLE SYNTAX THAT CREATES E-PIDT LOADMOD *
/******
//BATCH EXEC PGM=NDVRC1,PARM=C1BMEI00
//STEPLIB DD DISP=SHR,DSN=UPRFX.UQUAL.AUTHLIB
// DD DISP=SHR,DSN=IPRFX.IQUAL.AUTHLIB
//CONLIB DD DISP=SHR,DSN=IPRFX.IQUAL.CONLIB
//NDVRIN DD DISP=SHR,DSN=IPRFX.IQUAL.SOURCE(IMBMR) <=== SEE NOTES
/*
/******
/* THE IPRFX.IQUAL.SOURCE DATASET IS PROVIDED WITH ENDEVOR *
/* *
/* THESE POINT TO THE ENDEVOR-PROVIDED PIDT SOURCE FOR IIF *
/* RECORDS. IF YOU ARE USING *
/* PROBLEM LEVEL INTERFACE, USE THE LINES THAT POINT TO THE *
/* MEMBERS THAT START WITH 'TS0032' OTHERWISE USE THE LINES *
/* THAT POINT TO THE MEMBERS THAT START WITH 'TS0B06' AND *
/* 'TS0B07'. *
/******
//NDVRPIDT DD DSN=IPRFX.IQUAL.SOURCE(TS0B06IS),DISP=SHR <= CHANGE
// DD DSN=IPRFX.IQUAL.SOURCE(TS0B06RS),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0B06CS),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0B06US),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0B06AS),DISP=SHR
/*
// DD DSN=IPRFX.IQUAL.SOURCE(TS0B07IS),DISP=SHR <= ACTIVITY
// DD DSN=IPRFX.IQUAL.SOURCE(TS0B07RS),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0B07CS),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0B07US),DISP=SHR
/*
// DD DSN=IPRFX.IQUAL.SOURCE(TS0032US),DISP=SHR <= PROBLEM
// DD DSN=IPRFX.IQUAL.SOURCE(TS0032CS),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0032IS),DISP=SHR
// DD DSN=IPRFX.IQUAL.SOURCE(TS0032RS),DISP=SHR
/*
//NDVRRPT1 DD SYSOUT=*
//NDVRRPT2 DD SYSOUT=*
//NDVRRPT3 DD SYSOUT=*
/*
/*SAMPLE JCL FOR BC1JEX00 NDVRWK1 AND NDVRWK2 SHOULD BE AS FOLLOWS
/*
//NDVRWK1 DD DSN=UPRFX.UQUAL.WORK1,DISP=(,PASS),
// UNIT=TDISK,SPACE=(TRK,(10))
//NDVRWK2 DD DSN=UPRFX.UQUAL.WORK2,DISP=(,PASS),
// UNIT=TDISK,SPACE=(TRK,(10))
/*
//SYSMOD DD DISP=SHR,DSN=UPRFX.UQUAL.AUTHLIB <== E-PIDT STORED HERE
//SYSUT1 DD UNIT=TDISK,SPACE=(TRK,(1,1))
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
```

B.3.6 Step 6. Copy the BLGSESxx Module to STEPLIB

Assemble or copy the BLGSESxx member into your STEPLIB data set. See iprfx.igual.JCLLIB(BC1JEI05) for sample JCL.

BC1JEI05 JCL is shown next. Before submitting this JCL, provide a job card, confirm that uprfx.igual are the correct data set qualifiers, confirm that tdisk is a valid unit name, and replace xx in the string BLGSESxx with a session identifier.

```

/* ( COPY JOBCARD )
/*****
/*
/* BC1JEI05 - BUILD INFORMATION/MANAGEMENT SESSION TABLE.      *
/*          THIS JOB IS DESCRIBED IN IBM'S PLANNING AND          *
/*          INSTALLING THE INFORMATION/FAMILY MANUAL.           *
/*
/* STEP1 WILL ASSEMBLE THE MEMBER SPECIFIED INLINE              *
/*
/* STEP2 WILL LINKEDIT THE MEMBER AND STORE IN USER LOADLIB     *
/*
/*****
//STEP1 EXEC PGM=ASMA90,PARM='NODECK,OBJECT,NOTERM'
//SYSLIB DD DISP=SHR,DSN=uprfx.igual.SBLMMACS
//          DD DISP=SHR,DSN=SYS1.MACLIB
//SYSLIN DD DISP=(NEW,PASS,DELETE),DSN=&&SYSLIN,
//          UNIT=tdisk,SPACE=(TRK,(3,5),RLSE),
//          DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//SYSPUNCH DD DUMMY
//SYSUT1 DD UNIT=tdisk,SPACE=(CYL,(5,3))
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
BLGVDSN TITLE 'BLGVDSN - INFORMATION/MANAGEMENT SESSION PARAMETERS'
CSECT
BLGPARM DICTDS=DICTDS, X
        RFTDS=RFTDS, X
        RPANLDS=(RPANEL0,RPANEL1)
SYSTEM BLGCLUST NAME=5, X
        PRODUCT=MGMT, X
        SDDS=MGTSDDS, X
        SDIDS=MGTSDDIDS, X
        SDLDS=MGTSDDLDS
MGTSDDS BLGCLDSN DSN=uprfx.igual.SDDS
MGTSDDIDS BLGCLDSN DSN=uprfx.igual.SDIDS
MGTSDDLDS BLGCLDSN DSN=uprfx.igual.SDDLDS
RPANEL0 BLGCLDSN DSN=uprfx.igual.ENDPNLS <== CA-Endevor PANEL
RPANEL1 BLGCLDSN DSN=uprfx.igual.IBMPNLS <== IBM PANELS
RFTDS BLGCLDSN DSN=uprfx.igual.SBLMFMT,FILE=RFTDD
DICTDS BLGCLDSN DSN=uprfx.igual.DICT
BLGGEN ,
END

```

```
/*
/**
//STEP2 EXEC PGM=IEWL,PARM='LIST,NCAL,XREF,SIZE=(256K,64K)',
// COND=(0,NE)
//SYSLIN DD DISP=(OLD,DELETE),DSN=&&SYSLIN
// DD *
// ENTRY BLGVDSN
// NAME BLGSESxx
//SYSLMOD DD DISP=SHR,DSN=uprfx.uqua1.LOADLIB
//SYSUT1 DD UNIT=tisk,SPACE=(CYL,(5,3))
//SYSPRINT DD SYSOUT=*
```

B.3.7 Summary

the pieces are now in place for the interface on both the Info/Man and the Endeavor sides. In Phase 3, these pieces are connected to enable the interface.

B.4 Phase 3. Connecting the Interface

You must complete the tasks described for Phase 1 and Phase 2 before starting the tasks described in this section.

This phase of the installation process involves setting up the Info/Man and Endeavor sides of the interface to work together, as described next.

Step	Action
1	Modify the @EINFO macro.
2	Assemble and link BC1TEI90 into a LINKLIST or authorized library.
3	Assemble C1DEFLT5 with Info/Man password.
4	Make sure that required panels are in the ISPPLIB data set.
5	Make sure that members INFO01 and PKMR02 are in your ISPMLIB data set.
6	Edit C1SB3000 skeleton JCL.

These steps are described in the following sections.

B.4.1 Step 1. Modify the @EINFO Macro

Various parts of the interface access the user-assembled module BC1TEI90 to obtain user parameter information. BC1TEI90 is a load module used during exits 5, 2, 3, and 7 for start-up parameters, during display services, and during batch execution. The user must assemble this load module. Once assembled, this module should be linked into a LINKLIST or authorized library.

The @EINFO macro, supplied with the interface, provides input to BC1TEI90. The macro can be found embedded in member BC1JEI10 in iprfx.igual.JCLLIB. Edit this macro based on the information contained in the Phase 1 checklist and your site standards. The @EINFO macro is shown next.

```
@EINFO ENTRY=START,
      BLGSESS=00,
      APISESID=USERID,
      EPIDT=ENDEVOR,
      INVCLASS=MASTER,
      WAITTIME=300,
      RECTYPE=CHANGE,<tab>
@EINFO ENTRY=END
```

The entries in this macro are described next.

Parameter	Description
ENTRY=START	Indicates the beginning of the BC1TEI90 table.
BLGSES=	<p>Info/Man looks for session parameters in a module named BLGSESxx, where xx is a two-character identifier for the module. This module is described in the Info/Man documentation. You recorded the components of this module during Phase 1 of this installation, as well as the two-character identifier for the module. When modifying the @EINFO macro supplied with the interface, replace the default value 00 with the value you specified in Phase 1.</p> <p>The default is 00. This is the same as standard Info/Man.</p>
APISESID=	<p>The API requires a session ID during initialization. This session can be called USERID or can be a predefined Info/Man API ID.</p> <p>The default is USERID. Leaving this default allows the same start-up module to be used by many people, with the interface substituting the user ID of the caller for the literal USERID.</p> <p>If you include this keyword, you must provide a value.</p>
EPIDT=	<p>This keyword contains the name of the assembled Endeavor/INFO table produced by the batch utility. The default is Endeavor.</p> <p>If you include this keyword, you must provide a value.</p>
INVCLASS=	<p>Info/Man has its own security system. It is broken into classes with privileges. Every user or API ID is assigned to at least one CLASS. The API requires a class name.</p> <p>The default is MASTER.</p>
WAITTIME=	The API allows the application to define how long it will wait for a response from the Info/Man API server. If the user does not use this keyword, the default will be 300 seconds.
RECTYPE=	<p>Users must select one kind of Info/Man record for recording information about stand-alone actions. They can use either Change or Problem records. Display services also need this information.</p> <p>The default is CHANGE. If coded, it cannot be blank.</p>
ENTRY=END	Required. This statement denotes the end of the BC1TEI90 table.

B.4.2 Step 2. Assemble and Link BC1TEI90

Assemble and link BC1TEI90 using the JCL in member BC1JEI10 in iprfx.igual.JCLLIB. BC1JEI10 JCL is shown next. Before submitting this JCL, provide a job card, confirm that iprfx.igual and uprfx.igual are the correct data set qualifiers, and that tdisk is a unit name.

```

/* ( COPY JOBCARD )
/******
/*
/* BC1JEI10 - THIS JOB IS USED TO CUSTOMIZE AND BUILD THE
/* ENDEAVOR FOR OS/390 INFORMATION/MANAGEMENT INTERFACE
/* STARTUP BLOCK CALLED BC1TEI90
/*
/* THE FOLLOWING UPDATES MUST BE MADE TO THIS JCL BEFORE
/* IT CAN BE EXECUTED:
/*
/* 1. UPDATE THE JOBCARD TO REFLECT CORRECT SITE INFORMATION
/* 2. REVIEW THAT THE SOURCE AND CONLIB DATA SET NAMES ARE
/* CORRECT.
/* - IPRFX.IGUAL.SOURCE
/* - UPRFX.UQUAL.LOADLIB
/*
/******
//STEP1 EXEC PGM=ASMA90,PARM='NODECK,OBJECT,NOTERM'
//SYSLIB DD DISP=SHR,DSN=IPRFX.IGUAL.SOURCE
// DD DISP=SHR,DSN=SYS1.MACLIB
//SYSLIN DD DISP=(NEW,PASS,DELETE),DSN=&&SYSLIN,
// UNIT=TDISK,SPACE=(TRK,(3,5),RLSE),
// DCB=(RECFM=FB,LRECL=80,BLKSIZE=3120)
//SYSPUNCH DD DUMMY
//SYSUT1 DD UNIT=TDISK,SPACE=(CYL,(5,3))
//SYSPRINT DD SYSOUT=*
//SYSIN DD *
*****
* THE PURPOSE OF THIS JOB IS TO ASSEMBLE THE BC1TEI90 START
* UP LOAD MODULE
*
* THIS JOB SHOULD BE RUN AFTER THE E-PIDT HAS BEEN CREATED
* AND AFTER AN INFO/MAN SESSION PARAMETER BLOCK HAS BEEN
* ESTABLISHED. BOTH OF THOSE VALUES ARE REQUIRED FOR THIS
* ASSEMBLY.
*
*
*
*****
BC1TEI90 TITLE 'ENDEAVOR/INFO STARTUP BLOCK'
*****
* THIS IS A SAMPLE ENDEAVOR/INFO STARTUP BLOCK. DURING EXIT 5 PROCESS-
* ING THIS BLOCK WILL BE PROVIDE THE INTERFACE THE REQUIRED PARMS
* AND SESSION ID TO INITIALIZE THE INTERFACE AND THE INFO/MAN
* API SERVER.
*****

```

B.4 Phase 3. Connecting the Interface

```
*****
*
*   THERE ARE FIVE MAIN PIECES OF INPUT.
*
*           BLGSESS=00      <=== STANDARD INFO/MAN SESSION
*                               SUFFIX. DEFAULT IS 00.
*                               (BLGSE00)
*
*           APISEID=USERID  <=== THE 4.2 API WILL ALLOW USERS
*                               TO HAVE SESSION IDS.
*                               IF THIS FIELD IS NOT MODIFIED
*                               THEN THE REAL TIME USERID WILL
*                               BE SUBSTITUTED AT EXEC TIME.
*
*           ENDVPIDT=ENDEVOR <=== THE NAME OF THE LOAD MODULE
*                               THAT WAS CREATED BY THE BATCH
*                               UTILITY. THIS IS THE TABLE
*                               NAME. DEFAULT IS ENDEVOR.
*
*           INVCLASS=MASTER <=== INFO/MAN HAS ITS OWN SECURITY
*                               SYSTEM. THROUGH THE USE OF
*                               CLASSES, USERIDS ARE REGISTERED*
*                               IN ONE OR MORE CLASSES. EACH
*                               CLASS HAS A SET OF PRIVLEDGES.
*                               DEFAULT IS MASTER.
*
*           WAITTIME=300    <=== THE MAX TIME THE INTERFACE
*                               WILL WAIT FOR A REPSONSE BACK
*                               FROM THE INFO/MAN SERVER.
*                               DEFAULT IS 300 SECS.
*
*****
      @EINFO ENTRY=START,
      BLGSESS=00,
      APISEID=USERID,
      ENDVPIDT=ENDEVOR,
      INVCLASS=MASTER,
      WAITTIME=300,
      RECTYPE=CHANGE
      @EINFO ENTRY=END
/*
/******
/* THE INPUT SHOULD BE ASSEMBLED AND LINKED INTO A STEPLIB DATASET. *
/* PLEASE REFER TO THE INTERFACE CHAPTER COVERING IMPLEMENTATION AND*
/* INSTALLATION.
/*
/******
/*
//STEP2 EXEC PGM=IEWL,PARM='LIST,NCAL,RENT,XREF,SIZE=(256K,64K)',
// COND=(0,NE)
//SYSLIN DD DISP=(OLD,DELETE),DSN=&&SYSLIN
//SYSLMOD DD DISP=SHR,DSN=uprfx.uqua1.LOADLIB(BC1TEI90)
//SYSUT1 DD UNIT=tisk,SPACE=(CYL,(5,3))
//SYSPRINT DD SYSOUT=*
```

B.4.3 Step 3. Reassemble C1DEFLT5 with the Info/Man Password

Include the Info/Man password in your C1DEFLT5 table. The password parameter is in the TYPE=MAIN section of the defaults table, as shown next.

```

C1DEFLT5 TYPE=MAIN,
  ACCSTBL=, ACCESS SECURITY TABLE X
  ACMIDXUP=N, CROSS-REFERENCE DATA UPDATE X
  ACMROOT=, ROOT DATABASE X
  ACMXREF=, XREF DATABASE X
  APRVFLG=N, APPROVAL PROCESSING (Y/N) X
  ASCM=N, ASCM CONTROL OPTION X
  BATCHID=0, BATCH UID FROM JOBNAME/USER= X
  CIPODSN=, CCID VALIDATION DSN X
  CSP=N, CSP CONTROL OPTION X
  CUNAME='*** PUT YOUR COMPANY NAME HERE ***', (50 CHAR) X
  DB2=N, DB2 CONTROL OPTION X
  ELINK=N, CA-Endevor/LINK CONTROL OPTION X
  ESSI=N, ESSI CONTROL OPTION X
  INFO=N, INFOMAN CONTROL OPTION X
  LIBENV=, LIBRARIAN (LB), PANVALET (PV) X
  LIBENVP=N, LIBRARIAN/PANVALET OPTION X
  LIBRPGM=, LIBRARIAN BATCH PROGRAM NAME X
  LINESPP=60, LINES PER PAGE X
  MACDSN='IPRFX.IQUAL.SOURCE', E/OS/390 SOURCE LIBRARY X
  PKGDSN='UPRFX.UQUAL.PACKAGE', PACKAGE DATASET NAME X
  PKGTSO=N, FOREGROUND PACKAGE EXEC (Y//N) X
  PDM=N, PDM CONTROL OPTION X
  PKGSEC=, PACKAGE SECURITY X
  PKGCSEC=N, PACKAGE CAST SECURITY (Y/N) X
  PKGCVAL=0, PKG COMPONENT VALIDATION (Y/O) X
  PROC=N, PROCESSOR OPTION X
  RACFGRP=, E/OS/390 RACF GROUP NAME X
  RACFPWD=, E/OS/390 RACF OPTION X
  RACFUID=, E/OS/390 RACF USERID X
  SITEID=0, E/OS/390 SITE ID X
  SMFREC#=0, SMF RECORD NUMBER X
  SPFEDIT=SPFEDIT, DEFAULT PDS RESERVE X
  SYSIEWL=SYSIEWLP, DEFAULT PDS/LINK EDIT RESERVE X
  UIDLOC=(1,7), UID/JOBNAME START/LENGTH POS X
  VIUNIT=TDISK, UNIT FOR VIO-ELIGIBLE ALLOC X
  WRKUNIT=TDISK, UNIT NAME FOR WORK SPACE X
  WORKVOL= VOL SER NUMBER FOR WRKUNIT

```

Reassemble the C1DEFLT5 table with the JCL in member BC1JDEFT in iprfx.igual.JCLLIB.

B.4.4 Step 4. Make Sure Required Panels Are in ISPPLIB

Make sure that the panels listed next reside in your ISPPLIB. These panels can be found in `iprfx.igual.ISPPLIB`.

Panel ID	Description
C1EILIST, CITILIST	CCID list panel and its associated tutorial panel.
C1SEIBRW, CITEIBRW	Info/Man record display for packages and stand-alone actions and its associated tutorial panel.
C1SP1000, C1SP2000, C1SP3000, C1SP4000, C1SP5000, C1SP6000, C1SP7000	Package panels with capability to display correlation data.
CITP1000, CITP2000, CITP3000, CITP4000, CITP5000, CITP6000, CITP7000	Tutorials for package panels with capability to display correlation data.

B.4.5 Step 5. Make Sure INFO01, PKEX21, and PKMR02 Are in ISPMLIB

Members INFO01, PKEX21, and PKMR02 contain interface messages, and must reside in your ISPMLIB. These members can be found in `iprfx.igual.ISPMLIB`.

B.4.6 Step 6. Edit C1SB3000 Skeleton JCL

Member C1SB3000 is used to submit Endeavor package and batch processing requests. The member can be found in iprfx.igual.ISPSLIB. Before invoking this JCL, confirm that iprfx.igual are the correct data set qualifiers, and that tdisk is a valid unit name.

```

)CM THIS SKELETON IS USED TO GENERATE ENDEVOR FOR OS/390 JCL FOR BATCH.
)SEL &C1BJC1 -= &Z
&C1BJC1
)ENDSEL
)SEL &C1BJC2 -= &Z
&C1BJC2
)ENDSEL
)SEL &C1BJC3 -= &Z
&C1BJC3
)ENDSEL
)SEL &C1BJC4 -= &Z
&C1BJC4
)ENDSEL
/*
//*****
//* CA-Endevor JCL STATEMENTS *
//*****
//NDVRBAT EXEC PGM=NDVRC1,DYNAMNBR=1500,REGION=4096K,
)SEL &C1BLDPKG -= P
// PARM='C1BM3000'
)ENDSEL
)ENDSEL
)SEL &C1BLDPKG = P
// PARM='C1BM3000,,&VPHPKGID'
)ENDSEL
//CONLIB DD DSN=iprfx.igual.CONLIB,DISP=SHR
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//*****
//* SORT WORK FILES *
//*****
//SORTWK01 DD UNIT=tdisk,SPACE=(CYL,(2,1))
//SORTWK02 DD UNIT=tdisk,SPACE=(CYL,(2,1))
//SORTWK03 DD UNIT=tdisk,SPACE=(CYL,(2,1))
//SORTWK04 DD UNIT=tdisk,SPACE=(CYL,(2,1))
)SEL &VARSICSP = Y

```

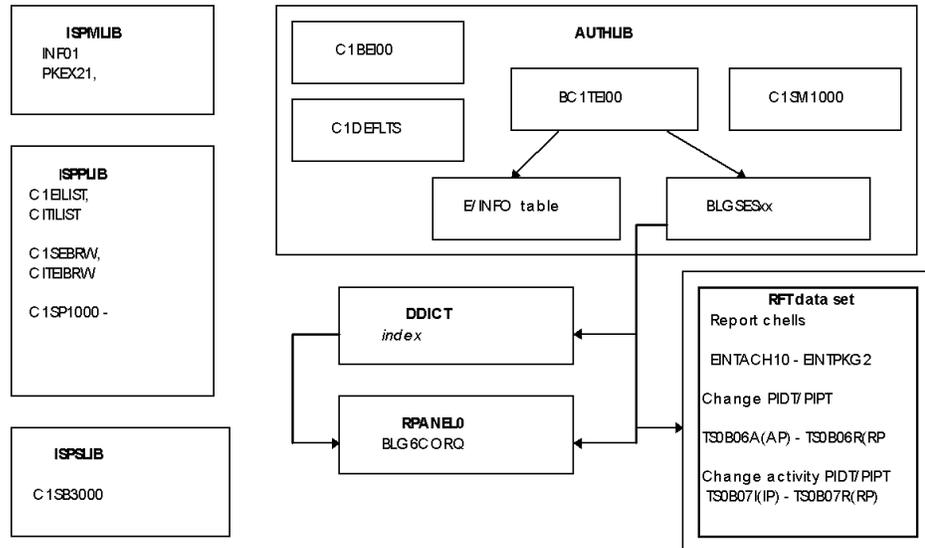
B.5 Phase 4. Verifying the Installation

The list next shows the members that should reside in the designated data sets for the interface to function as designed. If any of them are not there, the interface will not work as designed.

This data set	Must contain these members or information
AUTHLIB	<p>Load modules, as well as the following:</p> <ul style="list-style-type: none"> ■ C1BMEI00 ■ C1DEFLTS table. See 2.4.3, “Step 3. Reassemble C1DEFLTS with the Info/Man Password” on page 2-19. ■ C1SM1000 ■ BC1TEI90, the module containing user parameters, which you assembled in Phase 3, Step 1. See 2.4.1, “Step 1. Modify the @EINFO Macro” on page 2-14. ■ Endeavor/INFO table. This is the rule set that you build using the batch utility. See 2.3.5, “Step 5. Build a Rule Set” on page 2-9. ■ BLGSESxx, the module containing the Info/Man session parameters. See 2.2.2, “Step 2. Prepare a Session Parameter Member” on page 2-3.
CONLIB	<p>The Endeavor load modules installed from iprfx.igual.CONLIB. Make sure to save the current copy of C1BEXITS before installing the new interface. This allows you to go back to the old interface during testing.</p>
RFTDS	<p>EINTACH0, EINTACH1, EINTAPB0, EINTAPB1. EINTPKG0, EINTPKG1, EINTPKG2. PIDT and PIPT tables. See the list in the next section. These panels are found in iprfx.igual.SOURCE. BLGISPF, the Info/Man supplied ISPF panel.</p>

This data set	Must contain these members or information
RPANEL0	<p>BLG6CORQ, the modified version of the Assisted Entry panel.</p> <p>BLGAPI10, BLGAPI90, BLGAPI92, BLGAPI95, BLGAPI99.</p> <p>BLGZAACP, BLGAAUP.</p> <p>BTN0B100, BTN0C100, BTN0C101, BTN0L100, BTN0M100, BTN0S020, BTN6PTY1.</p> <p>See 2.3.3, “Step 3. Copy Dictionary Update” on page 2-7.</p>
DDICT	<p><i>index</i> RNCC/CCV15. This is the modified dictionary entry associated with the modified Assisted Entry Panel. See 2.3.4, “Step 4. Create the Standard IBM PIDT and PIPT Tables” on page 2-8.</p>
iprfx.igual.ISPPLIB	<p>C1EILIST, C1SEIBRW, C1SP1000 C1SP7000 and the associated tutorials (CITILIST, CITEIBRW, and C1TP1000-C1TP7000.</p>
iprfx.igual.ISPMLIB	<p>INFO01, PKEX21, PKMR02.</p>
iprfx.igual.ISPSLIB	<p>C1SB3000.</p>

The graphic displayed next shows how these modules tie the interface together and indicates the important members that should be at each location.



B.5.1 Required PIDT and PIPT Tables

For the interface to work correctly the following PIDT and PIPT tables must be in the RFT data set pointed to by the BLGSESxx module.

There are three categories of PIDT/PIPT tables: Change, Change Activity, and Problem.

B.5.1.1 Change Record PIDT and PIPT Tables

The following PIDT/PIPT tables related to Change records must be installed.

Table identifier	Description
TS0B06A	Change record, Add, PIDT.
TS0B06AP	Change record, Add, PIPT.
TS0B06I	Change record, Inquiry, PIDT.
TS0B06IP	Change record, Inquiry, PIPT.
TS0B06C	Change record, Create, PIDT.
TS0B06CP	Change record, Create, PIPT.
TS0B06U	Change record, Update, PIDT.
TS0B06UP	Change record, Update, PIPT.
TS0B06R	Change record, Retrieve, PIDT.
TS0B06RP	Change record, Retrieve, PIPT.

B.5.1.2 Change Activity PIDT and PIPT Tables

The following PIDT/PIPT tables related to Change Activity records must be installed.

Table identifier	Description
TS0B07I	Change activity record, Inquiry, PIDT.
TS0B07IP	Change activity record, Inquiry, PIPT.
TS0B07C	Change activity record, Create, PIDT.
TS0B07CP	Change activity record, Create, PIPT.
TS0B07U	Change activity record, Update, PIDT.
TS0B07UP	Change activity record, Update, PIPT.
TS0B07R	Change activity record, Retrieve, PIDT.
TS0B07RP	Change activity record, Retrieve, PIPT.

B.5.1.3 Problem Record PIDT and PIPT Tables

The following PIDT/PIPT tables related to Problem records must be installed.

Table identifier	Description
TS0032I	Problem record, Inquiry, PIDT.
TS0032IP	Problem record, Inquiry, PIPT.
TS0032C	Problem record, Create, PIDT.
TS0032CP	Problem record, Create, PIPT.
TS0032U	Problem record, Update, PIDT.
TS0032UP	Problem record, Update, PIPT.
TS0032R	Problem record, Retrieve, PIDT.
TS0032RP	Problem record, Retrieve, PIPT.