AllFusion™ Endevor® Change Manager

Getting Started

4.0

Computer Associates™

SP1

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.

© 2003 Computer Associates International, Inc.

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

Contents

Chapter 1: Introducing AllFusion Endevor Change Manager

Welcome! 1-1
Extensive Business Benefits 1-2
Distinctive Features 1-3
Change and Configuration Management Across the Enterprise 1-5
Documentation and Support 1-7

Chapter 2: A Conceptual View of Endevor

Introduction 2-1
Inventory Management 2-2
    Automated Inventory Manager 2-3
Change Control 2-7
    Automated Change Control 2-7
    Endevor Actions 2-14
    Endevor's Software Control Language (SCL) 2-15
Configuration Management 2-16
    AllFusion Endevor Change Manager Automated Configuration Option 2-17
Component Validation 2-21
Release Management 2-22
    Automated Release Management 2-22

Supporting Functionality 2-28
AllFusion Endevor Change Manager Parallel Development Option 2-29
Software Security Management 2-31
Software Information Management 2-33
Endevor Benefit Summary 2-35

Chapter 3: Implementing the Endevor Sample Application

Implement Endevor 3-1
Naming Conventions Used in the Sample Application 3-3
Editing the Endevor Defaults Table (C1DEFLTS) 3-4
The TYPE=MAIN Macro 3-5
The TYPE=ENVRNMNT Macro 3-9
TYPE=END Macro 3-10
Run SMPLDEFT 3-10
Defining and Allocatig Endevor Libraries 3-11
Allocating the PACKAGE FILE 3-11
Run SMPLJOB1 3-12
Allocating the Master Control Files, Element Catalog, Base, Delta, and Output Libraries 3-14
Allocating and Initializing ACM Query Files (SMPLACMD) 3-24
Implementation Checklist 3-27
Defining the Sample Application Inventory Structure (SMPLJOB3) 3-29
Populating the Sample Application (SMPLJOB4) 3-31
Verifying Implementation 3-33
Checking the Environment Selection Menu 3-34
Running the System Inventory Summary Report 3-34
Performing Foreground Verification 3-42
Performing Background Verification 3-49
Performing ACMQ Verification 3-53
Other Sample Jobs 3-55
In Summary 3-56

Chapter 4: A Day in the Endevor Life Cycle

Retrieving an Element 4-2
Adding an Element 4-5
Browsing an Element 4-8
Displaying the Element Master 4-11
Displaying Element Changes 4-12
Displaying Element History 4-13
Displaying the Summary of Levels 4-14
Displaying Component Information 4-14
Displaying Summary of Levels for Element Components 4-17
Moving Elements 4-18
Packaging the Element 4-23
Creating a Package 4-24
Moving Elements to the Package 4-26
Casting the Package 4-29
Executing a Package 4-33

Appendix A: Testing New Endevor Release Upgrades With Your Data

Creating Your Element Catalog A-2
Converting the Master Control Files (MCFs) A-2
Converting the Package File A-3
Updating Your C1DEFLTS Table A-3
Loading the Element Catalog A-3
Allocating ACMQ Files A-4
Synchronizing the Element Catalog with Your MCFs A-4
Allocating Base, Delta, Source/Processor Output Libraries A-5
Identifying Test Libraries to Endevor A-5
Test A-6

Appendix B: Frequently Asked Questions

Questions and Answers B-1

Chapter 1

Introducing AllFusion Endevor Change Manager

Welcome!

Congratulations! You have chosen AllFusion™ Endevor® Change Manager (referred to simply as Endevor), Computer Associates' comprehensive solution for managing your software development environment. Endevor enables organizations to control all software management tasks associated with the mainframe development environment through its automated transformation functionality, module relationship management, parallel development management and release automation. It eliminates the manual steps that bog down the software development process and ensures greater efficiency with fewer errors.

The software development process has never been more complex and difficult than it is today, with the pressure to manage complex application development projects incorporating a multitude of new technologies and spanning diverse platforms—all without sacrificing productivity. Today's organizations need an effective automated software management solution that will enable them to create a software development process as advanced as the tools they use and the platforms upon which they build applications.

Endevor automates the management of the entire development process—from initial design through distribution—for greater efficiency and complete control. Endevor provides the flexibility to adapt to an organization's specific business requirements, while ensuring consistency and complete control. Its life cycle processes are defined and administered according to the enterprise's unique requirements to categorize and protect software assets and provide application integrity.

Extensive Business Benefits

Endevor helps developers meet demanding delivery schedules while maximizing productivity and reducing development costs. This powerful tool provides the following unique benefits:

- **Expedites movement through the software life cycle** by enabling users to move any component from one phase of the development process to the next with a single command. For example, a single Endevor action prompts it to promote source and executables along with their associated forms, documents and JCL, as well as clean up prior libraries.
- **Drives the production turnover process** by verifying approvals, recording changes and executing a change package in the established time frame without duplication of effort or risk of manual errors.
- **Provides maximum application integrity** through customizable controls and comprehensive audit trails. Endevor can help organizations fully prepare for any EDP audit, enabling companies to enforce established policies and procedures for higher quality applications with less production downtime.

- **Categorizes and protects software assets** by enabling organizations to establish software life cycle procedures that conform to their unique requirements. It provides an inventory of all the components of the software environment across development groups and heterogeneous platforms. With the unique source classification, customizable life cycle, multi-platform management and comprehensive reporting of Endevor, organizations are in complete control of the software development process.
- **Ensures a flexible implementation** by enabling organizations to streamline existing development processes or examine and completely reengineer them upon implementation. Endevor conforms to the needs of the business.

Distinctive Features

- **Automated Source Transformation—**Executables are automatically created whenever source is changed. Endevor can optionally stamp all of the components used in the build process to ensure source to load synchronization in every stage of the development life cycle. Its powerful management capabilities are language-independent—source can encompass a wide variety of application components, including traditional program code that must be compiled and link-edited; JCL that must be validated against corporate standards; load modules for operating system software; database components; or any other site-specific entities used in application development.
- **Parallel and Concurrent Development Support—**Endevor protects against change regression by providing signin/sign-out security at the component level. While assigned, other users can browse or retrieve copies of the component, but are prohibited from updating it into Endevor. For parallel development situations in which more than one developer concurrently works on the same component, a secured sign-out override is available.

- **Release Automation—**The Release Automation functionality of Endevor streamlines the entire software development life cycle by providing productivity aids such as electronic approvals, automated backouts, reporting capabilities and integration with software distribution facilities.
- **Emergency Management—**Endevor allows organizations to define their life cycle to handle emergencies. In addition to standard change and release packages, organizations can build emergency packages with special approval requirements.
- **Robust Security—**Internal security capabilities help protect components and the actions performed on them. Users can implement the External Security Interface (ESI) to employ other security packages like *e*Trust™ CA-ACF2® Security, *e*Trust™ CA-Top Secret® Security and IBM's RACF.
- **Source and Executable Synchronization—**Endevor allows organizations to synchronize source and executables. With a unique audit stamp called a *footprint*, Endevor precisely identifies the version of the source that produced a defined output, linking an executable component directly back to its source.
- **Library Management—**Endevor provides its own library management facility, which stores changes in full-image, forward or reverse delta format. It can also interface with existing library management systems, such as AllFusion™ CA-Librarian® or AllFusion™ CA-Panvalet®.

Change and Configuration Management Across the Enterprise

Endevor and AllFusion™ Harvest CM provide integrated software management facilities across a multi-platform environment via AllFusion™ Change Manager Enterprise Workbench. The integration of Endevor and AllFusion Harvest CM helps organizations manage applications that are developed and deployed on multiple platforms. For multi-tier applications, this integration ensures that all the components of an application are promoted and demoted together so that the correct versions of all components move into production simultaneously.

AllFusion Change Manager Enterprise Workbench provides a web-based interface that allows access to Endevor and AllFusion Harvest CM functions, including approvals, promotions and demotions. This interface allows users to manage a multi-platform project as a single entity. Web-based access from anywhere means that approvers no longer need access to Endevor or AllFusion Harvest CM software and can perform CCM tasks with a minimum of expertise.

To manage application development performed on diverse platforms, Endevor interfaces with Unicenter® Event Manager. Through Unicenter Event Manager, Endevor can interface with other development-related solutions, such as software distribution, help desk and asset management solutions.

To help organizations fully exploit the advanced capabilities of Endevor, we offer the following additional solutions:

- **Endevor interfaces** are available to link Endevor with IBM's Tivoli Information Management for z/OS and Unicenter® CA-Netman®, Unicenter® CA-Roscoe®, Advantage™ CA-IDMS®/DB Database, AllFusion™ CA-Panvalet® and AllFusion™ CA-Librarian®. An extended security interface enhances its security facilities by allowing interconnection with existing systems like *e*Trust CA-Top Secret Security, *e*Trust CA-ACF2 Security and IBM's RACF.

- **AllFusion™ Endevor® Change Manager Option for DB2** provides effective management of DB2 applications and resources. This optional component enables users to track the development of all programs, macros, copybooks, object modules, load modules and components of DB2 applications. With this option, users can gauge what impact a change to a DB2 object will have on an application. It synchronizes the changes made to application components and their related DB2 components through fully integrated catalog management functionality.
- **AllFusion™ Endevor®/DB Change Manager for CA-IDMS®** runs in the Advantage CA-IDMS/DB dictionary-driven environment. This option provides automated facilities for performing change identification, change management and promotion management in the Advantage CA-IDMS/DB environment. And it accommodates both small and large-scale Advantage CA-IDMS/DB installations, working in conjunction with existing standards and procedures.
- **AllFusion™ Endevor® Change Manager Parallel Development Option** operates as a standalone solution or with Endevor to automatically integrate new releases of vendor application software with in-house customizations or concurrently developed in-house software. It serves as a highly effective tool for integrating concurrent development activities and applying vendor updates to customized packaged applications. Users can compare individual components or entire software systems from version to version and release to release, and identify changes and potential conflicts.
- **AllFusion™ Endevor® Change Manager Automated Configuration Option** automatically tracks changes to software components and component relationships, providing a comprehensive history of software configuration as it evolves. In this way, users can determine exactly what has changed from compile to compile, even if the program itself has not changed.

Documentation and Support

Endevor provides extensive documentation in convenient formats, including Adobe's Portable Document Format (PDF) and IBM's BookManager format. The documents are available on the product tape, on CD-ROM, and through the website esupport.ca.com.

This guide is your starting point. Use it to:

- Learn about Endevor's features, functions, and concepts
- Understand how to implement the Endevor sample application, which is provided as a training tool
- Trace Endevor lifecycle functions through scenarios using the sample application
- Review answers to questions frequently asked by Endevor clients
- Learn how to test new release upgrades using your own data

Computer Associates provides unsurpassed service and support to its clients. Technical support is available 24 hours a day, seven days a week. For assistance with this product, simply contact Computer Associates at esupport.ca.com.

Note: See the *Installation Guide* for detailed installation instructions. Use the installation checklist provided in that guide to lead you through the installation process. The checklist outlines the tasks that your site's system programmer and your Endevor administrator need to perform.

Chapter 2

A Conceptual View of Endevor

Introduction

Endevor provides automated facilities for performing all software management tasks from inventory management and change control to configuration and release management. These facilities are further enhanced through comprehensive change administration, parallel development management, software security and software information management facilities. As a fully-integrated, single-vendor solution, Endevor dramatically improves the operation and administration of IBM mainframe installations by providing:

- A comprehensive inventory of all programs and software assets that reside in partitioned data sets (PDSs), library management systems, HFS directories, and executable libraries.
- An absolute history of all changes that have occurred to the source.
- Control of the processes and procedures that translate source into executable.
- An inviolate and auditable link between the source code and its related executable forms.
- Protection and control of inventory items through extended security.

- Automated cross-referencing of software component relationships for purposes of historical analysis, change impact analysis, re-creation of prior versions and release management.
- Control and automation of the movement and distribution of software release packages from stage to stage, site to site, and across networks.

Endevor accommodates the diversity of both small and large-scale IS operations. And Endevor works in conjunction with existing procedures, structures and standards—rather than imposing new ones.

Inventory Management

In most organizations, the application inventory is large, complex and always changing. Today's applications have graduated beyond standard source to include a wide variety of components—most outstripping traditional 80-character storage limitations. In addition to the changing complexion of the inventory's physical structure, there is a growing proliferation of physical libraries and functional environments (Test, QA, Production, etc.). Programmers and IS support personnel typically require intimate knowledge of those library structures as they relate logically to business units within an organization. This logical view, divorced from the physical storage structure, is required both to ensure that logically related inventory elements are being manipulated consistently and correctly, and to provide a common user interface regardless of physical structures. An effective inventory management structure also forms the foundation for all change control, configuration management and release management activities.

Automated Inventory Manager

Traditional software classification systems (PDSs, library management systems, and so forth) provide limited inventory classification capabilities and highly restrictive methods for storing and re-creating prior versions of software modules. When using these traditional systems, IS departments resort to complex naming conventions and physically separate libraries to obtain a semblance of organization. This indirect naming and storage technique is both error-prone and inflexible. Consequently, as programmers move from project team to project team, they are forced to learn and re-learn the peculiarities of manipulating the software inventory as applied by each project team. Clearly, a common technique is required for classifying software throughout the IS organization—one that provides both consistency and flexibility.

Logical Structure

Endevor employs advanced classification techniques that form the necessary foundation for categorizing, viewing and manipulating the application software inventory in a consistent manner. An inventory item (element) is identified to Endevor's Inventory manager by a fully-qualified name consisting of its environment, stage (location), system, subsystem, type and element name.

Environment, Stage—As an application is modified and its elements are moved throughout the software development life cycle, those elements may reside in different functional locations (Test, QA, Production, Backup, etc.) at different times. Within Endevor's inventory classification scheme, the location of an element is part of the identification of that element.

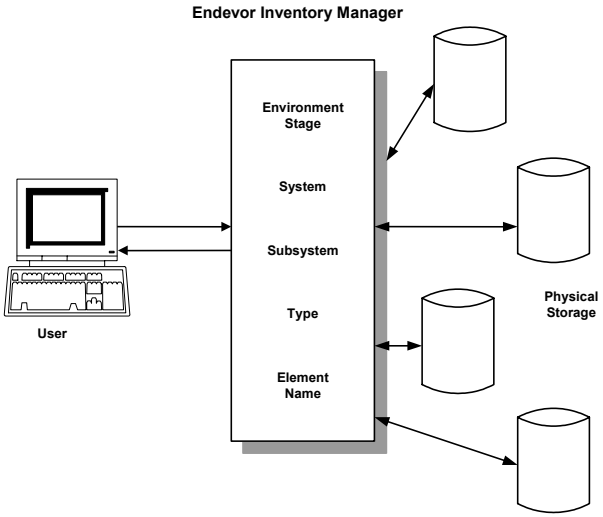
System—Systems represent the logical grouping of inventory elements as they apply to major applications, departments or work areas within an organization. Typically, a system is the "organizational owner" of a portion of the inventory. For example, the systems in an organization might include Finance, Personnel and Manufacturing.

Subsystem—Subsystems are logical sub-classifications within systems. For example, the system Finance might be divided into logical subsystems such as General Ledger, Accounts Receivable, Accounts Payable, Common Routines and Reports.

Type—Within the application inventory, types represent the form of the element, indicating how the element is created (the source language used) and how it is manipulated. To continue our example, the Finance system might have several types of elements, including COBOL programs, Assembler programs, C programs, PL/I programs, copy members, Assembler macros, screen definitions and linkage editor statements.

Element Name—The element name is the existing name of the element, as it is already established.

This logical classification scheme, as illustrated below, provides a consistent view of the inventory without requiring the user to understand the inner workings of the inventory's physical structure.



In Endevor, the logical attributes used to classify an element include its location,. system, subsystem and type are a direct part of the identification of that element. This eliminates the need to establish additional libraries, rename elements, or use cryptic naming conventions to define and maintain a classification scheme. And because these classification attributes are used in addition to the existing element name, there is no need to rename elements to bring them under the control of the Endevor Inventory Manager.

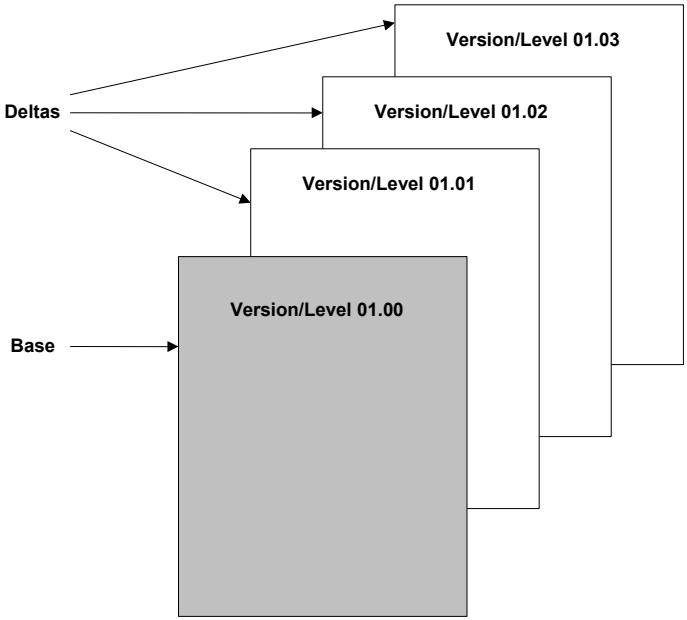
Physical Structure

In today's IT environment, the physical requirements of a properly managed inventory have grown beyond the capabilities of older library management systems and conventional PDS-based systems. Many library management systems are unable to store source of a length greater than 80 characters, despite the fact that many modern-day languages do not conform to this format. Conventional PDSs do not restrict record length, but elements of different record lengths must reside in physically separate libraries. Using these structures, the only way to keep multiple revisions of an element is to either replicate libraries to hold prior revisions or use a naming convention in which the revision is part of the element name.

Physical repository—Endevor's Inventory Manager has the flexibility to manage the physical data sets in which elements are stored, without regard to logical classification boundaries or trivial physical differences between the inventory elements being stored. With Endevor, it is possible to:

- Handle unrestricted source record lengths.
- Store elements of different record lengths in the same repository.
- Store multiple revisions of an element in a single repository.
- Store elements of the same name in the same repository, regardless of differences in logical classification.
- Change the underlying physical library structure without affecting the logical view of the inventory.

Base/Delta—Endevor employs advanced base/delta technology to store the elements within the application inventory. The first time an element is added into Endevor, it is stored in its entirety as the *base.* Subsequent updates to that element create *deltas* records containing only the changes to the element. These deltas are automatically assigned level numbers and stored by Endevor as illustrated below.



This method yields substantial savings in storage overhead by providing an efficient means of storing multiple revisions of an element. It also provides a wide variety of ways to view change information, including current, historical or changes only, available online or in hardcopy format. All change displays identify what changes were made and by whom, when and why. This is known as the forward delta format. Reverse delta format processing is similar; however, the delta source is kept in the current format and deltas are used to re-create prior iterations of the source.

Interfaces to AllFusion CA-Panvalet and AllFusion CA-Librarian

Interfaces are available that allow Endevor to read directly from and write directly to AllFusion CA-Panvalet and AllFusion CA-Librarian files. Additionally, these interfaces allow you to use AllFusion CA-Panvalet and AllFusion CA-Librarian files to contain base/delta source within Endevor. In this way, your initial investment in library management systems is enhanced rather than rendered obsolete.

Change Control

Applications typically undergo extensive modifications as companies customize and fine-tune software systems to meet their specific business requirements. Given the sheer volume of change requests in today's IT organization and the growing demand for more effective ways of tracking and documenting development change activity, manual change control procedures can no longer be relied upon to produce the accurate, reliable information needed to identify problems, track projects and analyze the impact of change.

Automated Change Control

Endevor forms an "envelope" around your application environment and captures all change events, enabling you to identify what has been changed, who made the changes, when and why. And since Endevor performs change control functions automatically, the resulting information is always accurate and up-to-date.

Identifying and Tracking Changes

As a prerequisite to understanding, controlling and auditing the software environment, change control systems must be able to identify changes as they occur, then further pinpoint when the changes were made, by whom, when and why.

CCIDs—Endevor Change Control Identifiers (CCIDs) provide a means of grouping and manipulating similar kinds of change activity. For example, a number of different elements (COBOL copy members, COBOL programs, and so forth) can be included in a single change request. By tagging each of those changes with a common CCID (such as 'FIX01'), every change made for that change request is categorized as belonging to that CCID. Subsequently, the CCID can be used to identify and manipulate all the elements within the change request, for example: "Move all elements for COD FIX01 from Unit Test to System Test." CCIDs can also be predefined and validated for change administration purposes.

Change History—Endevor automatically captures and dynamically tracks all source changes by creating a unique delta to record each change event. An element's delta levels identify not only the actual lines of code that have been changed, but also user, comment, CCID and date/time information for that change. The result is a complete and accurate history or audit trail of application change activity.

Regression Notification—In parallel development situations where many individuals work on the same module concurrently, change regression occurs when one individual unintentionally negates (rather than incorporates) the changes made by another individual. When unchecked, regression can be a major cause of production failures. During action processing, Endevor performs a regression check and notifies you when regression has occurred, allowing you to quickly pinpoint and correct the situation before it adversely affects the production environment.

Signin/Signout—In order to avoid possible regression during parallel development, Endevor provides signin/signout security at the element level. This optional capability is enabled on a system-by-system basis. When the facility is enabled, elements are signed out as they are retrieved. While other users can retrieve copies of those signed out elements, they cannot update those copies within Endevor without explicitly overriding the original user's signout. Only authorized users can exercise the explicit override feature.

Activity Logging—A complete log of all activity within Endevor can be written to a variety of data structures, including standard SMF records. This information, available online and in hardcopy format, provides a high level audit trail of change events which can be used to determine change patterns over time.

AllFusion Endevor Change Manager Interface for IBM's Tivoli Information Manager

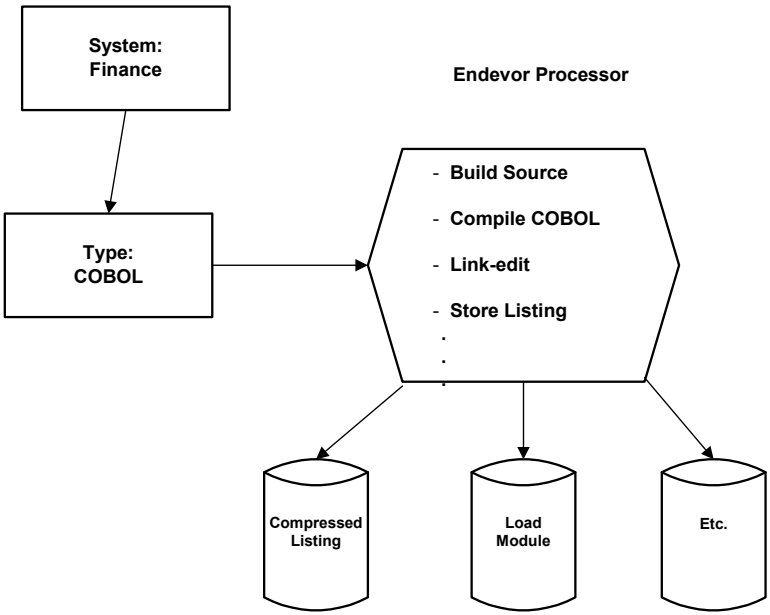
Endevor provides a flexible, programmable interface to IBM's Tivoli Information Manager change management product. This interface allows certain change administration functions to be performed through Tivoli Information Manager on behalf of Endevor. These functions include: the validation of CCIDs against the Tivoli Information Manager data base; the logging of Endevor actions to Tivoli Information Manager; the ability to use Tivoli Information Manager's approval and notification facilities; the ability to control Endevor actions based on Tivoli Information Manager's CCID status information; and the ability to query and utilize Tivoli Information Manager's change, problem and configuration information during Endevor action processing.

Automating Processes

In application development, a procedure or process is the set of steps used to transform source language statements into executable code. Some inventory elements, such as those written in 3rd generation languages (COBOL, Assembler, C, PL/I, etc.) have straightforward processes that compile the source and link-edit the object. Other element types require more complex processes, such as the resolution of data base calls or the automated generation of textual documents. Because these processes ultimately determine how an element will interact in production, the management of these procedures is critical.

One of the powerful facilities provided by Endevor involves the regulation and control of all transformation processes through automated procedures called processors. Endevor processors are written in job control language (JCL) statements that specify process steps, parameters, error conditions, data set names, output libraries and the like. Once defined, processors can be used to automatically carry out a variety of procedures and create the outputs associated with application elements. Element types within the inventory classification determine the appropriate processors. For instance, elements that are written in COBOL are typically handled by the COBOL compiler and linkage editor, while elements that are classified as copy members are usually validated and copied to a specific location.

Automated Output Generation—Endevor processors are often used to automate the procedures that generate outputs from source. As illustrated in the diagram below, you can create a processor for Batch COBOL programs that automatically performs a COBOL compile and link-edit into the Finance system's load library, and stores the listing in the Finance system's listing library.



Automated Promotion Procedures—A processor can also contain procedures for promoting applications throughout the development life cycle. When Endevor is instructed to perform a promotion, it reads the processor definition to obtain directions about the promotion procedures specified for a selected element type. Those procedures are then uniformly invoked by Endevor. When promoting a Batch COBOL program within the Finance system from Testing to Production, for example, a MOVE processor can be defined which contains instructions for automatically copying the load modules from the Test load library to the Production load library.

Automated Backup and Recovery—In addition to automating the procedures used to generate outputs from source and promote applications throughout the development life cycle, processors can be defined to execute emergency backup and recovery procedures. For example, a MOVE processor can contain a step which automatically copies production load modules to a backup library. A recovery procedure can additionally be defined to copy that module from the backup library into an emergency library in the event of a production failure.

Establishing Source-to-Executable Synchronization

In order to effectively and accurately maintain, debug and audit software, you must be able to link executable code back to its originating source. Endevor achieves this goal by controlling the processes by which outputs are created from the source, and further ensures source-to-executable synchronization by audit stamping or footprinting those outputs.

Process Control—Endevor's automated processors ensure that the source-to-executable link is not compromised, by tying the process used to create outputs directly to the manipulation of the source. Frequently, load modules and their associated source are processed at different points in time, presenting a precarious "window" in which a change to either one causes an out-of-sync condition. When source is presented to Endevor, the outputs are automatically created from the current source through the use of automated processors. When source is moved through the development life cycle, Endevor's automated processors ensure that the load modules and outputs are manipulated together with the source, rather than through separate procedures.

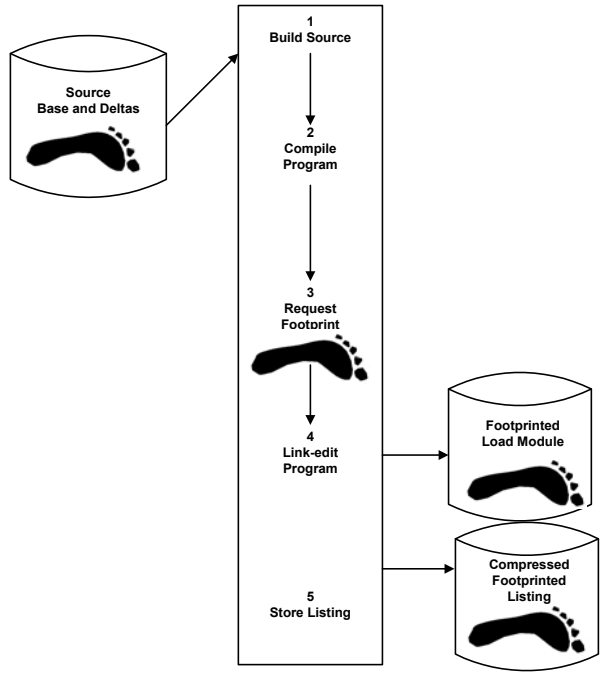
Footprints—Footprinting is a change control technique employed by Endevor to maintain synchronization between source and its related executable. During element processing , Endevor places an encrypted audit stamp called a footprint in the output source, object or load modules that are created. Only Endevor automatically builds footprints as an integral part of the source-to-executable transformation process. Endevor footprints contain the following information:

Location information
Element name
Element type
Element version/level
System and subsystem to which the element belongs
Date/time the footprint was assigned

Endevor employs standard operating system facilities to store its footprints, thereby ensuring compatibility with present and future operating systems. In a PDS, for example, the footprint is stored in the user data area of the directory; in AllFusion CA-Librarian, as a history record; in AllFusion CA-Panvalet, as a comment field; in a load module, in the user IDR record for the CSECT. All data structures managed by Endevor (source libraries, copy libraries, executable libraries, and processor listing libraries) contain footprinted elements.

Footprints can be used to validate the source-to-executable link of an entire load library. And because Endevor footprints uniquely identify the versions/levels of input components, they play a key role in configuration management.

The following diagram provides a high-level view of how automated processors and footprints work together.



As illustrated above: 1) the source is built from Endevor's internal base/delta format, and a footprint representing that element's source is created and made available to subsequent steps in the Endevor processor; 2) the source is compiled; 3) the output object from the compile is stamped with a footprint before being passed to the next step; 4) the linkage editor links the program into the appropriate load library; and 5) the listings from the previous steps are combined and stored (also with a footprint) in the appropriate listing library.

Audit Management

As valuable corporate assets, applications cannot be compromised when it comes to integrity, reliability and recoverability. For this reason, applications must be controlled and secured in a way which meets stringent auditability requirements. Endevor's automated change control functionality fully satisfies the scope of an audit and provides the necessary platform for producing management reports which can be used to evaluate software integrity.

Endevor Actions

Endevor performs software management operations through user-invoked actions which are available in both foreground (online) and background (batch) modes. These actions are briefly described below.

- ADD—The action used to introduce an element into an Endevor environment.
- ARCHIVE—The action used to copy an element and all of its change history and control information from an Endevor environment to an external data set.
- DELETE—The action used to remove an element from an Endevor environment.
- GENERATE—The action used to execute the Endevor automated processor for an element.
- LIST—The action used to scan members in a library or the Endevor inventory to generate a list of elements that meet specific selection criteria.
- MOVE—The action used to promote an element from stage to stage within an Endevor environment.
- PRINT—The action used to produce hardcopy output of Endevor element, change and change history information.
- RESTORE—The action used to re-introduce an element to Endevor from an archived data set.

- **RETRIEVE**—The action used to copy any level of the source of an element from an Endevor environment into an external data set.
- **SIGNIN**—The action used to remove the user signout associated with an element.
- **TRANSFER**—The action used to transport an element from one location to another, where each location can be either an Endevor location or an external data set.
- **UPDATE**—The action used to create a revision of an element in an Endevor environment.

When applicable, these actions drive the automated processors that have been defined to Endevor. Before an action is executed by Endevor, a security check confirms the user's authority to perform that action upon the requested element. All actions and their results are reflected in Endevor execution processing reports. Additionally, all Endevor activity (transactions) may be recorded via SMF.

Endevor's Software Control Language (SCL)

The tedious and repetitive tasks associated with manipulating the software inventory are time-consuming and prone to error. Endevor's Software Control Language (SCL) is a powerful yet simple language which, when combined with Endevor actions, eases the burden of manipulating and moving portions of the software inventory throughout the entire development life cycle.

SCL Capabilities

SCL saves substantial amounts of time by enabling you to work with as many (or as few) Endevor actions as are required to complete a specific job. With SCL, you can:

- Perform bulk data manipulation.
- Set up a single list or multiple lists of actions for manipulation by Endevor.
- Establish standardized global settings for action requests.

- Process actions in inventory type sequence order, automatically sorting elements according to specification.
- Generate software configuration lists based on different selection criteria.
- Use a single scan facility that will run against AllFusion CA-Panvalet, AllFusion CA-Librarian, a PDS and Endevor, eliminating the need to use separate utilities to scan source code.
- For problem-solving purposes, generate a list of only those elements which were not successfully processed at a specific time.
- Write user-defined front-ends for various vendor-supplied programs.
- Integrate Endevor into existing job scheduling systems.

SCL can be generated and manipulated through online screens, providing an easy-to-use, flexible environment for executing software procedures. Using SCL, repetitive tasks which normally require significant amounts of time can be accomplished with very little effort.

Configuration Management

Historically, a major deficiency in the management of software systems has been the lack of an accurate means of determining the interdependencies or *configurations* of applications and their related components. Semi-automated source scanning techniques have provided a partial solution. These techniques are no longer feasible, however, given the size and complexity of today's applications, the frequency of component changes—and the growing demand for a more auditable means of tracking and managing software configurations as they evolve over time.

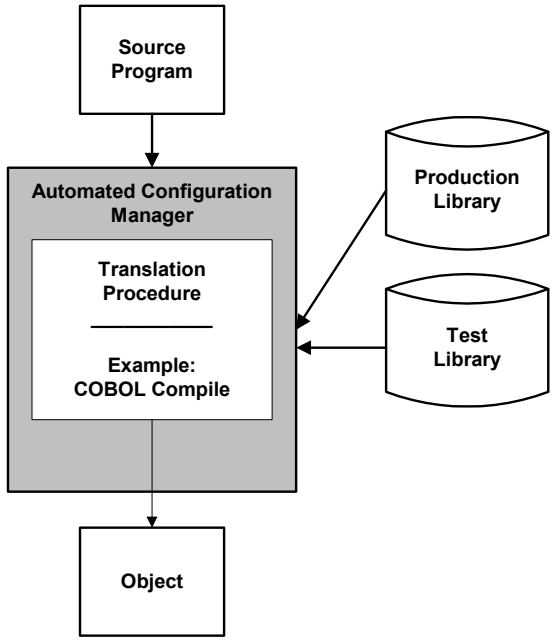
AllFusion Endevor Change Manager Automated Configuration Option

Built upon a logical inventory structure and a powerful change control platform, Endevor provides an automated configuration management facility (ACM) that monitors and establishes accurate configuration information.

Defining Software Configurations

When a program or module is translated from source to executable, all of its related components are collected from their resident libraries (copylibs, maclibs, and so forth). in order to keep an accurate record of changing software configurations, the process of tracking program-component relationships and associated change activity must be performed automatically as an integral part of the translation procedure.

The Component Monitor—While a program or module is being translated by a processor, ACM automatically captures the program-component relationships as they are resolved from the selected data sets, as illustrated below:



The Component List—As output from the translation procedure, ACM automatically produces a "snapshot" or Component list for each monitored program. This Component List is an internal data structure that preserves a physical profile of the configuration.

As illustrated below, the Component List identifies all input program components and where they originated, as well as the output components created as the result of the translation procedure. Footprint information, including the version and level of the component, is also noted if present.

Configuration Type	Name	Footprint Information	Step	Library
Element	PROGRAM X	v1.3		
Processor Record	COMPLINK	v1.8		
Input	COPYRECA	v2.1	Compile	COPYLIB1
Input	COPYRECB	v2.5	Compile	COPYLIB2
Output	PROGRAM X	v1.3	Compile	OBJLIB
Output	PROGRAM X	v1.3	Link-Edit	LOADLIB

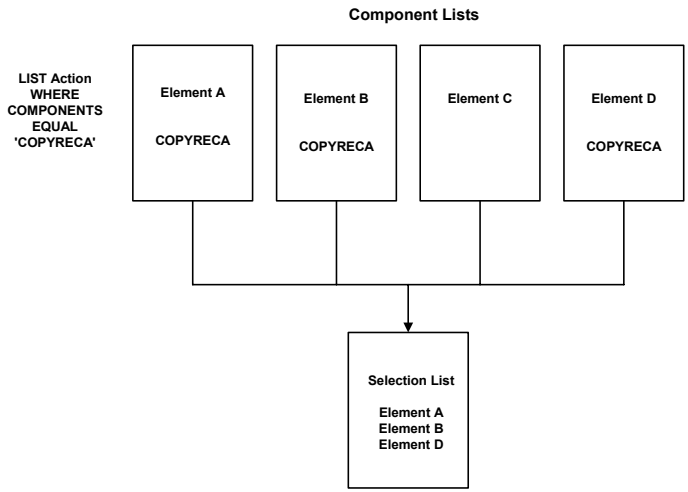
The element information describes the program being generated. The processor information describes the Endevor processor used to generate the element. The input components identify the component items referenced and any footprints that exist in the monitored data set. The output components identify the elements that were created during processor execution.

- **Storing Component Lists**—The first time a Component List is created, it is stored in its entirety as the *base*. Subsequent differences in Component lists from one program translation to the next are stored as *deltas* (changes only) and automatically assigned a component level number. By using base/delta technology to store Component Lists, ACM enables you to compare component changes from compile to compile—a capability unavailable through source scanning techniques.

- **Viewing Component Lists**—When stored over time, Component List information becomes the basis for maintaining and viewing component history. Through online displays, ACM lets you view Component List information from four perspectives: 1) summary of component levels, 2) current level of a component, 3) component changes, and 4) component history.

Using Software Configurations

Using the information contained within Component Lists, it is possible to create a selection list. In the diagram below, a selection list has been created by using Endevor to LIST all programs WHERE COMPONENTS EQUAL 'COPYRECA.'



As a powerful cross-reference facility, the LIST action in combination with selection lists in SCL format, provides the necessary foundation for performing a number of essential configuration management functions.

Propagating Component Changes to Related Programs— When a component is changed, it is necessary to propagate those changes to all affected programs. Using the LIST action, you can perform change impact analysis by identifying every program containing that changed component. And you can also use the resulting selection list to perform bulk data manipulation by re-compiling those programs, all at the same time, to reflect the change.

Validating a System for Consistent Use of Components —In order to prevent production failures, it is desirable to validate that all programs are using the correct component versions/levels. Execution reports from LIST action execution, in combination with selection lists, can be used to identify inconsistencies.

Re-Creating Past Configurations—It is sometimes necessary to re-create a program exactly as it has existed at a past point in time. Using the LIST action, you can re-create an older version/level of a program's load module that includes the old versions/levels of all related components used at the date/time the load module was originally created.

Component Validation

Component validation provides the assurance, prior to a promotion, that all dependencies such as Copy Member, Include Member, macros, etc. are included in the promotion and that all programs have been compiled with the current iteration of the copy members.

Release Management

Release management activities revolve around identifying and distributing software releases. Depending on the circumstances, a "release" can consist of the elements changed during a maintenance effort, or it can represent an entire cross-section of the software inventory. Distribution of a release can be as simple as promoting a change into production, or as complicated as managing multiple concurrent releases of entire software systems. Accurate and up-to-date information is a prerequisite for performing every task, from creating a list of exactly what is to be released to verifying that the release was successfully accomplished. This kind of information can only be obtained through an automated, integrated release management system.

Automated Release Management

In many installations, production turnover and release management are performed, often manually, as discrete activities by separate groups. Problems arise when software release packages are created manually "on paper," where there is a chance that the contents are incomplete or inadvertently changed prior to or during execution. Endevor provides integrated, automated release management functionality, ensuring that software release packages are built on a solid inventory management structure, created using automated techniques, and implemented as an integral part of the release process. In addition, Endevor's change administration facility provides online release package approval capabilities. Only with these combined capabilities is it possible to accurately create and manipulate a software release, perform change impact analyses, and track changes from release to release.

Change Administration

Endevor provides change administration, in addition to security, as a means of specifically organizing and controlling change to portions of the software inventory. Through the Endevor change administration facilities, it is possible to provide authorization capabilities to the person(s) responsible for portions of the software inventory.

Approver Groups—Endevor establishes change administration authorization through the definition of approver groups (named groups of user IDs) and the subsequent relation of those approver groups to portions of the Endevor inventory. For example, within the Payroll subsystem of the Finance system, you could establish an approver group which approves or denies changes to all elements within that system and subsystem. Some of the users within the group could be required to approve changes while the approval of other users would be optional. A quorum or minimum required number of approvers can be established for any approver group. "EMERGENCY" approver groups can be established consisting of those user IDs which must authorize emergency fixes to a portion of the inventory.

Change Packages—Authorized users build *change packages* which contain Endevor action requests. The user specifies whether the handling of that change package will be defined as 'STANDARD' or 'EMERGENCY'. The user also defines a date/time window within which the change package can be executed. Once the change package is built, it is then CAST, a process which causes Endevor to prepare the package for subsequent approval processing. Endevor automatically builds the list of approvers for a change package based on the previously defined approver groups. The change package then goes through approval processing, followed eventually by execution.

How It Works

- **Building a Change Package—**Authorized users build change packages consisting of any number of Endevor action requests. Change packages can be built from online screens, imported from outside data sets and/or copied from existing packages. They can be edited, modified and/or appended to at any point during the build process.
- **Preparing a Change Package for Review—**The change package is CAST. The process of casting a change package causes the approvers for that package to be built based on the previously defined approver groups. Once a package is CAST, it can no longer be modified.
- **Reviewing a Change Package—**Users in the approver group(s) associated with a package can approve or deny the change package. Required users must provide their approval. Once all required approvals have been made, and the established quorum of approvals has been met, the change package can be executed.
- **Executing a Change Package—**At execution time, Endevor validates that all approvals have been made and that none of the elements have changed between the time the change package was CAST and when it was executed. Endevor also validates that the change package is being executed within the date/time window established when the change package was built. Endevor then executes all of the actions in the change package. Checkpoint/restart facilities are available so that change packages that fail execution can be re-executed properly. Change packages can be executed online or in batch.
- **Automatic Package Backout—**Subsequent to execution, it is possible to quickly and easily back out a change package. BACKOUT has the effect of reversing any outputs created by the execution of that package to the state they were in prior to package execution. BACKIN reverses the process.

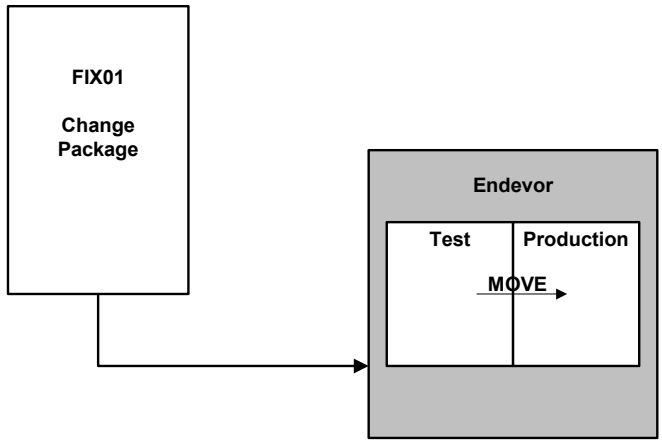
Managing Production Turnover

Using the Component Lists created by Endevor's Automated Configuration Manager in combination with Endevor's logical inventory structure, you can build a change package containing all of the programs and components that make up a maintenance release. Once the package has been built, it can be used to drive production turnover for that maintenance release.

Creating a Maintenance Change Package—When Change Control Identifiers (CCIDs) are used to tag all of the changes which make up a maintenance release, those CCIDs can be used as selection criteria to produce a package that itemizes all of the elements which comprise that maintenance release, as well as all related element components.

For example, you could build a package that contains MOVE commands for all of the elements changed for CCID 'FIX01' as well as all the input components for those elements.

Promoting a Maintenance Release—Once the change package has been constructed and has gone through the appropriate approvals, it can be used to perform the production turnover. In our example, execution of the ' FIX01' change package will automatically perform production turnover, promoting all of the elements contained in the package.

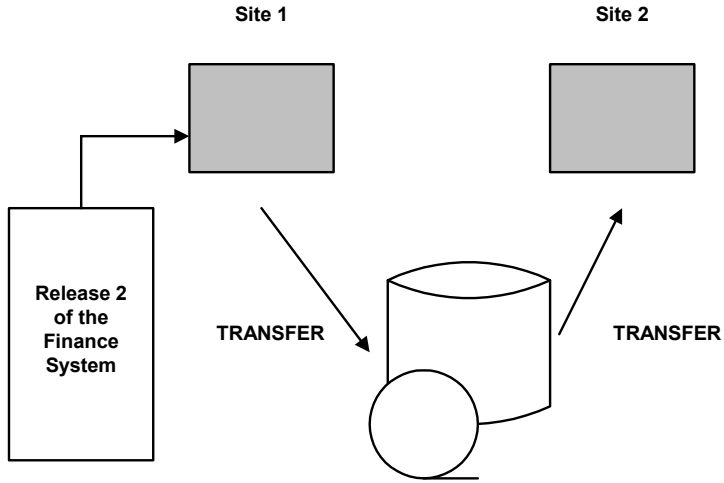


Managing Software Releases

Endevor's powerful inventory structure, together with ACM's Component Lists, can be used to manage all aspects of releasing software. Once a change package has been generated, its contents can be used to drive the distribution of the software release, track changes from release to release and re-create previous releases.

Creating a Software Release Package—The inventory management structure of Endevor and the configuration information created by ACM can be used to create complete, reliable and reusable software release packages. For example, a complete release package could be built containing all of the elements in Release 2 of the Finance system, and all of the components belonging to those elements. The resulting package would contain not only all of the elements that comprise the software release at this point in time, but information about the current versions and levels of those elements.

Release Distribution—Once a release package has been constructed, it can be used to drive the distribution process, regardless of whether the software is to be moved to another Endevor location in-house, transferred to another Endevor location at a remote machine or site, or distributed to external files, perhaps for shipping to customers or divisions. In our example, the contents of the package created above are used to drive the TRANSFER of Release *2* of the Finance system from Site 1 to Site 2 :



Using the contents of release packages to drive the distribution process provides the added advantage of automating both the distribution process and the documentation of that process.

Release to Release Comparison

An important aspect of any type of release management system is its ability to track changes from release to release. This broad, system-wide perspective is necessary when managing multiple software releases. Packages are kept in such a way that once created, their contents can be used in a number of ways. They can be compared to find out exactly what has changed from one release to the next. If the contents of the package are extracted and stored in Endevor's base/delta format, it is also possible to view changes to a software system over time. This capability is especially useful when trying to identify which modules of a software system have been added, removed or changed prior to distributing a partial release consisting of changed components only.

Supporting Functionality

In addition to its inventory management, change control, configuration management and release management facilities, Endevor provides capabilities in the areas of parallel development management, software security management and software information management. This broad range of product functionality is designed to support the entire software management life cycle, not just one aspect.

AllFusion Endevor Change Manager Parallel Development Option

Endevor provides parallel development management capabilities through the AllFusion Endevor Change Manager Parallel Development Option (PDM). Using this powerful development tool, it is possible to create a merged program from the comparison of a base program and two independently modified versions of that base program. By eliminating many of the time-consuming, manual tasks associated with merging parallel development updates, PDM saves time and reduces the margin for error. PDM also provides a means of identifying and resolving conflicts before merging program updates. For this reason, PDM is a necessary tool for effectively managing concurrent application development and resolving problems which arise when integrating in-house modifications with vendor updates.

Managing Parallel Development Activities

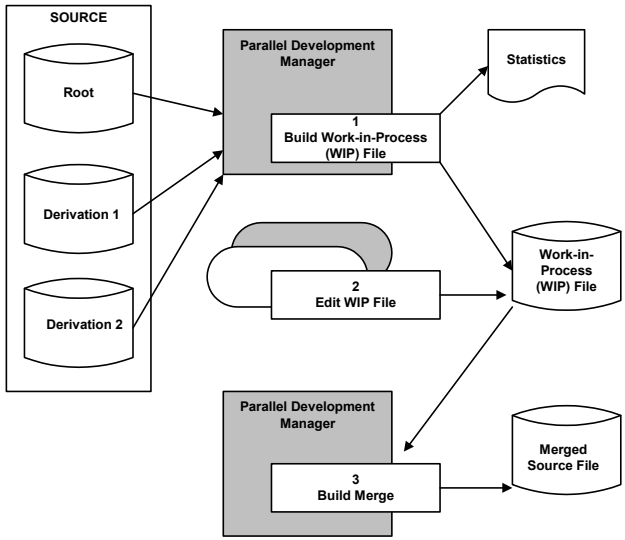
When several programmers work on the same module concurrently, there is always the danger that some modifications will be overridden by others or will be in direct conflict with one another. Beyond the basic frustration factor, this problem can have serious consequences from a control and maintenance standpoint.

Managing Vendor Updates

Purchased software packages are typically customized to meet in-house demands. Unless those modifications are painstakingly documented, problems arise when an update arrives from the vendor. What exactly was changed in-house? And what modules did the vendor change, add or delete? Manual methods for resolving these conflicts are time-consuming and error-prone, underlining the need for automated solutions for merging vendor updates with in-house modifications, and reconciling changes from release to release.

Basic PDM Operation

The diagram below provides a simple illustration of how PDM operates:



In Step 1, the Work-in-Process (WIP) file is created and statistics are reported. In Step 2, the WIP file is edited. In Step 3, the merged member is built.

Building the WIP File—The first step in using the Parallel Development Manager is to build a Work-in-Process (WIP) file. The WIP file is a PDS structure which combines three source files: the Root, Derivation 1 and Derivation 2. As an example, if PDM is being used to manage a vendor update, the Root would be the source for the old release, Derivation 1 would be the source for the in-house modifications made to that old release, and Derivation 2 would be the source for the vendor's new release. As the source input files are being compared to produce a WIP File, statistics are generated. These statistics reveal critical information about the source comparison, such as the number of inserts and deletes per module, the percentage of the module that has changed, and the number of simultaneous (and potentially conflicting) updates.

Editing the WIP File—With statistics in hand, it is now possible to view and edit the members which contain conflicts. At this point, you can decide which conflicts to resolve and how to resolve them.

Creating Merged Source—Once you've viewed and edited the WIP file to your satisfaction, you can use PDM to automatically merge the WIP source into an output source library. This output source is then ready for compilation or, if applicable, introduction into Endevor.

The Parallel Development Manager can be used in both online and batch modes, and can operate on a member-by-member basis on a list of members, or on any portion of the Endevor inventory.

Software Security Management

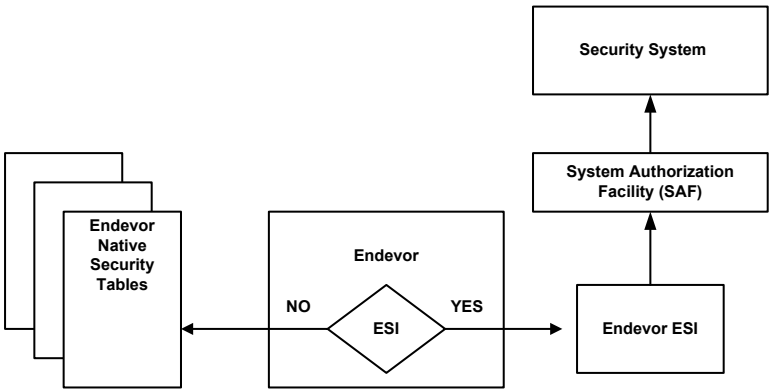
Traditionally, inventory security has been provided at the physical file level but not at the individual element level. Endevor security is based on its inventory structure, enabling security rules to be defined according to that structure—at the system, subsystem, type and element levels. Additionally, Endevor actions are secured, providing a means of restricting individuals to the appropriate change control functions at specific stages within the software development life cycle.

Native Security

Endevor native security provides protection against unauthorized access and processing through three tables: Access Security, User Security and Resource Security. The Access Security Table defines the environments to which each user has access. The User Security Table defines the systems/subsystems within a particular environment and the level of activity for which each user is permitted access (display only, delete, add, etc.). The Resource Security Table defines any element naming conventions that are restricted to a particular system/subsystem and the Endevor actions for which the restriction applies. The use of Endevor native security is optional and is enabled only if one or more of these tables is defined. When defined, access is permitted only through these tables.

AllFusion Endevor Change Manager Interface for External Security

In addition to native security, Endevor provides the AllFusion Endevor Change Manager Interface for External Security (Endevor ESI). With Endevor ESI, installations which currently use RACF, eTrust CA-ACF2 or ETrust CA-Top Secret security systems can secure access to Endevor functions and data sets through those centralized systems rather than locally through Endevor security tables. Endevor ESI accomplishes this through an interface to IBM's System Authorization Facility (SAF). As illustrated in the diagram below, Endevor ESI replaces standard Endevor security tables with an SAF interface module.



With Endevor ESI in place, each security request is checked through the centralized security system in order to validate access to Endevor environments, menu options and actions against inventory elements.

Software Information Management

Through a combination of its master inventory data base (Master Control File) and SMF log file (which notes every change attempted and completed, as well as security violation information), Endevor provides extensive display and reporting facilities. The resulting information provides managers with an accurate and reliable means of performing workflow analysis, project tracking, trouble-shooting and audit management.

Displays

Endevor provides extensive display capabilities for viewing status and change information online. The following displays can also be printed in hardcopy format:

- Element Master Control File Information
- Element Change Summary
- Element Change History
- Element Changes
- Element Browse
- Component Summary
- Component History
- Component Changes
- Component Browse
- Footprint Display

Reports

Endevor provides four types of reports: Master Control File, Historical, Footprint and Change Administration. SAS versions of many of these reports are available in source form to allow you to modify basic reports or create new ones.

Master Control File Reports. The following reports reflect the definitions of systems, subsystems, element types and elements, as specified to the Endevor Master Control File:

- System Inventory Profile
- System Inventory Summary
- Element Catalog
- Element Activity Profile
- Element Activity Summary
- Element Catalog by CCID
- System Definition Profile
- Element Signed Out Profile—By System
- Element Signed Out Profile—By User
- Approver Group Definition
- Approver Group Usage

Historical Reports. The following reports summarize security violations and element activity as recorded by Endevor:

- Security Violation Profile
- Security Violation Summary
- Element Activity Profile
- Element Activity Summary

Footprint Reports. The following reports document footprint information placed in source and load modules by Endevor:

- Library Member Footprint Report
- Library CSECT Listing

- Library ZAPped CSECT Report
- Footprint Exception Report

Change Administration Reports. The following reports document change package activity:

- Package Detail Reports
- Package Summary Report

Endevor Benefit Summary

Based on advanced change control technology which is implemented through an automated, integrated approach, Endevor yields a number of substantial benefits by:

- Providing extensive inventory management capabilities which accommodate all source and executable, regardless of type or record length.
- Providing a means of categorizing, viewing and manipulating physical inventory components as they relate to logical workgroups.
- Creating an unbreakable audit trail of all change events for purposes of historical analysis and auditing.
- Capturing and storing all change information using advanced, space-saving base/delta storage techniques.
- Assuring an inviolate link between executable code and its originating source.
- Enforcing standard, consistent change processes.
- Providing an accurate means of creating and tracking software configurations.
- Automating the movement of entire software release packages throughout the development life cycle.
- Providing additional approval and notification capabilities through change administration functionality.

- Extending security to the inventory member level.
- Providing a means to compare and implement vendor application updates.

Chapter 3

Implementing the Endevor Sample Application

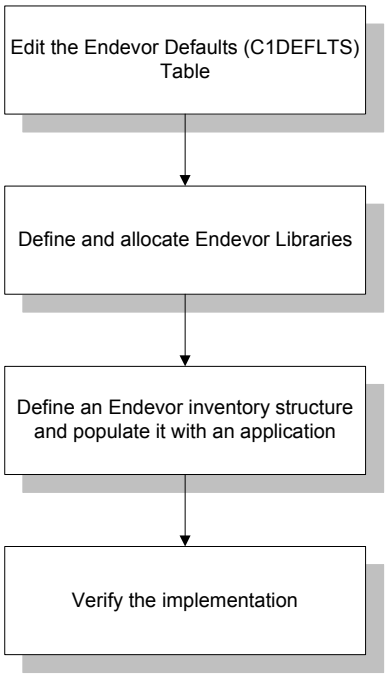
The Endevor Sample Application

Note: In this chapter, you will learn about implementing the sample Endevor application at your site. Ensure that the basic Endevor installation has been completed successfully before implementing the sample application. See the *Installation Guide*, for detailed instructions regarding the Endevor installation.

The Endevor sample application is provided as a training tool to help you understand how to implement Endevor. It reflects a basic software life cycle, including examples of a life cycle definition, inventory definitions, and processors, all of which can be useful to you during your site's implementation of Endevor. The sample application presents one way of implementing Endevor, which may be different from the way that you have implemented it at your site. It may include steps that you do not perform or it may sequence steps in a different order.

Important! If you are installing Endevor for the first time, you must implement this sample application. If you are an existing Endevor site, you can use your existing inventory and life cycle for testing purposes. See the Appendix, "Testing New Release Upgrades with Your Own Data" for information on how to do this.

In implementing the Endevor sample application, you will perform the following steps:



All JCL and source members used in the implementation steps and the processors for the sample environments are included in the Endevor iprfx.iqual.JCLLIB and iprfx.iqual.SOURCE libraries created during the installation process.

Naming Conventions Used in the Sample Application

The naming conventions for the files used by the sample application are described below.

```
iprfx.iqual.SMPLssss.xxxxxxxx
```

Where:

- iprfx is the highest-level qualifier (1-8 characters in length) for the Endevor and sample application libraries and files; for example, SYS3. This value was defined as part of the global edits performed prior to running the Endevor install process.
- iqual is the second-level qualifier (1-8 characters in length) for the Endevor and sample application libraries and files; for example, ENDEVOR.
- SMPL is the identifier used together with the stage name (indicated by the ssss shown above) to construct the third qualifier as follows:
  - TEST—For Stage 1 of SMPLTEST (unit test)
  - QA—For Stage 2 of SMPLTEST (quality assurance)
  - EMER—For Stage 1 of SMPLPROD (emergency fixes)
  - PROD—For Stage 2 of SMPLPROD (production stage)

    For example, if during the install process you changed iprfx to SYS3 and iqual to ENDEVOR, your data sets would be named as follows:

    SYS3.ENDEVOR.SMPLTEST.xxxxxxxx

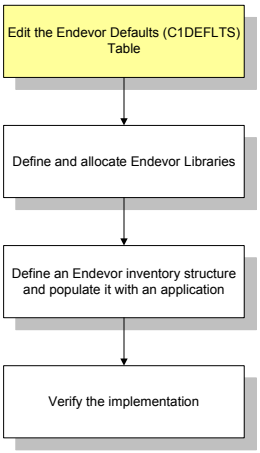
    SYS3.ENDEVOR.SMPLQA.xxxxxxxx

    SYS3.ENDEVOR.SMPLEMER.xxxxxxxx

    SYS3.ENDEVOR.SMPLPROD.xxxxxxxx
- xxxxxxxx is the fourth-level qualifier of the datasets used for the sample application and represents the type of data contained in the file; for example, MCF for the master control file, PACKAGE for the package control file, etc.

Important! File names are limited to a total of 44 characters, including periods. The maximum length of each data set name qualifier is 8 characters.

Editing the Endevor Defaults Table (C1DEFLTS)



The Endevor Defaults Table contains processing and environment definition information for your site. It is comprised of a set of macros which, when assembled and link-edited, are known collectively as the C1DEFLTS Table. The types of C1DEFLTS macros are distinguished by the keyword parameter TYPE.

- One TYPE=MAIN parameter is required for each site, to define general site-specific information. The TYPE=MAIN macro is the first macro in the definition of the table.
- Any number of TYPE=ENVRNMNT macros can be included in the table. You must have one TYPE=ENVRNMNT macro for each environment in your configuration. The TYPE=ENVRNMNT macros follow the TYPE=MAIN macro in the definition of the table.
- One TYPE=END macro is needed to indicate the end of the table definition. It follows the last TYPE=ENVRNMNT macro.

The Sample Application C1DEFLTS Table is in the library iprfx.iqual.JCLLIB, in member SMPLDEFT. For the sample application, you need to make changes as follows:

- Supply some site-specific parameters in the TYPE=MAIN macro.
- Verify MCF qualifiers in the TYPE=ENVRNMNT macro.

The TYPE=MAIN Macro

The TYPE=MAIN portion of the C1DEFLTS Table for the sample application is included below. Only those fields that require editing are shown.

```
Col          Col                                                  Col
1            16                                                   72
↓            ↓                                                     ↓
        C1DEFLTS TYPE=MAIN,                                        X
             ...
             ...
             ACMROOT='iprfx.iqual.SMPL.ACMROOT', ACMQ ROOT DATA SET    X
             ACMXREF='iprfx.iqual.SMPL.ACMXREF', ACMQ XREF DATA SET    X
             ASCM=Y                   ACM CONTROL OPTION               X
             ...
             ...
             CUNAME='*** PUT YOUR COMPANY NAME HERE ***', (50 CHAR)    X
             ...
             ELMCATL='iprfx.iqual.SMPL.ELMCATL', E/MVS ELEMENT CATLG   X
             ...
             ...
             LIBENV=,                                                  X
             ...
             MACDSN='iprfx.iqual.SOURCE', E/OS/390 SOURCE LIBRARY      X
             ...
             ...
             PKGDSN='.iprfx.iqual.SMPL.PACKAGE',    PACKAGE DATASET    X
             ...
             PROC=Y,                              EXTENDED PROCESSOR   X
             ...
             SITEID=0,                ENDEVOR  SITE IDENTIFIER         X
             ...
             ...
             VIOUNIT=TDISK,           UNIT FOR VIO-ELIGIBLE ALLOC      X
             ...
             ...
             WRKUNIT=TDISK,           UNIT NAME FOR WORK SPACE         X
```

You need to edit or supply the following parameter values, as appropriate, in the TYPE=MAIN parameter. Be sure the "iprfx.iqual" values you assigned to data set names in the C1DEFLTS Table match those you defined as part of the global edits done during the install process. See the *Installation Guide* for more information.

Tip: All items that appear in the TYPE=MAIN portion of the C1DEFLTS Table are explained in Appendix A, "The Endevor Defaults Table" in the *Installation Guide*. Refer to this appendix as necessary when creating your own C1DEFLTS Table.

Parameter	Definition
ACMROOT (Required if using ACM)	The data set name of the VSAM file your site will use to store the name of each Endevor element and all its related components. This data set is required if your site is licensed to use ACM. The data set name is supplied during the definition of the ACM Query data sets (see "Allocating and Initializing ACM Query Files" in this chapter for details). Be sure you use this same data set name when allocating these data sets in job SMPLACMD. If you are **not** using ACM, remove the dataset name; for example: **ACMROOT=,** See the *Automated Configuration Option Guide* for more information.

Parameter	Definition
ACMXREF (Required if using ACM)	The data set name of the VSAM file you will use to store the name of each component relationship. This data set is required if your site is licensed to use ACM. The data set name is supplied during the definition of the ACM Query data sets (see "Allocating and Initializing ACM Query Files" in this chapter for details). Be sure you use this same data set name when allocating these data sets in jon SMPLACMD. If you are **not** using ACM, remove the dataset name; for example: **ACMXREF=,** See the *Automated Configuration Option Guide* for more information.
ASCM	A value, Y or N, indicating whether your site is licensed for the Automated Configuration Option (ACM) facility. If you received an LMP key for Product AY, ACM, specify **Y**. Otherwise, specify **N**.
CUNAME	The 1-50 character name that describes your site. This name is used in report headings.
ELMCATL	The 1-44 character data set name for the Element Catalog. The name is supplied during the definition of the Element Catalog (see the section "Allocating the Master Control Files, Element Catalog, Base, Delta, and Output Libraries" later in this chapter). Be sure to use the same name when defining the Element Catalog in SMPLJOB2. Endevor incorporates the use of an Element Catalog file to support long element names and to boost performance by reducing the volume of I/O operations.

Parameter	Definition
LIBENV	Do not change the value of LIBENV until after running SMPLJOB4, regardless of whether you received an LMP key for this feature.
MACDSN	The name of the SOURCE library at your site that contains the Endevor macros (iprfx.iqual.SOURCE).
PKGDSN (Required entry if using packages)	The name of the package file for your site. The name is supplied during the definition of the package file (see the section "Defining and Allocating Endevor Libraries," later in this chapter for details). Be sure to use this same data set name when allocating the package data set in SMPLJOB1 (see "Run SMPLJOB1").
PROC	A value, Y or N, indicating whether your site is licensed for Endevor Extended Processors. If your site received an LMP key for Product A9, Extended Processors, specify **Y**. Otherwise, specify **N**.
SITEID	The 1-character name that identifies your site. This field is used internally to differentiate between sites. The default is 0 (zero). **Note:** The SITEID is an integral part of the Endevor footprint. Any changes to this parameter for existing Endevor applications will result in a footprint-compromised error for each element within that environment.
VIOUNIT	The unit name for temporary disk data sets that are stored on a virtual I/O unit.
WRKUNIT	The unit name for temporary disk data sets that are **not** stored on a virtual I/O unit.

The TYPE=ENVRNMNT Macro

One TYPE=ENVRNMNT section of the sample application's Defaults Table appears below.

Note: For the purposes of the sample application, you **cannot** change anything in the TYPE=ENVRNMNT parameter. Verify that the qualifiers for the Master Control File (MCF) data sets match those specified during the global edits. The data set name is supplied during the definition of the sample application data sets (see "Define and Allocate the Endevor Libraries" in this chapter for details). Be sure you use the same data set names when allocating these data sets in job SMPLJOB2 (see "Run SMPLJOB2").

```
*-----------------------------------------------------------------------*
* DEFINITION FOR ENVIRONMENT SMPLTEST
*  STAGE 1 ID: T
*  STAGE 1 NAME: TEST
*  STAGE 2 ID: Q
*  STAGE 2 NAME: QA
*-----------------------------------------------------------------------*
Col            Col                                                   Col
1              16                                                     72
↓              ↓                                                       ↓
         C1DEFLTS TYPE=ENVRNMNT,                                       X
               ENDBACT=N,               E/MVS DB BRIDGE OPTION (Y/N)   X
               ENDBAVL=N,               E/MVS DB BRIDGE OPTION (Y/N)   X
               ENVNAME='SMPLTEST',      ENVIRONMENT NAME (8 CHAR)      X
               ENVTITL='SAMPLE TEST ENVIRONMENT', TITLE (40 CHAR)      X
               JRNLGRP=,                 PITR JOURNAL GROUP ID         X
               NEXTENV=(SMPLPROD,P),    NEXT ENV/STG ID IN MAP (8,1)   X
               RSCETBL=,                RESOURCE SECURITY TABLE NAME   X
               SMFACT=N,                SMF ACTIVITY OPTION (Y/N)      X
               SMFSEC=N,                SMF SECURITY OPTION (Y/N)      X
               STG1ID='T',                                             X
               STG1NME='TEST',                                         X
               STG1TTL='UNIT TEST',                                    X
               STG1VSM='iprfx.iqual.SMPLTEST.MCF',                     X
               STG2ID='Q',                                             X
               STG2NME='QA',                                           X
               STG2TTL='QUALITY ASSURANCE',                            X
               STG2VSM='iprfx.iqual.SMPLQA.MCF',                       X
               USERTBL=
```

```
*-------------------------------------------------------------------*
* DEFINITION FOR ENVIRONMENT SMPLPROD
*  STAGE 1 ID: E
*  STAGE 1 NAME: EMER
*  STAGE 2 ID: P
*  STAGE 2 NAME: PROD
*-------------------------------------------------------------------*
Col            Col                                                Col
1              16                                                  72
↓               ↓                                                  ↓
         C1DEFLTS TYPE=ENVRNMNT,                                    X
               ENDBACT=N,               E/MVS DB BRIDGE OPTION (Y/N)  X
               ENDBAVL=N,               E/MVS DB BRIDGE OPTION (Y/N)  X
               ENVNAME='SMPLPROD',      ENVIRONMENT NAME (8 CHAR)     X
               ENVTITL='SAMPLE PROD ENVIRONMENT', TITLE (40 CHAR)     X
               JRNLGRP=,                 PITR JOURNAL GROUP ID        X
               NEXTENV=, NEXT ENV/STG ID IN MAP (8,1)                 X
               RSCETBL=,                RESOURCE SECURITY TABLE NAME  X
               SMFACT=N,                SMF ACTIVITY OPTION (Y/N)     X
               SMFSEC=N,                SMF SECURITY OPTION (Y/N)     X
               STG1ID='E',                                            X
               STG1NME='EMER',                                        X
               STG1TTL='EMERGENCY FIXES',                             X
               STG1VSM='iprfx.iqual.SMPLEMER.MCF',                    X
               STG2ID='P',                                            X
               STG2NME='PROD',                                        X
               STG2TTL='PRODUCTION',                                  X
               STG2VSM='iprfx.iqual.SMPLPROD.MCF',                    X
               USERTBL=
```

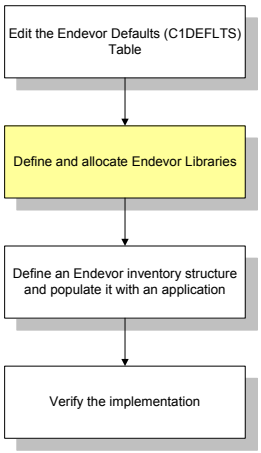
TYPE=END Macro

This indicates the end of the C1DEFLTS Table.

Run SMPLDEFT

After you have edited the sample defaults table, you need to assemble and link-edit it. If you have not already done so, copy your JOBCARD member to the beginning of member SMPLDEFT in iprfx.iqual.JCLLIB, then submit the job for execution. Store the load module in your authorized library as member C1DEFLTS. After you have implemented and worked with the sample application and you are ready to build your own life cycle, use member BC1JDEFT.

Defining and Allocating Endevor Libraries



Next, you will define and allocate the following libraries for the sample application:

- The Package File
- The Master Control File
- The Element Catalog
- The Base, Delta, and Output Libraries

Members SMPLJOB1 and SMPLJOB2, which are stored in iprfx.iqual.JCLLIB, are used to allocate these files.

Important! The VSAM files used by Endevor, the Master Control files, the Element Catalog and the Package File must be maintained at a two-level index for file integrity purposes.

Endevor automatically recognizes if a file has less than two index levels and dynamically adjusts it. The next user to access Endevor will not encounter a similar problem.

Allocating the PACKAGE FILE

The PACKAGE FILE is a VSAM data set used to store all packages built to perform element actions in any of the environments defined in the C1DEFLTS Table. There is only one package data set per site. It is defined in the C1DEFLTS Table. Be sure the name of the PACKAGE FILE allocated is the same as the name provided in the PKGDSN parameter in the C1DEFLTS Table (see the section “The TYPE=MAIN Macro” earlier in this chapter).

Specifically, this step allocates the following VSAM structure.

Library	Data Set Name
VSAM cluster	iprfx.iqual.SMPL.PACKAGE
VSAM cluster containing data	iprfx.iqual.SMPL.PACKAGE.DATA
VSAM cluster containing indexes	iprfx.iqual.SMPL.PACKAGE.INDEX

Run SMPLJOB1

If you have not done so already, copy your JOBCARD member to the beginning of SMPLJOB1, then submit the job for execution.

The table below details each step of SMPLJOB1.

In the JCL . . .	Does This . . .
Step 1	Executes IDCAMS to delete any existing package files the first time you run this job.
Step 2	Defines the (new) package file.

```
//*(JOBCARD)
//*-------------------------------------------------------------------*
//*                                                                   *
//*   (C) 2002 COMPUTER ASSOCIATES INTERNATIONAL, INC. (CA)           *
//*                                                                   *
//* NAME: SMPLJOB1                                                    *
//*                                                                   *
//* FUNCTION: TO ALLOCATE THE SITE PACKAGE FILE, AFTER OPTIONALLY     *
//*           DELETING THE EXISTING PACKAGE FILE                      *
//*                                                                   *
//*-------------------------------------------------------------------*
//*                                                                   *
//* STEP 1: DELETE THE EXISTING PACKAGE FILE                          *
//*                                                                   *
//* NOTE: THIS STEP HAS BEEN COMMENTED OUT TO PREVENT INADVERTENTLY   *
//* DELETING A PRODUCTION PACKAGE FILE.  IF YOU NEED TO DELETE THE    *
//* PACKAGE FILE, SIMPLY UNCOMMENT EACH JCL STATEMENT, INCLUDING      *
//* THE 'DELETE' STATEMENT.  THE DELETE STATEMENT IS UNCOMMENTED BY   *
//* REMOVING THE "//*" CHARACTERS.                                    *
//*                                                                   *
//*-------------------------------------------------------------------*
//*STEP1    EXEC PGM=IDCAMS
//*SYSPRINT DD SYSOUT=*
//*SYSIN    DD  *
//* DELETE 'iprfx.iqual.SMPL.PACKAGE' PURGE
//*-------------------------------------------------------------------*
//*                                                                   *
//* STEP 2: ALLOCATE THE PACKAGE FILE                                 *
//*                                                                   *
//*-------------------------------------------------------------------*
//STEP2    EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD  *
 DEFINE CLUSTER (NAME('iprfx.iqual.SMPL.PACKAGE')   -
   SPEED -
   UNIQUE -
   FREESPACE(30 30) -
   CYLINDERS(1 4) -
   VOLUMES(VVOLSER) -
   RECORDSIZE(640 3070) KEYS(64 8) SHR(3 3)) -
  DATA  (NAME('iprfx.iqual.SMPL.PACKAGE.DATA')
  INDEX (NAME('iprfx.iqual.SMPL.PACKAGE.INDEX')
/*
```

Allocating the Master Control Files, Element Catalog, Base, Delta, and Output Libraries

SMPLJOB2 defines the Master Control Files (MCF), the Element Catalog, and any base, delta, and output libraries used by the sample application. It deletes any existing libraries before allocating them.

An Endevor environment is made up of two stages; a unique MCF is required for each. The MCF values are defined to Endevor via the STG1VSAM and STG2VSAM parameters in the TYPE=ENVRNMNT section of the C1DEFLTS Table. The MCF stores the system, subsystem, type, and element definitions for the stage. Be sure the "iprfx.iqual" components of the Master Control File data set names are the same values as the dataset names coded in the C1DEFLTS table.

Endevor incorporates the use of an Element Catalog file to support long element names and to boost performance by reducing the volume of I/O operations. It is identified to Endevor via the ELMCATL parameter in the TYPE=MAIN section of the C1DEFLTS Table.

The base, delta, and output libraries are used during Endevor execution. Base libraries store the original or current source version of an element. Delta libraries track the differences between the numerous source levels of an element. Source output libraries contain readable copies of an Endevor element; for example, JCL, PROC, COPYBOOK, and data parms. Processor output libraries contain executables; for example, loadlibs, object decks, and DBRMs. Endevor uses Processor loadlibs to store the executable version of an Endevor processor. For more information on base and delta libraries, see the *Administrator Guide*. For more information on processors, see the *Extended Processor Guide*.

You can have multiple occurrences of the libraries listed below, usually one set of libraries for each stage. Only two processor load libraries are allocated - one for each production stage where type "process" is defined.

The base, delta, and output libraries that can be allocated include (but are not limited to) the following:

- Source (base) library
- Delta library
- Copy library
- Object library
- Load library
- Listing library
- Link-edit control card library
- JCL library
- JCL procedure library
- DBRM library (for DB2 users)

Run SMPLJOB2

SMPLJOB2 is found in the Endevor JCL library, iprfx.iqual.JCLLIB. If you have not already done so, copy your JOBCARD member to the beginning of SMPLJOB2, then submit the job for execution.

If the job is interrupted or terminates abnormally, restart the job from the beginning.

The table below details each step in job SMPLJOB2.

In the JCL . . .	Does This . . .
DELMCF	Executes IDCAMS to delete any Master Control Files that exist.
DEFMCF	Executes IDCAMS to define the (new) Master Control Files.
DELECATL	Executes IDCAMS to delete the sample Element Catalog, if one exists.

In the JCL . . .	Does This . . .
DELEIX	Executes IDCAMS to delete the Endevor element catalog cross-reference file.
DEFECATL	Executes IDCAMS to define the (new) Sample Element Catalog.
DEFEIX	Executes IDCAMS to define the Endevor element catalog cross-reference file.
DELPDS	Executes IDCAMS to delete the non-VSAM (base, delta, and output) libraries before they are reallocated.
DEFPDS	Executes IEFBR14 to allocate the non-VSAM (base, delta, and output) libraries.

```
//*(JOBCARD)
//*-------------------------------------------------------------------*
//*                                                                   *
//*   (C) 1987,2002 COMPUTER ASSOCIATES INTERNATIONAL, INC.           *
//*                                                                   *
//*  NAME: SMPLJOB2                                                   *
//*                                                                   *
//* PURPOSE: DELETE AND REALLOCATE THE MASTER CONTROL FILES           *
//*       AND DELETE AND REALLOCATE THE ELEMENT CATALOG USED BY       *
//*       THE SAMPLE APPLICATION.                                     *
//*                                                                   *
//*  CAUTION: THIS JOB DELETE ALL DEFINITIONS YOU HAVE ADDED          *
//*  SINCE THE LAST INSTALL OR UNLOAD OF THE SAMPLE APPLICATION.      *
//*                                                                   *
//*-------------------------------------------------------------------*
//*                                                                   *
//* NOTES:                                                            *
//* (1) IF THE JOB IS INTERRUPTED OR ABNORMALLY TERMINATES, BEGIN THE *
//*     JOB FROM DELMCF                                               *
//*                                                                   *
//* (2) STEPS DELMCF AND DELECATL MAY END WITH RC=8.  THIS IS NORMAL  *
//*     IF THE SAMPLE APPLICATION HAS NOT BEEN PREVIOUSLY INSTALLED.  *
//*                                                                   *
//*-------------------------------------------------------------------*
//*                                                                   *
//* DELMCF: DELETE THE EXISTING MASTER CONTROL FILES (MCF) FOR THE    *
//*   SMPLTEST AND SMPLPROD ENVIRONMENTS
//*                                                                   *
//* NOTE: THIS STEP MAY END WITH A CONDITION CODE OF EIGHT.  THIS IS  *
//*   NORMAL IF THE MCFS DID NOT EXIST.                               *
//*                                                                   *
//*-------------------------------------------------------------------*
//DELMCF   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
```

```
 /*                                                  */
 /*     DELETE PREVOUSLY EXISTING MCF FILES          */
 /*                                                  */
 DELETE 'IPRFX.IQUAL.SMPLTEST.MCF' CLUSTER PURGE
 DELETE 'IPRFX.IQUAL.SMPLQA.MCF'   CLUSTER PURGE
 DELETE 'IPRFX.IQUAL.SMPLEMER.MCF' CLUSTER PURGE
 DELETE 'IPRFX.IQUAL.SMPLPROD.MCF' CLUSTER PURGE
 SET MAXCC=0
//*-------------------------------------------------------------------*
//*                                                                   *
//* DEFMCF: ALLOCATE THE MASTER CONTROL FILES (MCF) FOR THE           *
//*   SMPLTEST AND SMPLPROD ENVIRONMENTS                              *
//*                                                                   *
//*-------------------------------------------------------------------*
//DEFMCF   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 /*                                                  */
 /*     DEFINE STAGE 1 SMPLTEST MCF                  */
 /*                                                  */
 DEFINE CLUSTER (NAME('IPRFX.IQUAL.SMPLTEST.MCF') -
        SPEED                  -
        UNIQUE                 -
        FREESPACE(30 30)       -
        VOLUMES(VVOLSER)       -
        CYLINDERS(2 2)         -
        RECORDSIZE(640 1200)   -
        KEYS(28 0)             -
        SHR(3 3))              -
  DATA  (NAME('IPRFX.IQUAL.SMPLTEST.MCF.DATA') CISZ(8192)) -
  INDEX (NAME('IPRFX.IQUAL.SMPLTEST.MCF.INDEX') CISZ(2048))
 /*                                                  */
 /*     DEFINE STAGE 2 SMPLTEST MCF                  */
 /*                                                  */
 DEFINE CLUSTER (NAME('IPRFX.IQUAL.SMPLQA.MCF') -
        SPEED                  -
        UNIQUE                 -
        FREESPACE(30 30)       -
        VOLUMES(VVOLSER)       -
        CYLINDERS(2 2)         -
        RECORDSIZE(640 1200)   -
        KEYS(28 0)             -
        SHR(3 3))              -
  DATA  (NAME('IPRFX.IQUAL.SMPLQA.MCF.DATA') CISZ(8192)) -
  INDEX (NAME('IPRFX.IQUAL.SMPLQA.MCF.INDEX') CISZ(2048))
 /*                                                  */
 /*     DEFINE STAGE 1 SMPLPROD MCF                  */
 /*                                                  */
 DEFINE CLUSTER (NAME('IPRFX.IQUAL.SMPLEMER.MCF') -
        SPEED                  -
        UNIQUE                 -
        FREESPACE(30 30)       -
        VOLUMES(VVOLSER)       -
        CYLINDERS(2 2)         -
        RECORDSIZE(640 1200)   -
        KEYS(28 0)             -
        SHR(3 3))              -
```

```
 DATA  (NAME('IPRFX.IQUAL.SMPLEMER.MCF.DATA') CISZ(8192)) -
 INDEX (NAME('IPRFX.IQUAL.SMPLEMER.MCF.INDEX') CISZ(2048))
/*                                                  */
/*     DEFINE STAGE 2 SMPLPROD MCF                  */
/*                                                  */
DEFINE CLUSTER (NAME('IPRFX.IQUAL.SMPLPROD.MCF') -
       SPEED                  -
       UNIQUE                 -
       FREESPACE(30 30)       -
       VOLUMES(VVOLSER)       -
       CYLINDERS(2 2)         -
       RECORDSIZE(640 1200)   -
       KEYS(28 0)             -
       SHR(3 3))              -
 DATA  (NAME('IPRFX.IQUAL.SMPLPROD.MCF.DATA') CISZ(8192)) -
 INDEX (NAME('IPRFX.IQUAL.SMPLPROD.MCF.INDEX') CISZ(2048))
//*********************************************************************
//*  THE NEXT FEW STEPS WILL DEFINE THE ELEMENT CATALOG & THE         *
//*               CATALOG CROSS-REFERENCE FILES.                      *
//*-------------------------------------------------------------------*
//*-------------------------------------------------------------------*
//* DELECATL:  DELETE THE EXISTING ELEMENT CATALOG (ELMCATL) FOR THE  *
//*   SAMPLE ALLOCATION                                               *
//*                                                                   *
//* NOTE: THIS STEP MAY END WITH A RC=8.                              *
//*  THIS IS NORMAL IF THE ELEMENT CATALOG DID NOT EXIST              *
//*-------------------------------------------------------------------*
//DELECATL EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 /*    DELETE PREVOUSLY EXISTING ELEMENT CATALOG  */
 DELETE 'IPRFX.IQUAL.SMPL.ELMCATL' CLUSTER PURGE
 SET MAXCC=0
//*-------------------------------------------------------------------*
//* DELETE  THE ENDEVOR ELEMENT CATALOG CROSS-REFERENCE FILE
//*-------------------------------------------------------------------*
//DELEIX EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 /*    DELETE PREVOUSLY EXISTING ELEMENT CATALOG  CROSS REF FILE */
 DELETE 'IPRFX.IQUAL.SMPL.ELMCATL.EINDEX' PURGE
 SET MAXCC=0
//*-------------------------------------------------------------------*
//*                                                                   *
//* DEFECATL: ALLOC THE ELEMENT CATALOG FOR THE SAMPLE APPLICATION    *
//*                                                                   *
//*-------------------------------------------------------------------*
//DEFECATL EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 /*     DEFINE ELEMENT CATALOG                      */
 DEFINE CLUSTER (NAME('IPRFX.IQUAL.SMPL.ELMCATL') -
                 SPEED -
                 UNIQUE -
                 SHR(3 3) -
                 FREESPACE(30 30) -
                 CYLINDERS(2 2) -
```

```
                 VOLUMES(VVOLSER) -
                 RECORDSIZE(640 1017) KEY(255 0)) -
        DATA  (NAME('IPRFX.IQUAL.SMPL.ELMCATL.DATA') CISZ(8192)) -
        INDEX (NAME('IPRFX.IQUAL.SMPL.ELMCATL.INDEX') CISZ(2048))
//*-------------------------------------------------------------------*
//* DEFEIX-ALLOCATE THE ENDEVOR ELEMENT CATALOG CROSS REF FILE
//*-------------------------------------------------------------------*
//DEFEIX EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
 DEFINE CLUSTER (NAME('IPRFX.IQUAL.SMPL.ELMCATL.EINDEX') -
                SPEED -
                UNIQUE -
                SHR(3 3) -
                FREESPACE(30 30) -
                CYLINDERS(2 2) -
                VOLUMES(VVOLSER) -
                RECORDSIZE(287 400) KEY(10 0) ) -
     DATA  (NAME('IPRFX.IQUAL.SMPL.ELMCATL.EINDEX.DATA') CISZ(8192)) -
     INDEX (NAME('IPRFX.IQUAL.SMPL.ELMCATL.EINDEX.INDEX') CISZ(2048))
//*-------------------------------------------------------------------*
//*DELPDS: DELETE THE BASE, DELTA, SOURCE OUTPUT AND OTHER LIBRARIES  *
//*   USED BY THE SMPLTEST AND SMPLPROD ENVIRONMENTS.                 *
//*-------------------------------------------------------------------*
//DELPDS   EXEC PGM=IDCAMS
//SYSPRINT DD SYSOUT=*
//SYSIN    DD *
     /*-----------------------------------------------------*/
     /* DELETE SMPLTEST STAGE TEST LIBRARIES                */
     /*-----------------------------------------------------*/
  DELETE  'IPRFX.IQUAL.SMPLTEST.BASE'
  DELETE  'IPRFX.IQUAL.SMPLTEST.DELTA'
  DELETE  'IPRFX.IQUAL.SMPLTEST.COPYLIB'
  DELETE  'IPRFX.IQUAL.SMPLTEST.LISTLIB'
  DELETE  'IPRFX.IQUAL.SMPLTEST.LOADLIB'
  DELETE  'IPRFX.IQUAL.SMPLTEST.OBJLIB'
  DELETE  'IPRFX.IQUAL.SMPLTEST.PARMLIB'
  DELETE  'IPRFX.IQUAL.SMPLTEST.SRCLIB'
  DELETE  'IPRFX.IQUAL.SMPLTEST.MACLIB'
     /*-----------------------------------------------------*/
     /* DELETE SMPLTEST STAGE QA LIBRARIES                  */
     /*-----------------------------------------------------*/
  DELETE  'IPRFX.IQUAL.SMPLQA.BASE'
  DELETE  'IPRFX.IQUAL.SMPLQA.DELTA'
  DELETE  'IPRFX.IQUAL.SMPLQA.COPYLIB'
  DELETE  'IPRFX.IQUAL.SMPLQA.LISTLIB'
  DELETE  'IPRFX.IQUAL.SMPLQA.LOADLIB'
  DELETE  'IPRFX.IQUAL.SMPLQA.OBJLIB'
  DELETE  'IPRFX.IQUAL.SMPLQA.PARMLIB'
  DELETE  'IPRFX.IQUAL.SMPLQA.SRCLIB'
  DELETE  'IPRFX.IQUAL.SMPLQA.MACLIB'
     /*-----------------------------------------------------*/
     /* DELETE SMPLPROD STAGE EMER LIBRARIES                */
     /*-----------------------------------------------------*/
  DELETE  'IPRFX.IQUAL.SMPLEMER.BASE'
  DELETE  'IPRFX.IQUAL.SMPLEMER.DELTA'
  DELETE  'IPRFX.IQUAL.SMPLEMER.COPYLIB'
```

```
  DELETE  'IPRFX.IQUAL.SMPLEMER.LISTLIB'
  DELETE  'IPRFX.IQUAL.SMPLEMER.LOADLIB'
  DELETE  'IPRFX.IQUAL.SMPLEMER.OBJLIB'
  DELETE  'IPRFX.IQUAL.SMPLEMER.PARMLIB'
  DELETE  'IPRFX.IQUAL.SMPLEMER.PRCSLOAD'
  DELETE  'IPRFX.IQUAL.SMPLEMER.PRCSLIST'
  DELETE  'IPRFX.IQUAL.SMPLEMER.SRCLIB'
  DELETE  'IPRFX.IQUAL.SMPLEMER.MACLIB'
     /*-----------------------------------------------------*/
     /* DELETE SMPLPROD STAGE PROD LIBRARIES                */
     /*-----------------------------------------------------*/
  DELETE  'IPRFX.IQUAL.SMPLPROD.BASE'
  DELETE  'IPRFX.IQUAL.SMPLPROD.DELTA'
  DELETE  'IPRFX.IQUAL.SMPLPROD.COPYLIB'
  DELETE  'IPRFX.IQUAL.SMPLPROD.LISTLIB'
  DELETE  'IPRFX.IQUAL.SMPLPROD.LOADLIB'
  DELETE  'IPRFX.IQUAL.SMPLPROD.OBJLIB'
  DELETE  'IPRFX.IQUAL.SMPLPROD.PARMLIB'
  DELETE  'IPRFX.IQUAL.SMPLPROD.PRCSLOAD'
  DELETE  'IPRFX.IQUAL.SMPLPROD.PRCSLIST'
  DELETE  'IPRFX.IQUAL.SMPLPROD.SRCLIB'
  DELETE  'IPRFX.IQUAL.SMPLPROD.MACLIB'
     /*-----------------------------------------------------*/
     /* DELETE THE DB2 LIBRARIES                            */
     /*-----------------------------------------------------*/
  DELETE  'IPRFX.IQUAL.SMPLTEST.DBRMLIB'
  DELETE  'IPRFX.IQUAL.SMPLQA.DBRMLIB'
  DELETE  'IPRFX.IQUAL.SMPLEMER.DBRMLIB'
  DELETE  'IPRFX.IQUAL.SMPLPROD.DBRMLIB'
  SET MAXCC=0
//*-------------------------------------------------------------------*
//*DEFPDS: ALLOCATE THE BASE, DELTA, SOURCE OUTPUT AND OTHER LIBRARIES*
//*   THAT ARE USED BY THE SMPLTEST AND SMPLPROD ENVIRONMENTS.        *
//*-------------------------------------------------------------------*
//DEFPDS   EXEC PGM=IEFBR14
//*******************************************************************
//*  ALLOCATE SMPLTEST STAGE 1 (TEST) LIBRARIES (EXCEPT DB2)
//*******************************************************************
//BASE1    DD DSN=IPRFX.IQUAL.SMPLTEST.BASE,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=VB,LRECL=516,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(16,16,50))
//DELTA1   DD DSN=IPRFX.IQUAL.SMPLTEST.DELTA,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=VB,LRECL=259,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(210,210,210))
//COPYLIB1 DD DSN=IPRFX.IQUAL.SMPLTEST.COPYLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(35,35,45))
//LISTLIB1 DD DSN=IPRFX.IQUAL.SMPLTEST.LISTLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=VBA,LRECL=259,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
```

```
//            SPACE=(TRK,(210,210,95))
//LOADLIB1 DD DSN=IPRFX.IQUAL.SMPLTEST.LOADLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=U,BLKSIZE=32760,LRECL=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(CYL,(5,15,95))
//OBJLIB1  DD DSN=IPRFX.IQUAL.SMPLTEST.OBJLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(5,5,25))
//PARMLIB1 DD DSN=IPRFX.IQUAL.SMPLTEST.PARMLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(5,5,15))
//SRCLIB1  DD DSN=IPRFX.IQUAL.SMPLTEST.SRCLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(60,60,45))
//MACLIB1  DD DSN=IPRFX.IQUAL.SMPLTEST.MACLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(5,5,5))
//*******************************************************************
//*  ALLOCATE SMPLTEST STAGE 2 (QA) LIBRARIES (EXCEPT DB2)
//*******************************************************************
//BASE2    DD DSN=IPRFX.IQUAL.SMPLQA.BASE,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=VB,LRECL=516,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(16,16,50))
//DELTA2   DD DSN=IPRFX.IQUAL.SMPLQA.DELTA,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=VB,LRECL=259,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(210,210,180))
//COPYLIB2 DD DSN=IPRFX.IQUAL.SMPLQA.COPYLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(35,35,95))
//LISTLIB2 DD DSN=IPRFX.IQUAL.SMPLQA.LISTLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=VBA,LRECL=259,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(210,210,95))
//LOADLIB2 DD DSN=IPRFX.IQUAL.SMPLQA.LOADLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=U,BLKSIZE=32760,LRECL=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(210,210,45))
//OBJLIB2  DD DSN=IPRFX.IQUAL.SMPLQA.OBJLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
```

```
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(5,5,25))
//PARMLIB2 DD DSN=IPRFX.IQUAL.SMPLQA.PARMLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(5,5,15))
//SRCLIB2  DD DSN=IPRFX.IQUAL.SMPLQA.SRCLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(210,210,45))
//MACLIB2  DD DSN=IPRFX.IQUAL.SMPLQA.MACLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(10,10,45))
//*******************************************************************
//*  ALLOCATE SMPLPROD/STAGE 1 (EMER) LIBRARIES (EXCEPT DB2)
//*******************************************************************
//BASE3    DD DSN=IPRFX.IQUAL.SMPLEMER.BASE,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=VB,LRECL=516,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(16,16,20))
//DELTA3   DD DSN=IPRFX.IQUAL.SMPLEMER.DELTA,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=VB,LRECL=259,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(210,210,210))
//COPYLIB3 DD DSN=IPRFX.IQUAL.SMPLEMER.COPYLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(35,35,45))
//LISTLIB3 DD DSN=IPRFX.IQUAL.SMPLEMER.LISTLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=VBA,LRECL=259,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(210,210,95))
//LOADLIB3 DD DSN=IPRFX.IQUAL.SMPLEMER.LOADLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=U,BLKSIZE=32760,LRECL=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(CYL,(20,20,95))
//OBJLIB3  DD DSN=IPRFX.IQUAL.SMPLEMER.OBJLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(5,5,25))
//PARMLIB3 DD DSN=IPRFX.IQUAL.SMPLEMER.PARMLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(5,5,25))
//PRCSLOD3 DD DSN=IPRFX.IQUAL.SMPLEMER.PRCSLOAD,
//            DISP=(NEW,CATLG,DELETE),
```

```
//            DCB=(RECFM=U,LRECL=0,BLKSIZE=32760),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(CYL,(45,45,45))
//PRCSLST3 DD DSN=IPRFX.IQUAL.SMPLEMER.PRCSLIST,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=VBA,LRECL=259,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(CYL,(30,15,30))
//SRCLIB3  DD DSN=IPRFX.IQUAL.SMPLEMER.SRCLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(210,210,45))
//MACLIB3  DD DSN=IPRFX.IQUAL.SMPLEMER.MACLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(210,210,45))
//*******************************************************************
//*  ALLOCATE SMPLPROD STAGE2 (PROD) LIBRARIES (EXCEPT DB2)
//*******************************************************************
//BASE4    DD DSN=IPRFX.IQUAL.SMPLPROD.BASE,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=VB,LRECL=516,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(16,16,20))
//COPYLIB4 DD DSN=IPRFX.IQUAL.SMPLPROD.COPYLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(35,35,45))
//DELTA4   DD DSN=IPRFX.IQUAL.SMPLPROD.DELTA,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=VB,LRECL=259,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(210,210,180))
//LISTLIB4 DD DSN=IPRFX.IQUAL.SMPLPROD.LISTLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=VBA,LRECL=259,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(210,210,45))
//LOADLIB4 DD DSN=IPRFX.IQUAL.SMPLPROD.LOADLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=U,BLKSIZE=32760,LRECL=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(CYL,(210,210,45))
//OBJLIB4  DD DSN=IPRFX.IQUAL.SMPLPROD.OBJLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(65,65,45))
//PARMLIB4 DD DSN=IPRFX.IQUAL.SMPLPROD.PARMLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(14,14,45))
//PRCSLOD4 DD DSN=IPRFX.IQUAL.SMPLPROD.PRCSLOAD,
```

```
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=U,LRECL=0,BLKSIZE=32760),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(CYL,(15,30,60))
//PRCSLST4 DD DSN=IPRFX.IQUAL.SMPLPROD.PRCSLIST,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=VBA,LRECL=259,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(CYL,(15,30,16))
//SRCLIB4  DD DSN=IPRFX.IQUAL.SMPLPROD.SRCLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(210,210,95))
//MACLIB4  DD DSN=IPRFX.IQUAL.SMPLPROD.MACLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(10,10,45))
//*******************************************************************
//*  ALLOCATE DB2 ONLY LIBRARIES (ALL FOUR STAGES)
//*******************************************************************
//DBRMLIB1 DD DSN=IPRFX.IQUAL.SMPLTEST.DBRMLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(14,14,45))
//DBRMLIB2 DD DSN=IPRFX.IQUAL.SMPLQA.DBRMLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(14,14,45))
//DBRMLIB3 DD DSN=IPRFX.IQUAL.SMPLEMER.DBRMLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(14,14,45))
//DBRMLIB4 DD DSN=IPRFX.IQUAL.SMPLPROD.DBRMLIB,
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=0),
//            UNIT=PDISK,VOL=SER=DVOLSER,
//            SPACE=(TRK,(14,14,45))
```

Allocating and Initializing ACM Query Files (SMPLACMD)

Next, you will define and initialize the sample application ACM Root (ACMROOT) and Cross-reference (ACMXREF) files used by the ACM Query Facility (ACMQ). This step is required if you specified **ASCM=Y**, indicating that you are using the Automated Configuration Manager at your site. Use member SMPLACMD, located in iprfx.iqual.JCLLIB, to define the Root and Cross-reference data sets.

Note: If your site does not have a license for ACM, skip this step and continue with "Implementation Checklist".

The ACMQ Root data set contains the name of each Endevor element used as an input component to other Endevor elements, as well as all related components. The ACMQ Cross-reference data set contains a record for each component relationship.

These files are defined to Endevor in the ACMROOT and ACMXREF parameters in the TYPE=MAIN section of the C1DEFLTS Table. There is only one ACMROOT and ACMXREF pair of data sets per site.

Be sure the name of the ACMROOT and ACMXREF files allocated are the same as the names provided in the ACMROOT and ACMXREF parameters in the Endevor C1DEFLTS Table (see "Editing the TYPE=MAIN macro"):

Library	Data Set Name
VSAM cluster	ipfrx.iqual.SMPL.ACMROOT
VSAM cluster	iprfx.iqual.SMPL.ACMXREF

Refer to "Enabling Endevor ACM and the ACM Query Facility" in the *Automated Configuration Manager Guide* for more information.

Use member SMPLACMD, located in iprfx.iqual.JCLLIB, to define (create) the Root and Cross-reference data sets used by the sample application. Once you have finished using the sample application and you are ready to build your production Endevor application, use JCLLIB member BC1JACMD to build your production ACMQ files.

Edit member SMPLACMD to add a valid jobcard and verify the following variables.

Variable	Description
iprfx	Highest-level qualifier used when assigning data set names for installation and execution data sets.
iqual	Second-level qualifier used when assigning data set names for installation and execution data sets.
vvolser	Volume serial number of the disk on which the ACMROOT and ACMXREF data sets are allocated.

Follow the instructions in SMPLACMD to further tailor the JCL and submit the JOB to create and initialize the sample application Root and Cross-reference data sets.

```
*-------------------------------------------------------------------*
//*(JOBCARD)                                                        *
//*-------------------------------------------------------------------*
//*                                                                 *
//*   (C) 2002 COMPUTER ASSOCIATES INTERNATIONAL, INC. (CA)         *
//*                                                                 *
//* NAME: SMPLACMD                                                  *
//*                                                                 *
//* FUNCTION: THIS JOB IS USED TO DEFINE AND INITIALIZE THE ACM     *
//*           EXTENDED QUERY ROOT AND CROSS-REFERENCE DATA SETS.    *
//*                                                                 *
//*  1) ADD A VALID JOBCARD AT THE FRONT OF THE JOB STREAM          *
//*  2) CHANGE THE DATASET NAMES AND OTHER VARIABLES TO APPROPRIATE *
//*     VALUES FOR YOUR INSTALLATION.                               *
//*                                                                 *
//* NOTE1: THE NAMES IN THE CLUSTER DEFINES MUST MATCH THE ACMROOT  *
//*        AND ACMXREF DATA SET NAME SPECIFICATIONS IN THE C1DEFLTS *
//*        TABLE.                                                   *
//*                                                                 *
//* NOTE2: ONE CYLINDER OF ACMROOT CAN HOLD 5760 RECORDS.           *
//*        ONE CYLINDER OF ACMXREF CAN HOLD 122880 RELATIONS.       *
//*        AVOID SECONDARY EXTENTS FOR PERFORMANCE REASONS.         *
//*                                                                 *
//*-------------------------------------------------------------------*
//*
//SMPLACMD EXEC PGM=IDCAMS
//SYSPRINT DD   SYSOUT=*
//SYSIN    DD   *
 SET MAXCC=8
 DELETE 'IPRFX.IQUAL.SMPL.ACMROOT' PURGE
```

```
 DELETE 'IPRFX.IQUAL.SMPL.ACMXREF' PURGE
 SET MAXCC=0
 DEFINE CLUSTER (NAME('IPRFX.IQUAL.SMPL.ACMROOT') -
   CYLINDERS(2 2) -
   VOLUMES(VVOLSER) -
   SHAREOPTIONS(3,3) -
   LINEAR)
 DEFINE CLUSTER (NAME('IPRFX.IQUAL.SMPL.ACMXREF') -
   CYLINDERS(1 1)  -
   VOLUMES(VVOLSER) -
   SHAREOPTIONS(3,3) -
   LINEAR)
//*==============================================================*
//* INITIALIZE ACME DATASETS
//*==============================================================*
//INIT    EXEC PGM=NDVRC1,PARM='BC1PACMOMODE=OPT'
//*INIT   EXEC PGM=NDVRC1,PARM='BC1PACMOMODE=ANY'
//*   'MODE=OPT' = RUN IN OPTIMIZED MODE
//*   'MODE=ANY' = RUN IN R3.8-COMPATIBLE MODE
//*
//STEPLIB  DD DISP=SHR,DSN=UPRFX.UQUAL.AUTHLIB
//         DD DISP=SHR,DSN=IPRFX.IQUAL.AUTHLIB
//CONLIB   DD DISP=SHR,DSN=IPRFX.IQUAL.CONLIB
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//ROOT     DD DISP=SHR,DSN='IPRFX.IQUAL.SMPL.ACMROOT'
//XREF     DD DISP=SHR,DSN='IPRFX.IQUAL.SMPL.ACMXREF'
```

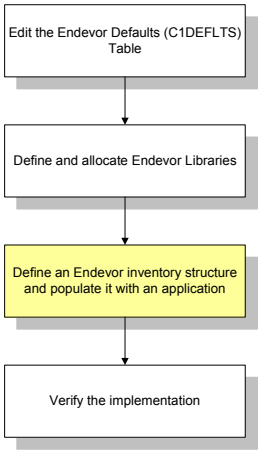
Implementation Checklist

Before implementing the rest of the sample application, you need to ensure that you have installed the Endevor software correctly. Ask yourself the following questions, which reflect typical problems that might occur during the sample install.

1. Have your LMP keys been installed?
2. Have all Endevor-supplied libraries been successfully created and populated? Do they contain the correct number of modules? (HINT: AUTHLIB 60+, CONLIB 700+) Refer to the *Installation Guide* for more information.
3. Have the MCFs, Element Catalog/EINDEX and Package File been created correctly?
4. Are TYPE=MAIN, TYPE=ENVRNMNT, and TYPE=END present and in the correct order in the Endevor C1DEFLTS Table?

5. If the AUTHLIB is not being placed in the linklist, did you add a STEPLIB to the proper JCL skeletons, LOGON procs, and CLISTs?
6. If AUTHLIB is in the linklist, did you perform an LLA REFRESH after creating the C1DEFLTS table in STEP 1?
7. Test system authorization for NDVRC1. Is the AUTHLIB authorized? Depending upon the release of TSO, has the NDVRC1 keyword been added to the proper system tables?
8. Can Endevor find and load NDVRC1, C1DEFLTS, and any other programs that typically reside in the AUTHLIB/CONLIB?

Defining the Sample Application Inventory Structure (SMPLJOB3)



Next, you will define the sample application inventory structure (systems, subsystems, types) to Endevor by executing the Batch Environment Definition Facility (also known as Batch Admin). For more information on using the Batch Environment Definition Facility, refer to the *SCL Reference Guide*. Use the JCL from SMPLJOB3, which is in the library iprfx.iqual.JCLLIB.

If you have not already done so, copy your JOBCARD member to the beginning of SMPLJOB3, then submit the job for execution.

The SMPLJOB3 JCL is shown below. Notice the last two lines in the JCL. The second-to-last line refers to member SCLXTND and is uncommented. The last line refers to member SCLBASE and is commented out.

If your site is using extended processors, do nothing. Otherwise, uncomment the SCLBASE, ddname ENESCLIN line and comment out the line specifying member SCLXTND by adding "//*" to the front of that line.

SCLXTND contains SCL to define system FINANCE, subsystems ACCTPAY and ACCTREC, and several types such as COBOL, ASM, and COPY, as well as several processor groups for each of these types. These definitions are made to both stages of the SMPLTEST and SMPLPROD environments. The FINANCE system is where the source elements used by the sample application are stored. An additional system, ADMIN, is defined to the SMPLPROD environment. This system contains one subsystem and one type – PROCESS. This is where the processors used by the sample application are stored.

SCL contains SCL similar to SCLXTEND in that it defines the same systems, subsystems, and types. It differs in the respect that no processor groups are defined.

```
//*(JOBCARD)
//*-------------------------------------------------------------------*
//*                                                                   *
//*   (C) 2002 COMPUTER ASSOCIATES INTERNATIONAL, INC. (CA)           *
//*                                                                   *
//* NAME: SMPLJOB3                                                    *
//*                                                                   *
//* PURPOSE: USES THE BATCH ENVIRONMENT ADMINISTRATION FACILITY TO    *
//*  DEFINE THE SAMPLE APPLICATION INVENTORY STRUCTURE.               *
//*                                                                   *
//*-------------------------------------------------------------------*
//ENBE1000 EXEC PGM=NDVRC1,PARM='ENBE1000',REGION=4096K
//STEPLIB  DD DISP=SHR,DSN=uprfx.uqual.AUTHLIB
//         DD DISP=SHR,DSN=iprfx.iqual.AUTHLIB
//CONLIB   DD DISP=SHR,DSN=iprfx.iqual.CONLIB
//C1MSGS1  DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SYSTERM  DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//*-------------------------------------------------------------------*
//* IF YOUR SITE DOES NOT USE ENDEVOR PROCESORS, USE MEMBER 'SCLBASE' *
//* IN PLACE OF 'SCLXTND'.                                            *
//*-------------------------------------------------------------------*
//ENESCLIN DD DISP=SHR,DSN=iprfx.iqual.SOURCE(SCLXTND)
//*ENESCLIN DD DISP=SHR,DSN=iprfx.iqual.SOURCE(SCLBASE)
//*
```

If this step does not complete satisfactorily (return code greater than 08), research the error, then rerun SMPLJOB2 before running SMPLJOB3 again.

Hint: To quickly find the error message associated with an RC=12:

- View the job's output in SDSF or another output management utility
- Page down to the C1MSGS1 output
- On the COMMAND line key: `F 'E ' 19 ALL` then press ENTER.

You should be presented with an ENBEnnnE message which is described in the *Error Codes and Messages Guide*.

Populating the Sample Application (SMPLJOB4)

Next, you will add processors and souce data to the sample application. Use the JCL member SMPLJOB4, which is in the library iprfx.iqual.JCLLIB. This job adds elements to the application and, if your site is using processors, adds the sample processors and generates the elements.

If you have not already done so, copy your JOBCARD member to the beginning of SMPLJOB4, then submit the job for execution.

If this step does not complete successfully, perform a restart at the step where termination occurred.

> **Tip:** This job may exceed the output line limit. Submit the job appropriately.

The table below details each step in job SMPLJOB4:

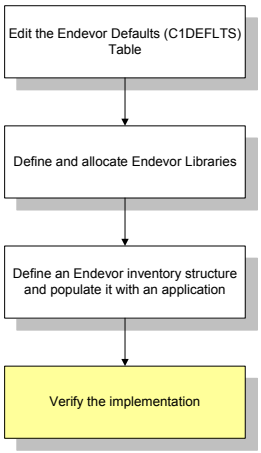
In the JCL . . .	Does This . . .
Step 1	Adds processors to the sample application, if your site is using them.
Step 2	Adds elements to the sample application
Step 3	Moves all elements to SMPLPROD stage 2.

The SMPLJOB4 JCL is shown below:

```
//*(JOBCARD)
//*-------------------------------------------------------------------*
//*                                                                   *
//*   (C) 2002 COMPUTER ASSOCIATES INTERNATIONAL, INC. (CA)           *
//*                                                                   *
//* NAME: SMPLJOB4                                                    *
//*                                                                   *
//* PURPOSE: JOB SMPLJOB4 ADDS THE SAMPLE APPLICATION PROCESSORS AND  *
//*  RESTORES THE SAMPLE APPLICATION ELEMENTS.  IT ALSO GENERATES     *
//*  THE ELEMENTS.                                                    *
//*                                                                   *
```

```
//*  ALL ELEMENTS ARE MOVED TO THE PROD STAGE OF ENVIRONMENT          *
//*  SMPLPROD.                                                        *
//*                                                                   *
//* RESTART INSTRUCTIONS:                                             *
//* - IF JOB IS INTERRUPTED OR TERMINATES ABNORMALLY, DO A RESTART    *
//*   AT THE STEP WHERE TERMINATION OCCURRED.
//*
//*-------------------------------------------------------------------*
//*******************************************************************
//* ADD AND MOVE PROCESSORS
//*******************************************************************
//STEP1    EXEC PGM=NDVRC1,REGION=4096K,DYNAMNBR=1500,
//             PARM='C1BM3000'
//STEPLIB  DD DISP=SHR,DSN=UPRFX.UQUAL.AUTHLIB
//         DD DISP=SHR,DSN=IPRFX.IQUAL.AUTHLIB
//CONLIB   DD DISP=SHR,DSN=IPRFX.IQUAL.CONLIB
//C1MSGS1  DD SYSOUT=*
//C1MSGS2  DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//BSTIPT01 DD DSN=IPRFX.IQUAL.SOURCE(ADPROC?),DISP=SHR
//*
//*******************************************************************
//* ADD AND GENERATE ELEMENTS TO STAGE 1
//*******************************************************************
//STEP2    EXEC PGM=NDVRC1,REGION=4096K,DYNAMNBR=1500,
//             PARM='C1BM3000'
//STEPLIB  DD DISP=SHR,DSN=UPRFX.UQUAL.AUTHLIB
//         DD DISP=SHR,DSN=IPRFX.IQUAL.AUTHLIB
//CONLIB   DD DISP=SHR,DSN=IPRFX.IQUAL.CONLIB
//C1MSGS1  DD SYSOUT=*
//C1MSGS2  DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//BSTIPT01 DD DSN=IPRFX.IQUAL.SOURCE(ADDELEM),DISP=SHR
//*******************************************************************
//* MOVE ELEMENTS TO SMPLPROD STAGE 2
//*******************************************************************
//STEP3    EXEC PGM=NDVRC1,REGION=4096K,DYNAMNBR=1500,
//             PARM='C1BM3000'
//STEPLIB  DD DISP=SHR,DSN=UPRFX.UQUAL.AUTHLIB
//         DD DISP=SHR,DSN=IPRFX.IQUAL.AUTHLIB
//CONLIB   DD DISP=SHR,DSN=IPRFX.IQUAL.CONLIB
//C1MSGS1  DD SYSOUT=*
//C1MSGS2  DD SYSOUT=*
//SYSABEND DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//BSTIPT01 DD DSN=IPRFX.IQUAL.SOURCE(MOVELEM),DISP=SHR
```

Verifying Implementation



Now that you have performed the installation steps and executed the sample application implementation jobs successfully, you need to verify that the sample environments have been created correctly. The procedures below summarize the verification process:

1. Be sure that the sample jobs have completed successfully. If a job has failed to execute successfully, you must rerun that job or, in some situations, restart from a step that failed within the job.
2. Bring up Endevor. Check the Endevor Environment Selection panel to ensure that the environments SMPLTEST and SMPLPROD appear.
3. Execute CONRPT02, the System Inventory Summary, as described later in this section, and review it to ensure that all sample application elements have been loaded.
4. Run foreground, batch, and ACMQ verification procedures, as described later in this section.

Checking the Environment Selection Menu

Before you begin this step, ask your system administrator how Endevor has been identified to ISPF. Typically, this is done as an ISPF menu option.

Select the Endevor option from the appropriate ISPF panel. With the sample application properly implemented, the AllFusion Endevor Environment Selection panel displays the sample environments as options.

```
-------------- AllFusion Endevor Environment Selection ------------ Row 1 of 2
Option ===> 1                                              Scroll ===> PAGE

  Select an environment to continue.  Enter the END command to exit.
  --  --------   ---------------------------------------
    1 SMPLTEST   SAMPLE TEST ENVIRONMENT
    2 SMPLPROD   SAMPLE PRODUCTION ENVIRONMENT
****************************** Bottom of data ********************************



 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Running the System Inventory Summary Report

In order to verify that all elements were correctly added to Endevor, run CONRPT02, the System Inventory Summary Report, for the SMPLPROD environment. Review the report to verify that the number of elements in SMPLPROD is correct.

To run the report, follow the steps below:

1. Select U, User Menu, from the AllFusion Endevor Primary Options Panel and press Enter.

```
--------------------AllFusion Endevor Primary Options Panel----------------------
Option ===> U

  0   DEFAULTS      - Specify Endevor ISPF default parameters
  1   DISPLAY       - Perform Display functions
  2   FOREGROUND    - Execute Foreground Actions
  3   BATCH         - Perform Batch Action processing
  4   ENVIRONMENT   - Define or Modify Environment information
  5   PACKAGE       - Perform Foreground Package processing
  6   BATCH PACKAGE - Perform Batch Package SCL Generation
  U   USER MENU     - Display user option menu
  T   TUTORIAL      - Display information about Endevor
  C   CHANGES       - Display summary of changes for this release of Endevor
  X   EXIT          - Exit the Endevor dialog



                        Current environment: SMPLPROD

    (C) 2002 Computer Associates International, Inc.


 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP      F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

2. The User Options Menu opens. Select Option 1, Reports, and press Enter.

```
-----------------------------  User Options Menu  ---------------------------
 Option ===> 1

   1  REPORTS     - Build Report Requests

   2  ACMQ        - Perform ACM Where-Used / Uses-What Queries



 Enter END to return to the Endevor Primary Options Menu
```

3. The Endevor Reporting Interface panel opens. Select option 1, Master, and press Enter.

```
------------------------- ENDEVOR REPORTING INTERFACE -------------------------
OPTION ===> 1

   1  MASTER      - Build Master Control File report JCL
   2  SMF         - Build SMF historical report JCL
   3  PACKAGE     - Build Package report JCL
   4  FOOTPRINT   - Build Footprint report JCL
   5  UNLOAD      - Build Unload/Reload report JCL
   6  SHIPMENTS   - Build Package Shipment report JCL

   E  EDIT        - Edit Report JCL
   S  SUBMIT      - Submit Report JCL for Batch execution


 Job statement information:

  ===>
  ===>
  ===>
  ===>

 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

4. The Endevor Master Control File Reports panel opens. Select 02, System Inventory Summary, type the highlighted values on the panel, and press Enter.

```
--------------------- ENDEVOR MASTER CONTROL FILE REPORTS ---------------------
COMMAND ===>02

SELECT REPORTS:
 _ 01  System inventory profile    _ 07  System definition profile
 _ 02  System inventory summary    _ 08  Element signed out profile - by system
 _ 03  Element catalog             _ 09  Element signed out profile - by user
 _ 04  Element activity profile    _ 10  Approver group definition
 _ 05  Element activity summary    _ 11  Approver group usage
 _ 06  Element catalog by CCID     _ 12  Element catalog by retrieve CCID

            FROM ENDEVOR/MVS
                 ENVIRONMENT  ===> SMPLPROD
                 SYSTEM       ===> *
                 SUBSYSTEM    ===> *
                 ELEMENT      ===> *
                 TYPE         ===> *
                 STAGE        ===> *

                 DAYS         ===>

      SEARCH ENVIRONMENT MAP  ===> N (Y/N)
 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

5. Endevor generates JCL to create CONRPT02, the System Inventory Summary, for environment SMPLPROD. The Endevor Reporting Interface panel opens. Endevor displays the message, Job Step Created, in the upper right hand corner of the panel.

```
------------------------- ENDEVOR REPORTING INTERFACE ------- JOB STEP CREATED
OPTION ===>

   1  MASTER      - Build Master Control File report JCL
   2  SMF         - Build SMF historical report JCL
   3  PACKAGE     - Build Package report JCL
   4  FOOTPRINT   - Build Footprint report JCL
   5  UNLOAD      - Build Unload/Reload report JCL
   6  SHIPMENTS   - Build Package Shipment report JCL

   E  EDIT        - Edit Report JCL
   S  SUBMIT      - Submit Report JCL for Batch execution



 Job statement information:

  ===>
  ===>
  ===>
  ===>

 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

6. Select option S, Submit, and press Enter. Endevor submits the JCL for execution and displays the message, Job Submitted, in the upper right hand corner of the panel.

```
------------------------- ENDEVOR REPORTING INTERFACE ---------- JOB SUBMITTED
OPTION ===> S

   1  MASTER      - Build Master Control File report JCL
   2  SMF         - Build SMF historical report JCL
   3  PACKAGE     - Build Package report JCL
   4  FOOTPRINT   - Build Footprint report JCL
   5  UNLOAD      - Build Unload/Reload report JCL
   6  SHIPMENTS   - Build Package Shipment report JCL

   E  EDIT        - Edit Report JCL
   S  SUBMIT      - Submit Report JCL for Batch execution



 Job statement information:

  ===>
  ===>
  ===>
  ===>

 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

Compare the output from the System Inventory Summary you generate to the sample report shown below. The values on your report should match the sample report values.

If there are any discrepancies between your report and the sample report, review the output from SMPLJOB4. Look for any error messages that might have occurred. Correct the problems, resubmit SMPLJOB4, and run CONRPT02 again.

Your System Inventory Summary should appear as follows.

```
(C) 2002 Computer Associates International, Inc (CA)         Endevor                          XX/XX/XX  XX:XX:XX     PAGE     3
                                                                                                     RELEASE X.X   SERIAL XXXXXX

                                        CONRPT02:  SYSTEM INVENTORY SUMMARY

          ENVIRON     SYSTEM      SUBSYS      TYPE        STAGE  NUMBER OF          TOTAL    AVERAGE # OF    LARGEST # OF
                                                          ID SEQ  ELEMENTS      STATEMENTS      STATEMENTS      STATEMENTS

          SMPLPROD    ADMIN       PROCESS     PROCESS     P    2         5             437              87             151
          --------    --------    --------    --------    ------  --------    ------------
          SMPLPROD    ADMIN       PROCESS     PROCESS                    5             437
          SMPLPROD    ADMIN       PROCESS                                5             437
          SMPLPROD    ADMIN                                              5             437
(C) 2002 Computer Associates International, Inc (CA)         Endevor                          XX/XX/XX  XX:XX:XX     PAGE     4
                                                                                                     RELEASE X.X   SERIAL XXXXXX

                                        CONRPT02:  SYSTEM INVENTORY SUMMARY

          ENVIRON     SYSTEM      SUBSYS      TYPE        STAGE  NUMBER OF          TOTAL    AVERAGE # OF    LARGEST # OF
                                                          ID SEQ  ELEMENTS      STATEMENTS      STATEMENTS      STATEMENTS

         SMPLPROD    FINANCE     ACCTPAY     ASM          P    2         2             103              52              83
         --------    --------    --------    --------     ------  --------    ------------
         SMPLPROD    FINANCE     ACCTPAY     ASM                        2             103
         SMPLPROD    FINANCE     ACCTPAY     COBOL        P    2         8             390              49             102
         --------    --------    --------    --------     ------  --------    ------------
         SMPLPROD    FINANCE     ACCTPAY     COBOL                      8             390
         SMPLPROD    FINANCE     ACCTPAY     COPY         P    2         4              32               8              14
         --------    --------    --------    --------     ------  --------    ------------
         SMPLPROD    FINANCE     ACCTPAY     COPY                       4              32
         SMPLPROD    FINANCE     ACCTPAY     MACRO        P    2         1              55              55              55
         --------    --------    --------    --------     ------  --------    ------------
         SMPLPROD    FINANCE     ACCTPAY                               15             580
         SMPLPROD    FINANCE     ACCTREC     COBOL        P    2         9             528              59             106
         --------    --------    --------    --------     ------  --------    ------------
         SMPLPROD    FINANCE     ACCTREC     COBOL                      9             528
         SMPLPROD    FINANCE     ACCTREC     COPY         P    2        10              81               8              16
         --------    --------    --------    --------     ------  --------    ------------
         SMPLPROD    FINANCE     ACCTREC     COPY                      10              81
         SMPLPROD    FINANCE     ACCTREC     LINKCARD     P    2         1             106             106             106
         --------    --------    --------    --------     ------  --------    ------------
         SMPLPROD    FINANCE     ACCTREC                               20             715
         SMPLPROD    FINANCE                                           35            1295
         SMPLPROD                                                      40            1732
****************************** BOTTOM OF DATA*****************************************************************************
```

If you ran SMPLJOB3 using SCLBASE as input, your report should look similiar to the following.

```
(C) 2002 Computer Associates International, Inc  (CA)        Endevor                          XX/XX/XX  XX:XX:XX     PAGE     3
                                                                                                     RELEASE X.X   SERIAL XXXXXX
                                        CONRPT02:  SYSTEM INVENTORY SUMMARY
        ENVIRON     SYSTEM      SUBSYS      TYPE        STAGE  NUMBER OF           TOTAL    AVERAGE # OF    LARGEST # OF
                                                        ID SEQ  ELEMENTS      STATEMENTS      STATEMENTS      STATEMENTS

        SMPLPROD    FINANCE     ACCTPAY     ASM         P    2         2            103              52              83
        --------    --------    --------    --------    ------   --------    ------------
        SMPLPROD    FINANCE     ACCTPAY     ASM                       2            103
        SMPLPROD    FINANCE     ACCTPAY     COBOL       P    2         8            390              49             102
        --------    --------    --------    --------    ------   --------    ------------
        SMPLPROD    FINANCE     ACCTPAY     COBOL                     8            390
        SMPLPROD    FINANCE     ACCTPAY     COPY        P    2         5             35               7              14
        --------    --------    --------    --------    ------   --------    ------------
        SMPLPROD    FINANCE     ACCTPAY     COPY                      5             35
        SMPLPROD    FINANCE     ACCTPAY     MACRO       P    2         1             55              55              55
        --------    --------    --------    --------    ------   --------    ------------
        SMPLPROD    FINANCE     ACCTPAY                              16            583
        SMPLPROD    FINANCE     ACCTREC     COBOL       P    2         9            528              59             106
        --------    --------    --------    --------    ------   --------    ------------
        SMPLPROD    FINANCE     ACCTREC     COBOL                     9            528
        SMPLPROD    FINANCE     ACCTREC     COPY        P    2        10             80               8              15
        --------    --------    --------    --------    ------   --------    ------------
        SMPLPROD    FINANCE     ACCTREC     COPY                     10             80
        SMPLPROD    FINANCE     ACCTREC                              19            608
        SMPLPROD    FINANCE                                          35           1191
        SMPLPROD                                                     35           1191
```

Performing Foreground Verification

Follow the steps below to verify, in foreground mode, that the sample environments have been loaded correctly.

1. Select option 2, Foreground, from the AllFusion Endevor Primary Options Panel and press Enter.

```
--------------------AllFusion Endevor Primary Options Panel----------------------
Option ===>2

  0   DEFAULTS      - Specify Endevor ISPF default parameters
  1   DISPLAY       - Perform Display functions
  2   FOREGROUND    - Execute Foreground Actions
  3   BATCH         - Perform Batch Action processing
  4   ENVIRONMENT   - Define or Modify Environment information
  5   PACKAGE       - Perform Foreground Package processing
  6   BATCH PACKAGE - Perform Batch Package SCL Generation
  U   USER MENU     - Display user option menu
  T   TUTORIAL      - Display information about Endevor
  C   CHANGES       - Display summary of changes for this release of Endevor
  X   EXIT          - Exit the Endevor dialog



                        Current environment: SMPLTEST

    (C) 2002 Computer Associates International, Inc.


Use the EXIT option to terminate Endevor
```

2. The Foreground Options Menu opens. Select option 4, Generate, from the Foreground Options Menu.

```
------------------------- Foreground Options Menu ---------------------------
Option ===>4

  1   DISPLAY    - Display an element
  2   ADD/UPDATE - Add or update an element into stage 1
  3   RETRIEVE   - Retrieve or copy an element
  4   GENERATE   - Execute the Generate Processor for this element
  5   MOVE       - Move an element to the next inventory location
  6   DELETE     - Delete an element
  7   PRINT      - Print elements, changes and detail change history
  8   SIGNIN     - Explicitly sign-in an element
```

3. The Generate Elements panel opens. Type the highlighted values on the panel and press Enter. Endevor compiles and links program FINARP01.

```
----------------------------- GENERATE ELEMENTS -----------------------------
OPTION  ===>G
                                ELEMENT DISPLAY OPTIONS:
   blank - Element list            S - Summary   B - Browse    H - History
   G - Generate element            M - Master    C - Changes

FROM ENDEVOR:                          ACTION OPTIONS:
  ENVIRONMENT ===> SMPLPROD               CCID             ===> SAMPLE
  SYSTEM      ===> FINANCE                COPYBACK         ===> N (Y/N)
  SUBSYSTEM   ===> ACCTREC                OVERRIDE SIGNOUT ===> N (Y/N)
  ELEMENT     ===> FINARP01               PROCESSOR GROUP  ===>
  TYPE        ===> COBOL
  STAGE       ===> P        T - TEST         Q - QA

  COMMENT     ===> VERIFY

LIST OPTIONS:
  DISPLAY LIST      ===> Y (Y/N)
  WHERE CCID EQ     ===>
  WHERE PROC GRP EQ ===>
  BUILD USING MAP   ===> Y (Y/N)
```

If the sample environments have been installed correctly, the message “ELEMENT FINARP01 GENERATED” appears in the upper right corner of the Generate Elements panel.

If the action fails, review the execution report, which is automatically generated, to determine the nature of the problem. If it fails due to errors during the compile or link steps, you will need to research the problem using the stored compile and link-edit listings. To view a stored listing:

1. From the AllFusion Endevor Primary Options Panel, select option 1, Display, and press Enter.

```
-----------------  AllFusion Endevor Primary Options Panel----------------------
Option ===> 1

  0   DEFAULTS      - Specify Endevor ISPF default parameters
  1   DISPLAY       - Perform Display functions
  2   FOREGROUND    - Execute Foreground Actions
  3   BATCH         - Perform Batch Action processing
  4   ENVIRONMENT   - Define or Modify Environment information
  5   PACKAGE       - Perform Foreground Package processing
  6   BATCH PACKAGE - Perform Batch Package SCL Generation
  U   USER MENU     - Display user option menu
  T   TUTORIAL      - Display information about Endevor
  C   CHANGES       - Display summary of changes for this release of Endevor
  X   EXIT          - Exit the Endevor dialog



                       Current environment: SMPLTEST

    (C) 2002 Computer Associates International, Inc.


 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

2. The Display Options Menu opens. Select Option 2, Footprint, and press Enter.

```
--------------------------  DISPLAY OPTIONS MENU  ---------------------------
OPTION  ===>

   1  ELEMENT          - Display element/component list information
   2  FOOTPRINT        - Display footprinted members and compressed listings
   3  SITE             - Display site information
   4  STAGE            - Display stage information
   5  SYSTEM           - Display system definitions
   6  SUBSYSTEM        - Display subsystem definitions
   7  TYPE             - Display type definitions
   8  PROCESSOR GROUP  - Display processor group definitions
   9  APPROVER GROUP   - Display approver groups
   A  RELATE GROUP     - Display inventory area/approver group relationships
   E  ENVIRONMENT      - Display information about the current environment
   S  SITE SYMBOLS     - Display site symbols definitions

 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

3. The Footprint Panel opens. Enter 'iprfx.iqual.SMPLPROD.LISTLIB' in the OTHER PARTITIONED DATA SET field.

```
------------------------- ENDEVOR - FOOTPRINT DISPLAY -------------------------
Option ===>

   blank - Member selection list
   I - Display load module CSECTS and ENDEVOR footprints
   L - Display the library member


FROM ISPF LIBRARY:
   PROJECT ===>
   LIBRARY ===>
   TYPE    ===>
   MEMBER  ===>            THRU MEMBER ===>

OTHER PARTITIONED DATA SET:
   DATA SET NAME ===> 'BST.SUPP40.SMPLPROD.LISTLIB'

 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

4. Endevor displays a list of members in that library. Tab down to, or issue a locate command for, member FINARP01. Select the member with an L and press Enter.

```
FOOTPRINT  -------------- LIBRARY SELECTION LIST ---------- Row 10 to 18 of 19
Command ===>                                                  Scroll ===> PAGE

                    Library: BST.SUPP40.SMPLPROD.LISTLIB
   I - Display load module CSECTS and ENDEVOR footprints
   L - Display the library member

 For ENDEVOR Elements:
   B - Browse element        C - Show changes only   S - Show change summary
   H - Show change history   M - Show Master Record

              |--------------------  F O O T P R I N T  --------------------|
    MEMBER     SYSTEM   SUBSYSTEM ELEMENT    TYPE     S  VV.LL  DATE   TIME LD
    FINAPS02   FINANCE  ACCTPAY   FINAPS02   COBOL    2  01.00 19SEP02 12:38
    FINAPS03   FINANCE  ACCTPAY   FINAPS03   COBOL    2  01.00 19SEP02 12:38
    FINAPS04   FINANCE  ACCTPAY   FINAPS04   COBOL    2  01.00 19SEP02 12:38
  L FINARP01   FINANCE  ACCTREC   FINARP01   COBOL    2  01.00 24OCT02 16:02
    FINARP02   FINANCE  ACCTREC   FINARP02   COBOL    2  01.00 15OCT02 14:33
    FINARP03   FINANCE  ACCTREC   FINARP03   COBOL    2  01.00 15OCT02 14:34
    FINARS01   FINANCE  ACCTREC   FINARS01   COBOL    2  01.00 19SEP02 12:40
    FINARS02   FINANCE  ACCTREC   FINARS02   COBOL    2  01.00 19SEP02 12:40
    FINARS03   FINANCE  ACCTREC   FINARS03   COBOL    2  01.00 19SEP02 12:41
 F1=HELP      F2=SPLIT     F3=END       F4=RETURN    F5=RFIND     F6=RCHANGE
 F7=UP        F8=DOWN      F9=SWAP     F10=LEFT     F11=RIGHT    F12=RETRIEVE
```

5. The stored listing is displayed. Review the listing for compile and/or link errors.

```
******************************** Top of Data ********************************
****************************************************************************
****************************************************************************
**
**  FFFFFFFFFFFF   IIIIIIIIII   NN        NN   AAAAAAAAAA   RRRRRRRRRRR   PPPPP
**  FFFFFFFFFFFF   IIIIIIIIII   NNN       NN  AAAAAAAAAAAA  RRRRRRRRRRRR  PPPPP
**  FF                 II       NNNN      NN  AA        AA  RR        RR  PP
**  FF                 II       NN NN     NN  AA        AA  RR        RR  PP
**  FF                 II       NN  NN    NN  AA        AA  RR        RR  PP
**  FFFFFFFF           II       NN   NN   NN  AAAAAAAAAAAA  RRRRRRRRRRRR  PPPPP
**  FFFFFFFF           II       NN    NN  NN  AAAAAAAAAAAA  RRRRRRRRRRR   PPPPP
**  FF                 II       NN     NN NN  AA        AA  RR    RR      PP
**  FF                 II       NN      NNNN  AA        AA  RR     RR     PP
**  FF                 II       NN       NNN  AA        AA  RR      RR    PP
**  FF             IIIIIIIIII   NN        NN  AA        AA  RR       RR   PP
**  FF             IIIIIIIIII   NN         N  AA        AA  RR        RR  PP
**
****************************************************************************
****************************************************************************


****************************************
****************************************
**      ********   GENERATE   ********
**
```

```
**      USER ID......... TURKA02
**      DATE............ 24OCT02 16:02
**      ENDEVOR RC...... 0000
**
**      ENVIRONMENT..... SMPLPROD
**      STAGE........... PROD
**      SYSTEM.......... FINANCE
**      SUBSYSTEM....... ACCTREC
**      ELEMENT......... FINARP01
**      VV.LL........... 01.00
**      TYPE............ COBOL
**      PROC GROUP...... CLENBL
**      PROCESSOR....... GCIINBL
**        STEP1......... RC=0000
**        CONWRITE...... RC=0000
**        CONWRIT2...... RC=0000
**        CONRELE....... RC=0000
**        COMPILE....... RC=0000
**        LINK.......... RC=0004
**
****************************************

PP 5648-A25 IBM COBOL for OS/390 & VM  2.2.1
Invocation parameters:
LIB,NOSEQ,OBJECT,APOST,
Options in effect:
    NOADATA
      ADV
    NOANALYZE
      APOST
      ARITH(COMPAT)
    NOAWO
      BUFSIZE(4096)
    NOCICS
.
.
. PP 5648-A25 IBM COBOL for OS/390 & VM  2.2.1                    FINARP01  Date 10
  LineID  PL SL  ----+-*A-1-B--+----2----+----3----+----4----+----5----+----6---
  000001         000100 IDENTIFICATION DIVISION.
  000002                PROGRAM-ID. FINARP01.
  000003         000300 ENVIRONMENT DIVISION.
  000004                INPUT-OUTPUT SECTION.
  000005         000500 FILE-CONTROL.
  000006                    SELECT REPORT-FILE ASSIGN U-T-SYSOUT.
  000007                    SELECT INPUT-FILE ASSIGN U-T-INPUT.
  000008         000700 DATA DIVISION.
  000009         000800 FILE SECTION.
  000010                FD REPORT-FILE
  000011                    LABEL RECORDS ARE OMITTED
RECORDING MODE IS F

.
.
.
ENTRY POINT AND ALIAS SUMMARY:

NAME:            ENTRY TYPE AMODE C_OFFSET CLASS NAME        STATUS
```

```
FINARP01          MAIN_EP     ANY 00000000 B_TEXT

                       ***** E N D   O F   R E P O R T *****



OS/390 V2 R10 BINDER     16:03:40 THURSDAY OCTOBER 24, 2002
BATCH EMULATOR  JOB(TURKA02 ) STEP(CATSO   ) PGM= IEWL      PROCEDURE(CATSO   )
IEW2008I 0F03 PROCESSING COMPLETED.  RETURN CODE =  4.



---------------------
MESSAGE SUMMARY REPORT
---------------------
SEVERE MESSAGES        (SEVERITY = 12)
NONE

ERROR MESSAGES         (SEVERITY = 08)
NONE

WARNING MESSAGES       (SEVERITY = 04)
2520

INFORMATIONAL MESSAGES (SEVERITY = 00)
2008  2278  2322


**** END OF MESSAGE SUMMARY REPORT ****

****************************** Bottom of Data ******************************
```

Performing Background Verification

Follow the steps below to verify, in batch mode, that the sample environments have been loaded correctly.

1. Select option 3, Batch, from the AllFusion Endevor Primary Options Panel and press Enter.

```
--------------------AllFusion Endevor Primary Options Panel----------------------
Option ===> 3

  0   DEFAULTS      - Specify Endevor ISPF default parameters
  1   DISPLAY       - Perform Display functions
  2   FOREGROUND    - Execute Foreground Actions
  3   BATCH         - Perform Batch Action processing
  4   ENVIRONMENT   - Define or Modify Environment information
  5   PACKAGE       - Perform Foreground Package processing
  6   BATCH PACKAGE - Perform Batch Package SCL Generation
  U   USER MENU     - Display user option menu
  T   TUTORIAL      - Display information about Endevor
  C   CHANGES       - Display summary of changes for this release of Endevor
  X   EXIT          - Exit the Endevor dialog



                        Current environment: SMPLPROD

    (C) 2002 Computer Associates International, Inc.


Use the EXIT option to terminate Endevor
```

2. The Batch Options Menu opens. In the PROJECT, GROUP, and TYPE fields, enter the qualifiers for any 80 byte partitioned dataset. In the Member field, type SCLVER. In the Option field type 1 and press Enter.

```
BATCH ----------------------  Batch Options Menu  ----------------------------
OPTION  ===> 1

   1 BUILD SCL - Build batch SCL actions
   2 EDIT      - Edit request data set
   3 SUBMIT    - Submit job for batch processing
   4 VALIDATE  - Check request data set for syntax errors
   5 BUILD JCL - Enter additional JCL to be included with the job

 REQUEST DATA SET:
    PROJECT  ===> MCCPE01         APPEND      ===> N (Y/N)
    GROUP    ===> PGM             INCLUDE JCL ===> N (Y/N)
    TYPE     ===> SCL
    MEMBER   ===> SCLVER        <<< This field is for the scl only

 OTHER PARTITIONED OR SEQUENTIAL DATA SET:
    DSNAME ===>

 JOB STATEMENT INFORMATION:
    ===> //MCCPE01U JOB (41200000),'ENDEVOR',MSGCLASS=X,
    ===> // NOTIFY=MCCPE01,TIME=1439,
    ===> // CLASS=A
    ===>
```

3. The SCL Generation menu opens. Select option 4, Generate, and press Enter.

```
------------------------------  SCL GENERATION  ------------------------------
OPTION  ===> 4

   1  DISPLAY       - Display an element
   2  ADD/UPDATE    - Add or update an element into stage 1
   3  RETRIEVE      - Retrieve or copy an element
   4  GENERATE      - Execute the Generate Processor for this element
   5  MOVE          - Move an element to the next inventory location
   6  DELETE        - Delete an element
   7  PRINT ELEMENT - Print elements, changes and detail change history
   8  SIGNIN        - Explicitly sign-in an element
   9  TRANSFER      - Transfer elements between two ENDEVOR locations
  10  PRINT MEMBER  - Print a compressed listing or member
  11  LIST ELEMENT  - Create List actions for ENDEVOR elements
  12  LIST MEMBER   - Create List actions for external members
  13  ARCHIVE       - Archive elements


      REQUEST DATA SET: MCCPE01.PGM.SCL(SCLVER)
      APPEND:           N
```

4. The Generate Elements panel opens. Type the highlighted values on the panel and press Enter.

```
 ------------------------------ GENERATE ELEMENTS ------------------------------
OPTION  ===> G
                                ELEMENT DISPLAY OPTIONS:
   blank - Element list           S - Summary   B - Browse    H - History
   G - Generate element           M - Master    C - Changes

FROM ENDEVOR:                        ACTION OPTIONS:
  ENVIRONMENT ===> SMPLPROD             CCID             ===> SAMPLE
  SYSTEM      ===> FINANCE              COPYBACK         ===> N (Y/N)
  SUBSYSTEM   ===> ACCTREC              OVERRIDE SIGNOUT ===> N (Y/N)
  ELEMENT     ===> FINARP01             PROCESSOR GROUP  ===>
  TYPE        ===> COBOL
  STAGE       ===> P        E - EMER         P - PROD

  COMMENT     ===> VERIFY

LIST OPTIONS:
  DISPLAY LIST      ===> Y (Y/N)
  WHERE CCID EQ     ===>
  WHERE PROC GRP EQ ===>
  BUILD USING MAP   ===> Y (Y/N)
```

5. Endevor responds by displaying the message SCL GENERATED in the upper right of the panel.

```
----------------------------- GENERATE ELEMENTS --------------- SCL GENERATED
OPTION  ===>
                                ELEMENT DISPLAY OPTIONS:
   blank - Element list          S - Summary   B - Browse    H - History
   G - Generate element          M - Master    C - Changes

FROM ENDEVOR:                        ACTION OPTIONS:
  ENVIRONMENT ===> SMPLPROD             CCID             ===> SAMPLE
  SYSTEM      ===> FINANCE              COPYBACK         ===> N (Y/N)
  SUBSYSTEM   ===> ACCTREC              OVERRIDE SIGNOUT ===> N (Y/N)
  ELEMENT     ===> FINARP01             PROCESSOR GROUP  ===>
  TYPE        ===> COBOL
  STAGE       ===> P        E - EMER         P - PROD

  COMMENT     ===> VERIFY

LIST OPTIONS:
  DISPLAY LIST      ===> Y (Y/N)
  WHERE CCID EQ     ===>
  WHERE PROC GRP EQ ===>
  BUILD USING MAP   ===> N (Y/N)
```

6. Press PF3 twice.

7. The Batch Options Menu opens. Type your JOBCARD information at the bottom of the panel. Select option 3 and press Enter.

```
BATCH -----------------------  BATCH OPTIONS MENU  ----------------------------
OPTION  ===> 3

   1 BUILD SCL - Build batch SCL actions
   2 EDIT      - Edit request data set
   3 SUBMIT    - Submit job for batch processing
   4 VALIDATE  - Check request data set for syntax errors
   5 BUILD JCL - Enter additional JCL to be included with the job

 REQUEST DATA SET:
    PROJECT  ===> MCCPE01         APPEND      ===> N (Y/N)
    GROUP    ===> PGM             INCLUDE JCL ===> N (Y/N)
    TYPE     ===> SCL
    MEMBER   ===> SCLVER        <<< This field is for the scl only

 OTHER PARTITIONED OR SEQUENTIAL DATA SET:
    DSNAME ===>

 JOB STATEMENT INFORMATION:
    ===> //MCCPE01U JOB (41200000),'ENDEVOR',MSGCLASS=X,
    ===> // NOTIFY=MCCPE01,TIME=1439,
    ===> // CLASS=A
    ===>
```

8. The job is submitted for execution. Endevor displays a message that confirms that the job was submitted.

   ```
   IKJ56250I JOB MCCPE01U(JOB07814) SUBMITTED
   ```

9. Use SDSF, Unicenter® CA-SYSVIEW® Realtime Performance Management, or a similar facility to view the batch job output. Check the last page for a return code of 0000. If the return code is 0000, the job executed successfully. If the return code is not 0000, review the execution report (which is generated automatically) to determine the problem.

Performing ACMQ Verification

If you have not already done so, copy your JOBCARD member to the beginning of SMPLACMX in iprfx.iqual.JCLLIB, then submit the job for execution. SMPLACMX performs a batch ACM query request that will find all programs that use copybook PAGING.

```
//*(JOBCARD)
//*-------------------------------------------------------------------*
//*                                                                   *
//*   (C) 2002 COMPUTER ASSOCIATES INTERNATIONAL, INC. (CA)           *
//*                                                                   *
//* NAME: BC1JACMX                                                    *
//*                                                                   *
//* FUNCTION: THIS JOB PROVIDES AN "EXPLOSION" REPORT OF ALL          *
//*           CHILDREN LEVELS RELATED TO A SELECTED PARENT STARTING   *
//*           POINT IN THE ACM XREF FILES.                            *
//*                                                                   *
//*  1) ADD A VALID JOB CARD AT THE FRONT OF THE JOB STREAM           *
//*  2) CHANGE THE DATASET NAMES AND OTHER VARIABLES                  *
//*     TO APPROPRIATE VALUES FOR YOUR INSTALLATION.                  *
//*-------------------------------------------------------------------*
//*******************************************************************
//**  EXECUTE THE ACM QUERY EXPLOSION REPORT                        **
//*******************************************************************
//STEP1    EXEC PGM=NDVRC1,PARM='BC1PACMQ',REGION=4096K
//STEPLIB  DD  DISP=SHR,DSN=UPRFX.UQUAL.AUTHLIB
//         DD  DISP=SHR,DSN=IPRFX.IQUAL.AUTHLIB
//CONLIB   DD  DISP=SHR,DSN=IPRFX.IQUAL.CONLIB
//ACMOUT   DD  SYSOUT=*,DCB=RECFM=FBA
//ACMSCLIN    DD  *
LIST USING COMPONENTS FOR
   ELEMENT       PAGING
   ENVIRONMENT   *
   SYS           *
   SUB           *
```

```
   TYPE          COPY
   STAGE NUMBER  *

/*
//ACMMSGS1 DD SYSOUT=*
//ACMMSGS2 DD SYSOUT=*
```

The output should look similar to the following.

```
          (C) 2002 Computer Associates International, Inc. (CA)
                            ACMQ INPUT PARAMETERS
REQUESTED BY : USER0001
16:10:09  C1Y0015I  STARTING PARSE OF REQUEST CARDS
 STATEMENT #1
   LIST USING COMPONENTS FOR
   ELEMENT PAGING
   ENVIRONMENT *
   SYS *
   SUB *
   TYPE COPY
   STAGE NUMBER *
   .
 STATEMENT #2
 EOF STATEMENT GENERATED
16:10:09  C1Y0016I  REQUEST CARDS SUCCESSFULLY PARSED
          (C) 2002 Computer Associates International, Inc. (CA)
                            ACMQ QUERY REQUEST
REQUESTED BY : USER0001
16:10:13  C1G0202I  ACTION #1 / STMT #1
16:10:13  ACM0203I     LIST USING COMPONENTS FOR
16:10:13  ACM0204I        ELEMENT: PAGING     ENVIRONMENT : SMPLPROD
16:10:13  ACM0205I        SYSTEM : FINANCE    SUBSYSTEM   : ACCTREC
16:10:13  ACM0206I        TYPE   : COPY       STAGE NUMBER: 1
16:10:13  ACM0210I  STMT #1     COMPLETED RC = 0000 .

REQUEST COMPLETED HIGHEST RC = 0

 (C) 2002 Computer Associates International, Inc. (CA)
                            ACMQ ACTION SUMARY REPORT
REQUESTED BY : USER0001

                                                            # ELM  # ELM
ELEMENT      TYPE       ENVIRON    SYSTEM     SUBSYS    STG  USING   USED   RC
----------   --------   --------   --------   --------  ---  ------ ------  --
PAGING       COPY       SMPLPROD   FINANCE    ACCTREC    1        5      0  00
          (C) 2002 Computer Associates International, Inc. (CA)
                            ACMQ QUERY RESULTS
REQUESTED BY : USER0001

+=====+============+==========+==========+==========+==========+=====+
| LVL | ELEMENT    | TYPE     | ENVIRON  | SYSTEM   | SUBSYS   | STG |
+=====+============+==========+==========+==========+==========+=====+
    1   PAGING       COPY       SMPLPROD   FINANCE    ACCTREC     1
    2   FINAPP01     COBOL      SMPLPROD   FINANCE    ACCTPAY     2
    2   FINAPP02     COBOL      SMPLPROD   FINANCE    ACCTPAY     2
    2   FINARP01     COBOL      SMPLPROD   FINANCE    ACCTREC     2
    2   FINARP02     COBOL      SMPLPROD   FINANCE    ACCTREC     2
    2   FINARP03     COBOL      SMPLPROD   FINANCE    ACCTREC     2
****************************** BOTTOM OF DATA******************************
```

Other Sample Jobs

The Endevor JCL library iprfx.iqual.JCLLIB contains several JCL members that can be used in conjunction with the sample environments. A description of each job is included below.

- **SMPLTRNG**—This job uses the SCL statements in iprfx.iqual.SOURCE (TRNGSCL) to add 20 Cobol programs and copybooks for use in on-site training classes offered by Computer Associates. If desired, you could also add these elements for in-house training.
- **SMPLDEL**—Executes IDCAMS and IEFBR14 to delete all the sample application files from disk.
- **SMPLGENR**—Executes Endevor Batch SCL to generate all elements in the sample application.
- **SMPLLIBR**—Executes Endevor Batch Admin SCL to update type definitions when using CA-Librarian.
- **SMPLPANV**—Executes Endevor Batch Admin SCL to update type definitions when using CA-Panvalet.
- **SMPLJREL**—Executes Endevor Batch Admin SCL to define new type definitions that can be used in conjunction with conrele in a processor to demonstrate the use of ACMQ to query on user-related information.
- **SMPLUNLD**—Executes the Endevor Unload Utility to backup the sample application data.
- **SMPLRELD**—Executes the Endevor Reload Utility to restore the sample application data (using Unload utility data as input).

For detailed information of the Endevor Unload/Reload/Validate utility, refer to the *Utilities Guide*.

In Summary

This chapter provided step-by-step instructions for implementing Endevor's sample application. The tasks included:

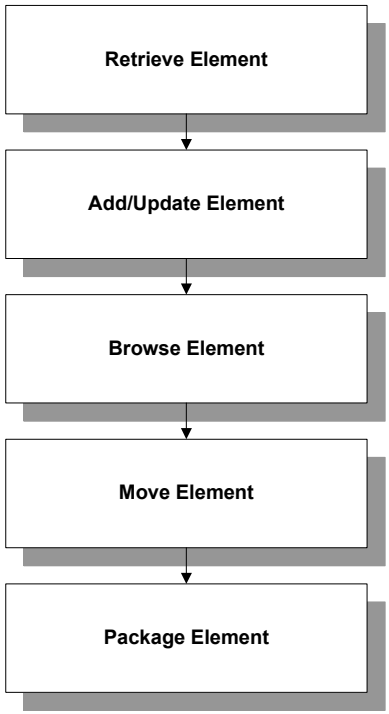
- Editing the Endevor Defaults Table
- Defining and allocating Endevor libraries
- Defining an Endevor inventory structure and populating it with an application
- Verifying the implementation

In the next chapter, we will walk you through a basic development life cycle, retrieving and updating an element, putting it back through the lifecycle and ultimately returning it to production.

Chapter 4

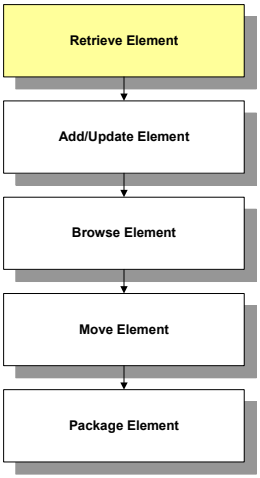
A Day in the Endevor Life Cycle

In the previous chapter you submitted several batch jobs that provided a glimpse into the duties of an Endevor administrator, including such tasks as defining systems, subsystems and types. In this chapter, you will become familiar with the basic concepts of an Endevor development life cycle by performing typical activities – retrieving and updating an element, moving it back through the Endevor life cycle, and eventually returning to the Production environment. The steps are summarized in the following diagram.



Our scenario will provide you with experience that you can apply to your own applications and development life cycle. It uses the elements COBOL program FINARP01 and copybook HEADER1.

Retrieving an Element



The first step in a typical Endevor life cycle is the RETRIEVE action, which copies Endevor elements to a user's library and signs them out to that user ID. You can perform a RETRIEVE in both foreground and batch mode. In our scenario, we will RETRIEVE in foreground mode. For more information on the RETREIVE action, see the *User Guide*.

To RETRIEVE the elements in foreground:

1. Bring up Endevor as you did in the previous chapter.
2. From the Environment Selection Menu select 1, SMPLTEST.
3. From the Primary Options Menu select 2, Foreground.

4. From the Foreground Options Menu select: 3, Retrieve. The Retrieve Elements panel opens. Enter the highlighted information in the appropriate fields and press Enter.

   Note: In the TO ISPF LIBRARY or TO OTHER PARTITIONED OR SEQUENTIAL DATASET fields, enter the name of a library or dataset that you can update.

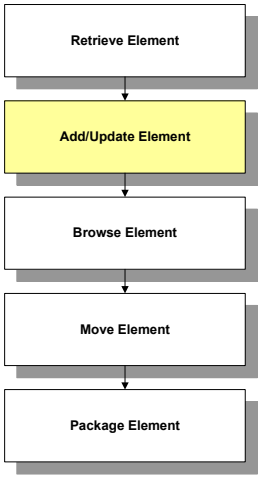
```
------------------------------ RETRIEVE ELEMENTS ------------------------
OPTION  ===>
                                 ELEMENT DISPLAY OPTIONS:
   blank - Element list             S - Summary   B - Browse    H - History
   R - Retrieve element             M - Master    C - Changes

FROM ENDEVOR:                         ACTION OPTIONS:
  ENVIRONMENT ===> SMPLTEST              CCID              ===> LIFECYCLE
  SYSTEM      ===> FINANCE               EXPAND INCLUDES   ===> N (Y/N)
  SUBSYSTEM   ===> ACCTREC               SIGNOUT ELEMENT   ===> Y (Y/N)
  ELEMENT     ===>                       OVERRIDE SIGNOUT ===> N (Y/N)
  TYPE        ===>                       REPLACE MEMBER    ===> Y (Y/N)
  STAGE       ===> T        T - TEST         Q - QA
  COMMENT     ===> SAMPLE LIFE CYCLE

TO ISPF LIBRARY:                      LIST OPTIONS:
   PROJECT ===> DEVEL                    DISPLAY LIST      ===> Y (Y/N)
   LIBRARY ===> TEST                     WHERE CCID EQ     ===>
   TYPE    ===> JCLLIB                   WHERE PROC GRP EQ ===>
   MEMBER  ===>                          BUILD USING MAP   ===> Y (Y/N)
                                         FIRST FOUND       ===> Y (Y/N)
TO OTHER PARTITIONED OR SEQUENTIAL DATA SET:
   DATA SET NAME ===>
```

5. The Element Selection List panel opens, allowing you to choose the elements you want to retrieve. To RETRIEVE an element, enter an R next to the element name. In our scenario, we select elements FINARP01 and HEADER1 and press Enter. The selected elements are marked *RETRIEVED and copied to the library designated on the Retrieve Panel.

```
RETRIEVE ----------------------  ELEMENT SELECTION LIST  --- Row 1 to 25 of 25
 COMMAND ===>                                                 SCROLL ===> PAGE

 TO   Data set:    DEVEL.TEST.JCLLIB

   ELEMENT    NEWNAME    TYPE      ENVIRON  S  SYSTEM    SUBSYSTEM  VV.LL  RC
   FARCOB01              COBOL     SMPLPROD P  FINANCE   ACCTREC    01.00  0000
   FARINCL               COPY     SMPLPROD P  FINANCE   ACCTREC    01.00  0004
   FARSUB02              COBOL     SMPLPROD P  FINANCE   ACCTREC    01.00  0000
   FINARC01              COPY      SMPLPROD P  FINANCE   ACCTREC    01.00  0004
   FINARC02              COPY      SMPLPROD P  FINANCE   ACCTREC    01.00  0004
 R FINARP01   *RETRIEVED COBOL     SMPLPROD P  FINANCE   ACCTREC    01.00  0000
   FINARP02              COBOL     SMPLPROD P  FINANCE   ACCTREC    01.00  0000
   FINARP03              COBOL     SMPLPROD P  FINANCE   ACCTREC    01.00  0000
   FINARS01              COBOL     SMPLPROD P  FINANCE   ACCTREC    01.00  0000
   FINARS02              COBOL     SMPLPROD P  FINANCE   ACCTREC    01.00  0000
   FINARS03              COBOL     SMPLPROD P  FINANCE   ACCTREC    01.00  0000
   FINARS04              COBOL     SMPLPROD P  FINANCE   ACCTREC    01.00  0000
   FOOTER                COPY      SMPLPROD P  FINANCE   ACCTREC    01.00  0004
 R HEADER1    *RETRIEVED COPY      SMPLPROD P  FINANCE   ACCTREC    01.00  0004
   PAGEHDR               COPY      SMPLPROD P  FINANCE   ACCTREC    01.00  0004
   PAGING                COPY      SMPLPROD P  FINANCE   ACCTREC    01.00  0004
   STOPRUN               COPY      SMPLPROD P  FINANCE   ACCTREC    01.00  0004
   TITLE1                COPY      SMPLPROD P  FINANCE   ACCTREC    01.00  0004
   TITLE2                COPY      SMPLPROD P  FINANCE   ACCTREC    01.00  0004
```

6. Using ISPF Edit, edit these members in the dataset you designated in the Retrieve Elements panel. (For this exercise simply add a few comment lines to each member). After editing, ADD the updated members back into Endevor.

Adding an Element



In the next step in our scenario, we will add the HEADER1 and FINARP01 elements into Endevor.

Repeat the steps you performed for the RETRIEVE action; that is, start Endevor, select the SMPLTEST environment and select the Foreground option from the AllFusion Endevor Primary Options Menu. From the Foreground Options Menu, select option 2 ADD/UPDATE to open the ADD/UPDATE Elements panel.

Since the updated COBOL program, FINARP01, uses the updated copybook, HEADER1, you must add the copybook first. Type the highlighted values in the panel and press Enter. Be sure to use the library name that you retrieved the elements to in the FROM ISPF LIBRARY fields.

```
---------------------------  ADD/UPDATE ELEMENTS  ----------------------
OPTION  ===>  A

   blank - Member list     A - Add an element    U - Update an element

TO ENDEVOR:                          ACTION OPTIONS:
   ENVIRONMENT ===> SMPLTEST            CCID                  ===> LIFECYCLE
   SYSTEM      ===> FINANCE             GENERATE ELEMENT      ===> Y (Y/N)
   SUBSYSTEM   ===> ACCTREC             DELETE INPUT SOURCE   ===> N (Y/N)
   ELEMENT     ===> HEADER1            NEW VERSION          ===>
   TYPE        ===> COPY                OVERRIDE SIGNOUT      ===> N (Y/N)
   STAGE:           T                   PROCESSOR GROUP       ===>
                                        UPDATE IF PRESENT     ===> Y (Y/N)
   COMMENT      ===> SAMPLE LIFECYCLE

FROM ISPF LIBRARY:                   LIST OPTIONS:
   PROJECT ===> DEVEL                DISPLAY LIST      ===> Y (Y/N)
   LIBRARY ===> TEST
   TYPE    ===> JCLLIB
   MEMBER  ===>             THRU MEMBER ===>

FROM OTHER PARTITIONED OR SEQUENTIAL DATA SET:
   DATA SET NAME ===>
```

A brief sysout listing is displayed, indicating that the version of HEADER1 you added was compared to the version in SMPLPROD. It identifies how many lines you changed and confirms that the element was successfully added.

```
BROWSE    TURKA02.C1TEMPR1.ECA31.MSGS                Line 00000000 Col 001
 Command ===>                                            Scroll ===>
******************************** Top of Data****************************


09:42:56  C1G0203I     ADD      ELEMENT HEADER1
09:42:56  C1G0205I        FROM DSNAME:  DEVEL.TEST.JCLLIB
09:42:56  C1G0204I        TO   ENVIRONMENT: SMPLTEST SYSTEM: FINANCE SUBSYSTEM: ACCTREC
TYPE: COPY     STAGE ID: T
09:42:56  C1G0232I        OPTIONS:  UPDATE
09:42:56  C1G0232I                  CCID: LIFECYCLE
09:42:56  C1G0232I                  COMMENT: SAMPLE LIFECYCLE
09:43:05  C1G0265I  PROCESSOR GROUP *NOPROC* FOR THIS ELEMENT WAS OBTAINED FROM SECONDARY
ELEMENT RECORD
09:43:06  SMGR139I  ELEMENT FOUND AT LOCATION SMPLPROD/P/FINANCE/ACCTREC/COPY SELECTED
FOR FETCH PROCESSING
09:43:09  SMGR120I  ELEMENT VV.LL 01.00 FETCHED FOR PROCESSING AT STAGE TEST
09:43:11  SMGR136I  ELEMENT VV.LL 01.01 CHANGES INCLUDE 1 LINES INSERTED AND 0 LINES
DELETED
09:43:11  SMGR127I  ELEMENT VV.LL 01.01 WRITTEN TO:
09:43:11  C1G0000I     DATA SET BST.SUPP40.SMPLTEST.COPYLIB
09:43:11  C1G0000I     MEMBER HEADER1
****************************** Bottom of Data *************************
```

Press F3 to exit out of the MSGS dataset and return to the ADD/UPDATE Elements panel. In the upper right corner of the screen the message HEADER1 ADDED is displayed.

With the copybook successfully added, you can add the program that uses it. On the ADD/UPDATE Element panel, change the element name to FINARP01 and change TYPE to COBOL as highlighted below. Leave all other field values unchanged. Press Enter.

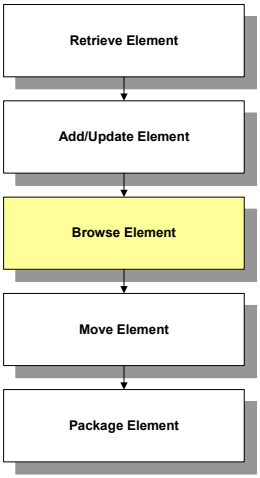
```
---------------------------  ADD/UPDATE ELEMENTS  ----------------------
OPTION  ===>   A

   blank - Member list     A - Add an element    U - Update an element

TO ENDEVOR:                          ACTION OPTIONS:
   ENVIRONMENT ===> SMPLTEST            CCID                 ===> LIFECYCLE
   SYSTEM      ===> FINANCE             GENERATE ELEMENT     ===> Y (Y/N)
   SUBSYSTEM   ===> ACCTREC             DELETE INPUT SOURCE  ===> N (Y/N)
   ELEMENT     ===> FINARP01              NEW VERSION          ===>
   TYPE        ===> COBOL               OVERRIDE SIGNOUT     ===> N (Y/N)
   STAGE:           T                   PROCESSOR GROUP      ===>
                                        UPDATE IF PRESENT    ===> Y (Y/N)
    COMMENT      ===> SAMPLE LIFECYCLE

FROM ISPF LIBRARY:                   LIST OPTIONS:
   PROJECT ===> DEVEL                DISPLAY LIST      ===> Y (Y/N)
   LIBRARY ===> TEST
   TYPE    ===> JCLLIB
   MEMBER  ===>             THRU MEMBER ===>

FROM OTHER PARTITIONED OR SEQUENTIAL DATA SET:
    DATA SET NAME ===>
```

Endevor adds the program member and executes the COBOL generate processor to compile and link the program. When the processor is finished the MSGS dataset is displayed again. Source management messages, similar to those issued for HEADER1, are displayed along with the processor listing and processor execution messages. At the end of the listing, you will see the final return code for your action:

```
C1G0144I  PROCESSOR GCIINBLR EXECUTION COMPLETED, HIGHEST PROCESSOR STEP RC WAS 0004.
```

Press F3 and the message FINARP01 ADDED is displayed in the upper right corner of the ADD/UPDATE Element panel. You have successfully added and generated your copybook and program changes.

Browsing an Element



You can verify the changes that you just made using Endevor Display functions. The Display functions enable you to view detailed information about the element master, element changes, element history, and summary information. For more information the Display functions, see the *User Guide*.

Start by selecting option 1, DISPLAY, from the Foreground Option panel. The Display Elements/Component Lists panel opens. This panel enables you to filter your element selection list based on the criteria specified when you added the element. For example, you could use CCID or processor group as a filter. Because you added both elements using the same CCID (LIFECYCLE), you can use that value to limit the element selection list. To build your element list enter the highlighted information in the appropriate fields and press Enter.

```
------------------  Display Elements/Component Lists  ------------------

OPTION  ===>

   blank - Display selection list      B - Browse element current level
   S - Display summary of levels       C - Display changes current level
   M - Display element master info     H - Display history current level

   Enter SX, BX, CX or HX to display component list information

                                   LIST OPTIONS:
FROM ENDEVOR:                         DISPLAY PROC GRP NAME ===> N  (Y/N)
  ENVIRONMENT ===> SMPLTEST           DISPLAY LIST          ===> Y  (Y/N)
  SYSTEM      ===> FINANCE            WHERE CCID EQ         ===> LIFECYCLE
  SUBSYSTEM   ===> ACCTREC            WHERE PROC GRP EQ     ===>
  ELEMENT     ===>                    DISPLAY SYS/SBS LIST  ===> N  (Y/N)
  TYPE        ===>                    BUILD USING MAP       ===> N  (Y/N)
  STAGE       ===> T          T - TEST        Q - QA
```

The Element Selection List opens.

```
--------------------------  ELEMENT SELECTION LIST  --------- Row 1 to 2 of 2
COMMAND ===>                                                 SCROLL ===> PAGE

                                                             ---- DATES ----
   ELEMENT    TYPE     ENVIRON    S SYSTEM   SUBSYSTEM  VV.LL CURRENT GENERATE
   FINARP01   COBOL    SMPLTEST   T FINANCE  ACCTREC    01.01 07NOV02 07NOV02
   HEADER1    COPY     SMPLTEST   T FINANCE  ACCTREC    01.01 07NOV02 07NOV02
****************************** Bottom of data *********************
```

From the Element Selection List you can choose any element by entering one of the following values next to the element name.

- blank—Display selection list
- B—Browse element current level
- S—Display summary of levels
- C—Display changes current level
- M—Display element master info
- H—Display history current level
- SX, BX, CX or HX—Display Component List information

Note: If you used SCLBASE to define the sample application, or if the element you are displaying does not use a processor, the BX, SX, CX, etc. Component List options will not be available.

Option B opens the Element Browse panel, which displays all statements in the current level of the element, and the level at which each statement was inserted.

```
******************************** Top of Data *************************
**********************************************************************
**
**  ELEMENT BROWSE                                    11NOV02  11:45  *
**
**    ENVIRONMENT:  SMPLTEST    SYSTEM: FINANCE     SUBSYSTEM: ACCTREC       **
      ELEMENT:      FINARP01    TYPE:   COBOL       STAGE ID:  T             **
**    SIGNED OUT TO: TURKA02                                                 **
**********************************************************************
**********************************************************************
------------------------- SOURCE LEVEL INFORMATION ----------------------

 VV.LL SYNC USER     DATE    TIME     STMTS CCID         COMMENT
```

```
 ----- ---- -------- ------- ----- -------- ------------ -----------------
 01.00      PILRO01C 19SEP02 12:34      102 SAMPLE       INSTALL SAMPLE AP
 01.01      TURKA02  07NOV02 13:29      105 LIFECYCLE    SAMPLE LIFECYCLE
GENERATED   TURKA02  07NOV02 13:29          LIFECYCLE    SAMPLE LIFECYCLE

 +00    000100 IDENTIFICATION DIVISION.
 +00            PROGRAM-ID. FINARP01.
 +00    000300 ENVIRONMENT DIVISION.
 +00           INPUT-OUTPUT SECTION.
 +00    000500 FILE-CONTROL.
 +00               SELECT REPORT-FILE ASSIGN U-T-SYSOUT.
 +00               SELECT INPUT-FILE ASSIGN U-T-INPUT.
 +00               02 INPUT-FIELD6           PIC X(6).
 +00               02 FILLER                 PIC X(24).
  .
  .
  .
 +00           ****************************************************
%+01           *  ADDED 3 COMMNET LINES FOR SAMPLE LIFE CYCLE DOC
%+01           *
%+01           ****************************************************
 +00            WORKING-STORAGE SECTION.
 +00            COPY HEADER1.
```

Displaying the Element Master

The Display functions enable you to view element master information. If you specify M in the Element Selection List, the Element Master panel opens. This panel displays all the information Endevor stores about your element, including the date and time it was added to Endevor, the date it was last generated, the user ID associated with the last action, and the signout user ID, if applicable. The Element Master panel below shows information for the element FINARP01.

```
------------------------------ ELEMENT MASTER -----------------------------
COMMAND ===>                                                (PANEL 1 OF 2)

ELEMENT:  FINARP01   ENV: SMPLTEST SYS: FINANCE  SUB: ACCTREC  TYPE: COBOL
PROC GRP: CLENBLR    STG: T      VV.LL: 01.01          LAST ACTION: ADD
DESCRIPTION: INSTALL SAMPLE APPLICATION ELEMENTS         SIGNOUT ID: TURKA02
PKG ID (SOURCE):                     PKG ID (OUTPUT):
LOCKED FOR PKG:
--------------------------  LAST ELEMENT ACTION  --------------------------
USERID:  TURKA02    DATE/TIME: 07NOV02 13:29          CCID: LIFECYCLE
COMMENT: SAMPLE LIFECYCLE                            ACTION: ADD
NDVR RC: 0000       PROCESSOR: GCIINBLR  (GEN)      PROC RC: 0004

-----------------------------  CURRENT SOURCE  ----------------------------
USERID:  TURKA02    DATE/TIME: 07NOV02 13:29          CCID: LIFECYCLE
COMMENT: SAMPLE LIFECYCLE                         DELTA FMT: R
ADD/UPDATE FROM DSN: TURKA02.TEST.JCLLIB(FINARP01)

--------------------------------  GENERATE  -------------------------------
USERID:  TURKA02    DATE/TIME: 07NOV02 13:29          CCID: LIFECYCLE
COMMENT: SAMPLE LIFECYCLE
COMPONENT LIST VV.LL: 01.00                       DELTA FMT: R
                                             (Press ENTER for next panel)
```

Press Enter to display the second Element Master panel.

```
------------------------------  ELEMENT MASTER  ------------------------------
COMMAND ===>                                                   (PANEL 2 OF 2)

ELEMENT:  FINARP01   ENV: SMPLTEST SYS: FINANCE  SUB: ACCTREC  TYPE: COBOL
PROC GRP: CLENBLR    STG: T      VV.LL: 01.01          LAST ACTION: ADD
                                                        SIGNOUT ID: TURKA02
--------------------------------  RETRIEVE  ---------------------------------
USERID:             DATE/TIME:                    CCID:
COMMENT:
RETRIEVE TO DSN:
----------------------------------  BASE  ----------------------------------
USERID:  PILRO01C   DATE/TIME: 19SEP02 12:34
COMMENT: INSTALL SAMPLE APPLICATION ELEMENTS
--------------------------  FROM ENDEVOR LOCATION  --------------------------
USERID:             DATE/TIME:                  ACTION:
ELEMENT:            ENV:          SYS:          SUB:          TYPE:
    STG:          VV.LL:
```

Displaying Element Changes

If you specify C in the Element Selection List, the Element Changes panel opens. This panel shows all insertions and deletions made to the element at its current level.

```
                                                (Press ENTER for previous panel)
BROWSE    TURKA02.FINARP01.S1.ECA31.COBOL            Line 00000000 Col 001 080
 Command ===>                                                Scroll ===> CSR
******************************** Top of Data ********************************
****************************************************************************
****************************************************************************
**                                                                        **
** ELEMENT CHANGES                                          08NOV02  17:56 **
**                                                                        **
**    ENVIRONMENT:  SMPLTEST    SYSTEM: FINANCE     SUBSYSTEM: ACCTREC      **
**    ELEMENT:      FINARP01    TYPE:   COBOL       STAGE ID:  T            **
**                                                                        **
**    SIGNED OUT TO: TURKA02                                              **
**                                                                        **
****************************************************************************
****************************************************************************

------------------------- SOURCE LEVEL INFORMATION --------------------------

 VV.LL SYNC USER     DATE    TIME     STMTS CCID         COMMENT
 ----- ---- -------- ------- ----- -------- ------------ ----------------------
 01.00      PILRO01C 19SEP02 12:34      102 SAMPLE       INSTALL SAMPLE APPLICA
 01.01      TURKA02  07NOV02 13:29      105 LIFECYCLE    SAMPLE LIFECYCLE
GENERATED   TURKA02  07NOV02 13:29          LIFECYCLE    SAMPLE LIFECYCLE

 +01          *  ADDED 3 COMMNET LINES FOR SAMPLE LIFE CYCLE DOC
 +01          *
```

```
 +01          ****************************************************************
****************************** Bottom of Data ******************************
```

Displaying Element History

If you specify H in the Element Selection List, the Element History panel opens. This panel shows all statements in all levels of the element, and the level at which each insertion and deletion occurred.

```
BROWSE    TURKA02.FINARP01.S1.ECA31.COBOL            Line 00000000 Col 001 080
 Command ===>                                                Scroll ===> CSR
******************************** Top of Data ********************************
******************************************************************************
******************************************************************************
**                                                                          **
** ELEMENT HISTORY                                          08NOV02  17:57  **
**                                                                          **
**    ENVIRONMENT:  SMPLTEST    SYSTEM: FINANCE     SUBSYSTEM: ACCTREC       **
**    ELEMENT:      FINARP01    TYPE:   COBOL       STAGE ID:  T             **
**                                                                          **
**    SIGNED OUT TO: TURKA02                                                **
**                                                                          **
******************************************************************************
******************************************************************************

------------------------- SOURCE LEVEL INFORMATION ---------------------------

 VV.LL SYNC USER     DATE    TIME     STMTS CCID         COMMENT
 ----- ---- -------- ------- ----- -------- ------------ ----------------------
 01.00      PILRO01C 19SEP02 12:34      102 SAMPLE       INSTALL SAMPLE APPLICA
 01.01      TURKA02  07NOV02 13:29      105 LIFECYCLE    SAMPLE LIFECYCLE
GENERATED   TURKA02  07NOV02 13:29          LIFECYCLE    SAMPLE LIFECYCLE

 +00    000100 IDENTIFICATION DIVISION.
 +00           PROGRAM-ID. FINARP01.
 +00    000300 ENVIRONMENT DIVISION.
 +00           INPUT-OUTPUT SECTION.
 +00               02 INPUT-FIELD6           PIC X(6).
 +00               02 FILLER                 PIC X(24).
  .
  .
  .
 +00          ************************************************
%+01          *  ADDED 3 COMMNET LINES FOR SAMPLE LIFE CYCLE DOC
%+01          *
%+01          ************************************************
 +00           WORKING-STORAGE SECTION.
 +00           COPY HEADER1.
.
.
```

Displaying the Summary of Levels

If you specify S in the Element Selection List, the Summary of Levels panel opens. This panel shows a summary of the change history for an element. From this panel you can display a specific level of the element using option B, C, or H.

```
----------------------------  SUMMARY OF LEVELS  ------------ Row 1 to 2 of 2
COMMAND ===>                                                 SCROLL ===> PAGE

        Environment: SMPLTEST     System: FINANCE       Subsystem: ACCTREC
        Element:     FINARP01     Type:   COBOL         Stage:     T

------------------------  SOURCE LEVEL INFORMATION  -------------------------
     VV.LL    USER     DATE     TIME      STMTS    INSERTS   DELETES    SYNC
     01.00    PILRO01C 19SEP02  12:34          102      0         0
     01.01    TURKA02  07NOV02  13:29          105      3         0
****************************** Bottom of data ******************************
```

Displaying Component Information

When an element uses a generate processor that uses the Automated Configuration Manager interface, Endevor builds a Component List, which is a compilation of all the input and output members associated with the generated element, and the libraries they came from. For detailed information on Component Lists, see the *Automated Configuration Option Guide*.

If you specify BX in the Element Selection List, the Component List is displayed. Note there is not a one-to-one relationship between the element's vv.ll and its Component List's vv.ll. Each time an element is generated at its current inventory location, a new level of its Component List is created, even though the element's source may not have changed. Using FINARP01 as an example, its current source level is 01.01, but its Component List is 01.00, because this is the first Component List created for this element at SMPLTEST stage T.

The Component List provides five types of information, as described below:

- The **element** information describes the program being generated.

- The **processor** information describes the Endevor processor used to generate or move the element.
- The **symbol** information identifies the user-defined symbolics used in the Endevor processor, and their values.
- The **input** components identify (by step, DDname, dsname, and volume) the component items referenced and any footprints that existed in a monitored data set.
- The **output** components identify (by step, DDname, dsname, and volume) the members that were created during processor execution.

```
BROWSE    TURKA02.FINARP01.S1.ECA31.COBOL            Line 00000000 Col 001 080
Command ===>                                                  Scroll ===> CSR
****************************** Top of Data ********************************
*******************************************************************************
*******************************************************************************
*                                                                             **
* COMPONENT BROWSE                                          08NOV02  17:58    **
*                                                                             **
*    ENVIRONMENT:  SMPLTEST    SYSTEM: FINANCE     SUBSYSTEM: ACCTREC          **
*    ELEMENT:      FINARP01    TYPE:   COBOL       STAGE ID:  T                **
*                                                                             **
*    SIGNED OUT TO: TURKA02                                                   **
*                                                                             **
*******************************************************************************
*******************************************************************************

------------------------- SOURCE LEVEL INFORMATION ---------------------------

VV.LL SYNC USER     DATE    TIME     STMTS CCID         COMMENT
----- ---- -------- ------- ----- -------- ------------ ----------------------
01.00      TURKA02  07NOV02 13:29       48 LIFECYCLE    SAMPLE LIFECYCLE

-------------------------  ELEMENT INFORMATION  ------------------------------

       VV.LL  DATE   TIME  SYSTEM   SUBSYS   ELEMENT    TYPE     GROUP    STG
+00    01.01 07NOV02 13:29 FINANCE  ACCTREC  FINARP01   COBOL    CLENBLR   1

-----------------------  PROCESSOR INFORMATION  ------------------------------

       VV.LL  DATE   TIME  SYSTEM   SUBSYS   ELEMENT    TYPE     GROUP    STG
+00    01.00 15OCT02 14:30 ADMIN    PROCESS  GCIINBLR   PROCESS            2

-------------------------  SYMBOL INFORMATION  ------------------------------

       DEFINED      SYMBOL        VALUE
+00    PROCESSOR    CIICOMP       IBMPROD.V1R4M0.COB2COMP
+00    PROCESSOR    CIILIB        IBMPROD.V1R4MO.COB2LIB
+00    PROCESSOR    CLECOMP       IGY.SIGYCOMP
+00    PROCESSOR    CLELKED       CEE.SCEELKED
+00    PROCESSOR    CLERUN        CEE.SCEERUN
+00    PROCESSOR    CSYSLIB1      &PROJECT..&GROUP.&C1ST..COPYLIB
```

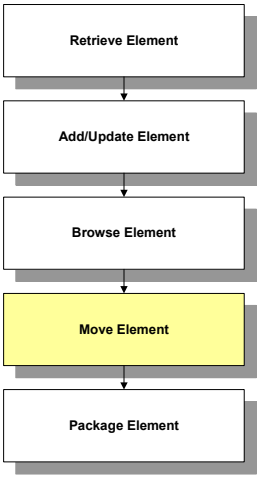
```
+00    PROCESSOR    CSYSLIB2      &PROJECT..&GROUP.&C1ST2..COPYLIB
+00    PROCESSOR    CSYSLIB3      &PROJECT..&GROUP.&STG3..COPYLIB
+00    PROCESSOR    CSYSLIB4      &PROJECT..&GROUP.&STG4..COPYLIB
+00    PROCESSOR    EXPINC        N
+00    PROCESSOR    GROUP         SMPL
+00    PROCESSOR    LISTLIB       &PROJECT..&GROUP.&C1ST..LISTLIB
+00    PROCESSOR    LOADLIB       &PROJECT..&GROUP.&C1ST..LOADLIB
+00    PROCESSOR    LSYSLIB1      &PROJECT..&GROUP.&C1ST..LOADLIB
+00    PROCESSOR    LSYSLIB2      &PROJECT..&GROUP.&C1ST2..LOADLIB
+00    PROCESSOR    LSYSLIB3      &PROJECT..&GROUP.&STG3..LOADLIB
+00    PROCESSOR    LSYSLIB4      &PROJECT..&GROUP.&STG4..LOADLIB
+00    PROCESSOR    MEMBER        &C1ELEMENT
+00    PROCESSOR    MONITOR       COMPONENTS
+00    PROCESSOR    PARMCOB       LIB,NOSEQ,OBJECT,APOST,
+00    PROCESSOR    PARMLNK       LIST,MAP,XREF
+00    PROCESSOR    PROJECT       BST.SUPP40
+00    PROCESSOR    STG3          EMER
+00    PROCESSOR    STG4          PROD
+00    PROCESSOR    SYSOUT        A
+00    PROCESSOR    WRKUNIT       SYSDA

--------------------------   INPUT COMPONENTS   ------------------------------

STEP: COMPILE  DD=SYSLIB   VOL=NDVS03 DSN=BST.SUPP40.SMPLPROD.COPYLIB

       MEMBER     VV.LL  DATE   TIME  SYSTEM   SUBSYS   ELEMENT    TYPE     STG
+00    PAGING     01.00 19SEP02 12:01 FINANCE  ACCTREC  PAGING     COPY      2

TEP: COMPILE  DD=SYSLIB   VOL=NDVS05 DSN=BST.SUPP40.SMPLTEST.COPYLIB

       MEMBER     VV.LL  DATE   TIME  SYSTEM   SUBSYS   ELEMENT    TYPE     STG
+00    HEADER1    01.01 07NOV02 13:28 FINANCE  ACCTREC  HEADER1    COPY      1

STEP: LINK     DD=SYSLIB   VOL=NDVS06 DSN=BST.SUPP40.SMPLPROD.LOADLIB

        MEMBER     VV.LL  DATE   TIME  SYSTEM   SUBSYS   ELEMENT    TYPE     STG
 +00    FINARS01   01.00 19SEP02 12:32 FINANCE  ACCTREC  FINARS01   COBOL     1
 +00    FINARS02   01.00 19SEP02 12:32 FINANCE  ACCTREC  FINARS02   COBOL     1

--------------------------  OUTPUT COMPONENTS  ------------------------------

STEP: LINK     DD=SYSLMOD  VOL=NDVS04 DSN=BST.SUPP40.SMPLTEST.LOADLIB

        MEMBER     VV.LL  DATE   TIME  SYSTEM   SUBSYS   ELEMENT    TYPE     STG
 +00    FINARP01   01.01 07NOV02 13:29 FINANCE  ACCTREC  FINARP01   COBOL     1

STEP: STORLIST DD=C1LLIBO  VOL=NDVS02 DSN=BST.SUPP40.SMPLTEST.LISTLIB

        MEMBER     VV.LL  DATE   TIME  SYSTEM   SUBSYS   ELEMENT    TYPE     STG
 +00    FINARP01   01.01 07NOV02 13:29 FINANCE  ACCTREC  FINARP01   COBOL     1
****************************** Bottom of Data ******************************
```

Displaying Summary of Levels for Element Components

You can display changes, history and a summary of levels for an element's Component List as well as the element source. If you specify SX in the Element Selection, the Summary of Levels panel opens. In this case, the Summary of Levels for the Element Components reveals FINARP01 has only a base level (01.00). As a result, there are no changes (CX) to display, and a display of the Component List history (HX) would be identical to a browse (BX).

```
----------------------------  SUMMARY OF LEVELS  ------------ Row 1 to 1 of 1
COMMAND ===>                                                  SCROLL ===> PAGE

        Environment: SMPLTEST     System: FINANCE       Subsystem: ACCTREC
        Element:     FINARP01     Type  : COBOL         Stage:     T

----------------------  COMPONENT LEVEL INFORMATION  -------------------------
     VV.LL    USER     DATE     TIME      STMTS    INSERTS   DELETES
     01.00    TURKA02  07NOV02  13:29          48      0         0
************************* Bottom of data ****************************
```

Moving Elements



Now that you have confirmed and tested your changes, you are ready for the next stage of the life cycle, the MOVE to QA. The MOVE action can be performed in foreground or in batch. In our scenario, we will use the foreground MOVE action.

To perform the MOVE in foreground, follow the steps below:

1. From the Foreground option, select option 5, Move. The Move Elements panel is displayed. As with the Display panel, you can filter your selection list on this panel. In order to build your element list based on the CCID LIFECYCLE, enter the highlighted information in the appropriate fields and press Enter.

```
------------------------------- MOVE ELEMENTS --------------------------
OPTION  ===>
                              ELEMENT DISPLAY OPTIONS:
   blank - Element list          S - Summary   B - Browse   H - History
   O - Move element              M - Master    C - Changes

FROM ENDEVOR:                            ACTION OPTIONS:
  ENVIRONMENT ===> SMPLTEST            CCID                  ===> LIFECYCLE
  SYSTEM      ===> FINANCE             SYNC                  ===> N (Y/N)
  SUBSYSTEM   ===> ACCTREC             WITH HISTORY          ===> N (Y/N)
  ELEMENT     ===>                     RETAIN SIGNOUT        ===> Y (Y/N)
  TYPE        ===>                     SIGNOUT TO            ===>
  STAGE       ===> T                   ACKNOWLEDGE ELM JUMP  ===> N (Y/N)
          T - TEST     Q - QA          DELETE 'FROM' ELEMENT ===> Y (Y/N)

  COMMENT     ===> SAMPLE LIFECYCLE

                                    LIST OPTIONS:
                                      DISPLAY LIST         ===> Y (Y/N)
                                      WHERE CCID EQ        ===> LIFECYCLE
                                      WHERE PROC GRP EQ    ===>
                                      BUILD USING MAP      ===> N (Y/N)
```

2. The Element Selection List is displayed. Next, verify that the elements were moved in the correct order. To do so, tab down to HEADER1, select it with O and press Enter.

```
MOVE ------------------------  ELEMENT SELECTION LIST  ------- Row 1 to 3 of 3
COMMAND ===>                                                 SCROLL ===> PAGE

  ELEMENT              TYPE      ENVIRON  S  SYSTEM    SUBSYSTEM  VV.LL  RC
  FINARP01             COBOL     SMPLTEST T  FINANCE   ACCTREC    01.01  0000
O HEADER1              COPY      SMPLTEST T  FINANCE   ACCTREC    01.01  0000
****************************** Bottom of data ******************************
```

3. The messages dataset is displayed, indicating HEADER1 was successfully written to SMPLTEST/Q/FINANCE/ACCTREC/COPY/HEADER1 and deleted from SMPLTEST/T/FINANCE/ACCTREC/COPY/HEADER1.

```
BROWSE    TURKA02.C1TEMPR1.ECA31.MSGS                Line 00000000 Col 011 090
 Command ===>                                                  Scroll ===> CSR
******************************** Top of Data ***********************************


C1G0203I     MOVE     ELEMENT HEADER1
C1G0204I        FROM ENVIRONMENT: SMPLTEST SYSTEM: FINANCE  SUBSYSTEM: ACCTREC
C1G0232I        OPTIONS:  RETAIN SIGNOUT
C1G0232I                  CCID: LIFECYCLE
C1G0232I                  COMMENT: sample lifecycle
C1G0265I  PROCESSOR GROUP *NOPROC* FOR THIS ELEMENT WAS OBTAINED FROM ENVIRONMEN
SMGR139I  ELEMENT FOUND AT LOCATION SMPLPROD/P/FINANCE/ACCTREC/COPY SELECTED FOR
SMGR120I  ELEMENT VV.LL 01.00 FETCHED FOR PROCESSING AT STAGE QA
SMGR136I  ELEMENT VV.LL 01.01 CHANGES INCLUDE 1 LINES INSERTED AND 0 LINES DELET
SMGR130I  ELEMENT VV.LL 01.01 CREATED AT LOCATION SMPLTEST/Q/FINANCE/ACCTREC/COP
SMGR127I  ELEMENT VV.LL 01.01 WRITTEN TO:
C1G0000I     DATA SET BST.SUPP40.SMPLQA.COPYLIB
C1G0000I     MEMBER HEADER1
C1G0265I  PROCESSOR GROUP *NOPROC* FOR THIS ELEMENT WAS OBTAINED FROM PRIMARY EL
SMGR112I  ELEMENT VV.LL 01.01 DELETED FROM LOCATION SMPLTEST/T/FINANCE/ACCTREC/C
******************************* Bottom of Data *********************************
```

4. Press F3 to return to the MOVE Element Selection List. HEADER1 will be marked *MOVED. Tab next to FINARP01, select it with O and press Enter. The messages dataset will be displayed again, along with the processor listing. Note that the processor listing is different from the one that was displayed with the ADD action. An ADD/UPDATE action executes a generate processor. A MOVE action can execute either a move processor or generate processor. This choice is made when the processor group is defined. In our scenario, type COBOL's processor group executes a move processor, designed to copy the outputs created by the generate processor from one stage's libraries to another. At this stage of the life cycle, the outputs are copied from the SMPLTEST libraries to the SMPLQA libraries. You will also see the delete processor listed and a message indicating that the element was successfully deleted from stage T.

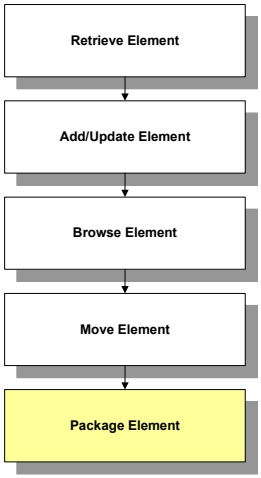
```
C1G0203I     MOVE     ELEMENT FINARP01
C1G0204I        FROM ENVIRONMENT: SMPLTEST SYSTEM: FINANCE  SUBSYSTEM: ACCTREC
C1G0232I        OPTIONS:  RETAIN SIGNOUT
C1G0232I                  CCID: LIFECYCLE
C1G0232I                  COMMENT: sample lifecycle
C1G0265I  PROCESSOR GROUP CLENBLR FOR THIS ELEMENT WAS OBTAINED FROM ENVIRONMENT
SMGR139I  ELEMENT FOUND AT LOCATION SMPLPROD/P/FINANCE/ACCTREC/COBOL SELECTED FO
SMGR120I  ELEMENT VV.LL 01.00 FETCHED FOR PROCESSING AT STAGE QA
SMGR136I  ELEMENT VV.LL 01.01 CHANGES INCLUDE 3 LINES INSERTED AND 0 LINES DELET
SMGR130I  ELEMENT VV.LL 01.01 CREATED AT LOCATION SMPLTEST/Q/FINANCE/ACCTREC/COB
SMGR127I  ELEMENT VV.LL 01.01 WRITTEN TO:
C1G0000I    DATA SET BST.SUPP40.SMPLQA.SRCLIB
C1G0000I    MEMBER FINARP01
C1G0143I  BEGINNING EXECUTION OF MOVE PROCESSOR MLODNNL OF GROUP CLENBLR.

C1G0246I  PROCESSOR MLODNNL LOADED FROM BST.SUPP40.SMPLPROD.PRCSLOAD
C1G0247I    FOOTPRINT: ENV: SMPLPROD SYS: ADMIN SBS: PROCESS ELM: MLODNNL TYPE:
C1G0248I               VER: 01.00 DATE: 19SEP02 11:59
C1G0249I  //*****************************************************************
C1G0249I  //*
C1G0249I  //* COPY LOAD MODULES FROM STAGE 1 TO STAGE 2 AND THEIR ASSOCIATED
C1G0249I  //* COMPONENT LIST AND LISTINGS.
C1G0249I  //*
C1G0249I  //*****************************************************************
C1G0249I  //*
C1G0249I  //MLODNNL  PROC  LISTLIB='YES',
C1G0249I  //               LISTLIB1='&PROJECT..&GROUP.&STG1..LISTLIB',
C1G0249I  //               LISTLIB2='&PROJECT..&GROUP.&STG2..LISTLIB',
C1G0249I  //               LOADLIB1='&PROJECT..&GROUP.&STG1..LOADLIB',
C1G0249I  //               LOADLIB2='&PROJECT..&GROUP.&STG2..LOADLIB',
C1G0249I  //               PROJECT='BST.SUPP40',
C1G0249I  //               GROUP='SMPL',
C1G0249I  //               STG1='&C1SSTAGE.',     CURRENT STAGE
```

```
C1G0249I  //               STG2='&C1STAGE.',      TO STAGE
C1G0249I  //               SYSOUT=*,
C1G0249I  //               WRKUNIT=SYSDA
C1G0249I  //*****************************************************************
C1G0249I  //*   ALLOCATE TEMPORARY LISTING DATASETS
C1G0249I  //*****************************************************************
C1G0249I  //INIT      EXEC  PGM=BC1PDSIN
C1G0249I  //C1INIT01 DD DSN=&&COPYLIST,DISP=(,PASS,DELETE),
C1G0249I  //            UNIT=&WRKUNIT,SPACE=(CYL,(1,2),RLSE),
C1G0249I  //            DCB=(RECFM=V,LRECL=121,BLKSIZE=125,DSORG=PS)
C1G0249I  //*****************************************************************
C1G0249I  //* COPY THE LOAD MODULE
C1G0249I  //*****************************************************************
C1G0249I  //BSTCOPY  EXEC PGM=BSTCOPY,MAXRC=04
C1G0249I  //SYSPRINT  DD  DSN=&&COPYLIST,DISP=(OLD,PASS)
C1G0249I  //SYSUT3    DD  UNIT=&WRKUNIT,SPACE=(TRK,(1,1))
C1G0249I  //SYSUT4    DD  UNIT=&WRKUNIT,SPACE=(TRK,(1,1))
C1G0249I  //INDD      DD  DSN=&LOADLIB1,DISP=SHR
C1G0249I  //OUTDD     DD  DSN=&LOADLIB2,DISP=SHR
C1G0249I  //SYSIN     DD  *
C1G0249I    COPY O=OUTDD,I=INDD
C1G0249I    SELECT MEMBER=((&C1ELEMENT,,R))
C1G0249I  //*****************************************************************
C1G0249I  //* MOVE THE COMPONENT LIST
C1G0249I  //*************************************************************
C1G0249I  //MOVECL   EXEC PGM=BC1PMVCL,COND=(0,NE)
C1G0249I  //*************************************************************
C1G0249I  //*     COPY & STORE THE LISTINGS IF:  &LISTING=LISTING LIBRARY
C1G0249I  //*************************************************************
C1G0249I  //COPYLIST EXEC PGM=CONLIST,MAXRC=0,PARM=COPY,COND=EVEN,
C1G0249I  //         EXECIF=(&LISTLIB,EQ,YES)
C1G0249I  //C1LLIBI  DD DSN=&LISTLIB1,DISP=SHR
C1G0249I  //C1LLIBO  DD DSN=&LISTLIB2,DISP=SHR
C1G0249I  //C1BANNER DD DSN=&&BANNER,DISP=(,PASS,DELETE),
C1G0249I  //            UNIT=&WRKUNIT,SPACE=(TRK,(1,1)),
C1G0249I  //            DCB=(RECFM=FBA,LRECL=121,BLKSIZE=6171,DSORG=PS)
C1G0249I  //LIST01   DD DSN=&&COPYLIST,DISP=(OLD,DELETE)
C1G0249I  //*************************************************************
C1G0249I  //*     PRINT THE LISTINGS IF:  &LISTING=NO
C1G0249I  //*************************************************************
C1G0249I  //PRNTLIST EXEC PGM=CONLIST,MAXRC=0,PARM=PRINT,COND=EVEN,
C1G0249I  //         EXECIF=(&LISTLIB,EQ,NO)
C1G0249I  //C1BANNER DD UNIT=&WRKUNIT,SPACE=(TRK,(1,1)),
C1G0249I  //            DCB=(RECFM=FBA,LRECL=121,BLKSIZE=6171,DSORG=PS)
C1G0249I  //C1PRINT  DD SYSOUT=&SYSOUT,
C1G0249I  //            DCB=(RECFM=FBA,LRECL=133,BLKSIZE=1330,DSORG=PS)
C1G0249I  //LIST01   DD DSN=&&COPYLIST,DISP=(OLD,DELETE)
C1G0249I  //*
C1G0006I  SYMBOLIC SUBSTITUTION:
C1G0007I    SYMBOLIC GROUP DEFINED BY PROCESSOR: SMPL
C1G0007I    SYMBOLIC LISTLIB DEFINED BY PROCESSOR: YES
C1G0007I    SYMBOLIC LISTLIB1 DEFINED BY PROCESSOR: &PROJECT..&GROUP.&STG
C1G0007I    SYMBOLIC LISTLIB2 DEFINED BY PROCESSOR: &PROJECT..&GROUP.&STG
C1G0007I    SYMBOLIC LOADLIB1 DEFINED BY PROCESSOR: &PROJECT..&GROUP.&STG
.
.
.
```

```
C1G0009I      ORIGINAL    : &LOADLIB1
C1G0009I      SUBSTITUTED : BST.SUPP40.SMPLTEST.LOADLIB
C1G0009I      ORIGINAL    : &LOADLIB2
C1G0009I      SUBSTITUTED : BST.SUPP40.SMPLQA.LOADLIB
C1G0009I      ORIGINAL    : &&C1VIOUNIT
C1G0009I      SUBSTITUTED : SYSDA
C1G0009I      ORIGINAL    :   SELECT MEMBER=((&C1ELEMENT,,R))
C1G0009I      SUBSTITUTED :   SELECT MEMBER=((FINARP01,,R))
C1G0009I      ORIGINAL    : &LISTLIB
C1G0009I      SUBSTITUTED : YES
C1G0009I      ORIGINAL    : &LISTLIB1
C1G0009I      SUBSTITUTED : BST.SUPP40.SMPLTEST.LISTLIB
C1G0009I      ORIGINAL    : &LISTLIB2
C1G0009I      SUBSTITUTED : BST.SUPP40.SMPLQA.LISTLIB
.
.
.
C1G0144I  PROCESSOR DLODNNL EXECUTION COMPLETED, HIGHEST PROCESSOR STEP RC WAS 0
SMGR112I  COMPONENT VV.LL 01.00 DELETED FROM LOCATION SMPLTEST/T/FINANCE/ACCTREC
SMGR112I  ELEMENT VV.LL 01.01 DELETED FROM LOCATION SMPLTEST/T/FINANCE/ACCTREC/C
****************************** Bottom of Data ******************************
```

Press F3 to return to the Element Selection List. Note that FINARP01 is marked as *MOVED.

Packaging the Element



At most sites, the final move into production requires packaging. A package is a set of Endevor actions that may require approval before being executed. Packages offer the following benefits:

- **Element Locking**—An optional feature that allows you to lock the elements in a package, thereby preventing modification of the element at the source and/or target locations of the package action.
- **Component Validation**—For ACM users, component validation verifies all the input components associated with an element's outputs are in the same state as when the output component was created. For example, if HEADER1 was changed a second time and FINARP01 was not re-generated, then component validation could fail the CAST of the package moving FINARP01 to the next stage in the life cycle
- **Restart**—A package can be restarted if it fails during execution. The package is "checkpointed" and, when re-executed, begins at the first action that failed and re-executes the failed actions.
- **Backout**—Package outputs can be backed out (and subsequently backed in) after a package has been executed.
- **Approval Processing**—Endevor provides the ability to define approver groups and assign them to specific inventory areas. This relationship forces the use of packages and provides an additional layer of security by only allowing members of these approver groups to execute the package once its been cast.
- **Inspect**—Checks approval, each element for security, signout, synchronization conflicts and source changes that might affect the successful execution of the package.

In packaging an element, the following task are performed:

- A package is created to house the elements.
- The elements are moved to the package.
- The package is CAST to prepare it for execution.
- The package is executed.

In our scenario, we will perform each of these tasks.

For detailed information on Packages, see the *Packages Guide*.

Creating a Package

In the following steps, we begin by creating a package.

1. To access the Package menu select option 5, Package, from the Primary Options Menu.
2. The Package Foreground Options Menu opens. The first step in building a package is to create it. However, before you can create it, you have to name it. In our scenario, we'll name our package SMPLPKG. As highlighted below, specify 2, CREATE/MODIFY, on the Option line, type SMPLPKG in the Package ID field, and press Enter.

```
---------------------- Package Foreground Options Menu ------
 Option  ===> 2

   1  DISPLAY        - Display Package Information
   2  CREATE/MODIFY  - Create or Modify Package
   3  CAST           - Prepare Package for Review
   4  REVIEW         - Approve or Deny Package
   5  EXECUTE        - Submit or Execute Package
   6  SHIP           - Ship Packages
   7  BACKOUT        - Perform Backout or Backin Processing
   8  COMMIT         - Clear Backout Information
   9  UTILITIES      - Reset, Delete, or Export Package

        Package ID ===>  SMPLPKG

 Limit selection list options. These options are used by the
 DISPLAY and UTILITIES functions:

       In-Edit......... Y            In-Execution.... Y
       In-Approval..... Y            Executed........ Y
       Denied.......... Y            Committed....... Y
       Approved........ Y            Enterprise Pkg.. A
```

3. The CREATE/MODIFY Package panel opens. Create a package by building SCL that specifies the actions to be performed against the elements. To Build SCL, choose option B, Build, specify the other highlighted information, and press Enter.

```
MODIFY -------------------- CREATE/MODIFY PACKAGE ----------------------
 OPTION  ===> B

    B - Build Package Actions             I  - Import SCL
    E - Edit Package                      C  - Copy Package
    N - Add Notes to Package

 PACKAGE ID: SMPLPKG                       STATUS: IN-EDIT
 DESCRIPTION  ===> SAMPLE PACKAGE
 PACKAGE TYPE ===> STANDARD
 SHARABLE PACKAGE  ===> Y (Y/N)            APPEND TO PACKAGE ===> N (Y/N)
 ENABLE BACKOUT    ===> Y (Y/N)
 EXECUTION WINDOW FROM ===> 11NOV02 00:00  TO ===> 31DEC79 00:00

 INPUT PACKAGE ID ===>

 FROM ISPF LIBRARY:
    PROJECT ===> DEVEL
    GROUP   ===> TEST
    TYPE    ===> JCLLIB
    MEMBER  ===>

 OTHER PARTITIONED OR SEQUENTIAL DATA SET:
    DATA SET NAME ===>
```

Moving Elements to the Package

1. After completing the Create/Modify Package panel and pressing Enter, the SCL Generation panel opens. From this panel, you can build as many actions as you want into the Package. For our scenario, we will only be using MOVE. Choose option 5, MOVE, and press Enter.

```
------------------------------  SCL GENERATION  -----------------------
OPTION  ===>     5

   1  DISPLAY       - Display an element
   2  ADD/UPDATE    - Add or update an element into stage 1
   3  RETRIEVE      - Retrieve or copy an element
   4  GENERATE      - Execute the Generate Processor for this element
   5  MOVE          - Move an element to the next inventory location
   6  DELETE        - Delete an element
   7  PRINT ELEMENT - Print elements, changes and detail change history
   8  SIGNIN        - Explicitly sign-in an element
   9  TRANSFER      - Transfer elements between two ENDEVOR locations
  10  PRINT MEMBER  - Print a compressed listing or member
  11  LIST ELEMENT  - Create List actions for ENDEVOR elements
  12  LIST MEMBER   - Create List actions for external members
  13  ARCHIVE       - Archive elements


      REQUEST DATA SET: PACKAGE - SMPLPKG
      APPEND:           N
```

2. The MOVE ELEMENTS panel opens. At this step, you are building SCL to perform the MOVE at a later time. Filter the selection list by using the WHERE CCID EQ LIFECYCLE field, modify the other highlighted fields, and press Enter. Note that RETAIN SIGNOUT is set to N. This setting will remove your ownership from the element when it gets written at the target location (PROD).

```
------------------------------- MOVE ELEMENTS ----------------------------
OPTION  ===>
                                ELEMENT DISPLAY OPTIONS:
   blank - Element list            S - Summary   B - Browse   H - History
   O - Move element                M - Master    C - Changes

FROM ENDEVOR:                         ACTION OPTIONS:
  ENVIRONMENT ===> SMPLTEST             CCID                  ===> LIFECYCLE
  SYSTEM      ===> FINANCE              SYNC                  ===> N (Y/N)
  SUBSYSTEM   ===> ACCTREC              WITH HISTORY          ===> N (Y/N)
  ELEMENT     ===>                      RETAIN SIGNOUT        ===> N (Y/N)
  TYPE        ===>                      SIGNOUT TO            ===>
  STAGE       ===> Q                    ACKNOWLEDGE ELM JUMP  ===> N (Y/N)
          T - TEST     Q - QA           DELETE 'FROM' ELEMENT ===> Y (Y/N)

  COMMENT     ===> SAMPLE LIFECYCLE

                                      LIST OPTIONS:
                                        DISPLAY LIST         ===> Y (Y/N)
                                        WHERE CCID EQ        ===> LIFECYCLE
                                        WHERE PROC GRP EQ    ===>
                                        BUILD USING MAP      ===> N (Y/N)
```

3. The same element selection list is displayed. Because package actions are executed in type sequence order, we can select the elements in any order, as long as they reside in the same environment, system and stage. Tab down next to FINARP01 and select with O. Select HEADER1 in the same way. Press Enter. This time they are marked *WRITTEN not *MOVED, indicating that the SCL to MOVE the elements has been written to a temporary dataset which will be written to the package.

```
MOVE ------------------------  ELEMENT SELECTION LIST  ------- Row 1 to 2
COMMAND ===>                                                 SCROLL ===>

  ELEMENT              TYPE      ENVIRON  S  SYSTEM    SUBSYSTEM  VV.LL
  FINARP01   *WRITTEN   COBOL     SMPLTEST Q  FINANCE   ACCTREC    01.01
  HEADER1    *WRITTEN   COPY      SMPLTEST Q  FINANCE   ACCTREC    01.01
****************************** Bottom of data **************************
```

4. Press F3 to return to the Move Elements panel. Since there are no other elements to MOVE, press F3 again to return to SCL Generation panel. The message 2 REQUEST(S) BUILT is displayed in upper right corner.

```
------------------------------  SCL GENERATION  ---------- 2 REQUEST(S) BUILT
OPTION  ===>

   1  DISPLAY       - Display an element
   2  ADD/UPDATE    - Add or update an element into stage 1
   3  RETRIEVE      - Retrieve or copy an element
   4  GENERATE      - Execute the Generate Processor for this element
   5  MOVE          - Move an element to the next inventory location
   6  DELETE        - Delete an element
   7  PRINT ELEMENT - Print elements, changes and detail change history
   8  SIGNIN        - Explicitly sign-in an element
   9  TRANSFER      - Transfer elements between two ENDEVOR locations
  10  PRINT MEMBER  - Print a compressed listing or member
  11  LIST ELEMENT  - Create List actions for ENDEVOR elements
  12  LIST MEMBER   - Create List actions for external members
  13  ARCHIVE       - Archive elements


      REQUEST DATA SET: PACKAGE - SMPLPKG
      APPEND:           N
```

5. At this point, you could select additional options to build more actions for the package, if needed. Because no additional actions are needed for our scenario, press F3 to return to the CREATE/MODIFY PACKAGE panel. Note the PACKAGE CONSTRUCTED message in the upper right corner.

```
MODIFY -------------------- CREATE/MODIFY PACKAGE -------- PACKAGE CONSTRUCTED
OPTION  ===>

   B - Build Package Actions             I  - Import SCL
   E - Edit Package                      C  - Copy Package
   N - Add Notes to Package

PACKAGE ID: SMPLPKG                       STATUS: IN-EDIT
DESCRIPTION  ===> sample lifecycle pkg
PACKAGE TYPE ===> STANDARD
SHARABLE PACKAGE  ===> N (Y/N)            APPEND TO PACKAGE ===> N (Y/N)
ENABLE BACKOUT    ===> Y (Y/N)
EXECUTION WINDOW FROM ===> 11NOV02 00:00  TO ===> 31DEC79 00:00

INPUT PACKAGE ID ===>

FROM ISPF LIBRARY:
   PROJECT ===> DEVEL
   GROUP   ===> TEST
   TYPE    ===> JCLLIB
   MEMBER  ===>


OTHER PARTITIONED OR SEQUENTIAL DATA SET:
   DATA SET NAME ===>
```

6. At this point if you decide you want to add more actions to the package you need to change the APPEND TO PACKAGE flag to Y; otherwise the MOVE SCL you just created will be overwritten by any new actions you build. Press F3 to return to the Package Foreground Options Menu.

Casting the Package

Once you are satisfied with the contents of your package, it needs to be CAST. Casting a package prepares it for review and subsequent execution. When you CAST a package, Endevor does the following:

- Determines whether approvers have been assigned to the inventory area(s) included in the package.
- Checks the integrity of the package components.

- Makes sure that the package contains the most recent versions of all components.
- Validates lock status, then locks the elements in a package to prevent their modification or inclusion in another package. A package can no longer be edited once it is cast.

To perform the CAST, follow the steps below.

1. Press F3 at the Create/Modify Package panel to return to the Package Foreground Options Menu. Select 3, Cast, and press Enter. The Cast Package panel is displayed. On this panel you have the choice of casting the package, displaying the SCL contained in the package or adding notes or comments to the package. To display the SCL enter S on the option line and press enter:

```
CAST -------------------------- CAST PACKAGE ----------------------
 OPTION  ===>  C

    C - Cast Package                      S  - Display SCL
    N - Add Notes to Package

 PACKAGE ID: SMPLPKG                       STATUS:  IN-EDIT
 DESCRIPTION:  sample lifecycle pkg
 PACKAGE TYPE: STANDARD
 SHARABLE PACKAGE:   Y
 VALIDATE COMPONENTS   ===> Y (Y/N/W)
 ENABLE BACKOUT        ===> Y (Y/N)

 EXECUTION WINDOW FROM ===> 11NOV02 00:00  TO ===> 31DEC79 00:00

               USER ID  DATE    TIME
 CREATED:      TURKA02  11NOV02 16:56
 LAST UPDATED:
```

2. Endevor displays the SCL.

```
CAST - PACKAGE ID: SMPLPKG ------------------------- LINE 00000000 CO
 COMMAND ===>                                                  SCROLL
 DESCRIPTION: sample lifecycle pkg
********************************* Top of Data ************************
MOVE ELEMENT 'FINARP01'
  FROM ENVIRONMENT 'SMPLTEST' SYSTEM 'FINANCE' SUBSYSTEM 'ACCTREC'
    TYPE 'COBOL' STAGE Q
  OPTIONS CCID 'LIFECYCLE' COMMENTS "sample lifecycle" RETAIN SIGNOUT
 .
MOVE ELEMENT 'HEADER1'
  FROM ENVIRONMENT 'SMPLTEST' SYSTEM 'FINANCE' SUBSYSTEM 'ACCTREC'
    TYPE 'COPY' STAGE Q
  OPTIONS CCID 'LIFECYCLE' COMMENTS "sample lifecycle" RETAIN SIGNOUT
 .
******************************** Bottom of Data **********************
```

3. To add notes, enter N at the option line. Endevor displays a freeform text panel:

```
---------------------------- Package Note Text --------------------
 Command ==>

        .........1.........2.........3.........4.........5.........6
    1.  ____________________________________________________________
    2.  ____________________________________________________________
    3.  ____________________________________________________________
    4.  ____________________________________________________________
    5.  ____________________________________________________________
    6.  ____________________________________________________________
    7.  ____________________________________________________________
    8.  ____________________________________________________________


    Press ENTER to process the Package Notes.  When all the note
    text has been entered, press ENTER and then enter the END
    command.

    To cancel the Package Notes, enter the END command.
```

4. To Cast the package type C on the Option line and press Enter. Once the cast process is complete, the Cast Report is displayed.

```
Computer Associates International, Inc.
                          E N D E V O R   P A C K A G E   C A S T   R E P O R T
                           S C L   S T A T E M E N T   S Y N T A X   P A R S E


15:06:08  C1Y0015I  STARTING PARSE OF REQUEST CARDS

 STATEMENT #1
 MOVE ELEMENT 'FINARP01'
   FROM ENVIRONMENT 'SMPLTEST' SYSTEM 'FINANCE' SUBSYSTEM 'ACCTREC'
     TYPE 'COBOL' STAGE Q
   OPTIONS CCID 'LIFECYCLE' COMMENTS "SAMPLE LIFECYCLE" RETAIN SIGNOUT
  .

 STATEMENT #2
 MOVE ELEMENT 'HEADER1'
   FROM ENVIRONMENT 'SMPLTEST' SYSTEM 'FINANCE' SUBSYSTEM 'ACCTREC'
     TYPE 'COPY' STAGE Q
   OPTIONS CCID 'LIFECYCLE' COMMENTS "SAMPLE LIFECYCLE" RETAIN SIGNOUT
  .

 STATEMENT #3
 EOF STATEMENT GENERATED
15:06:08  C1Y0016I  REQUEST CARDS SUCCESSFULLY PARSED

15:06:08  PKMR400I  BEGINNING ACTION VALIDATION AND SEARCH FOR APPLICABLE APPROVER
15:06:08  PKMR401I  ACTION VALIDATION COMPLETED WITHOUT ERRORS
15:06:08  PKMR402I  NO APPROVER GROUP(S) FOUND APPLICABLE FOR PACKAGE
Computer Associates International, Inc.
                          E N D E V O R   P A C K A G E   C A S T   R E P O R T



15:06:08  PKMR791I  COMPONENT VALIDATION STARTED

15:06:08  PKMR799I  COMPONENT VALIDATION COMPLETED : HIGHEST RC = 00

15:06:08  ENMP302I  PACKAGE INSPECT:  ELEMENT AND MEMBER VALIDATION STARTED
15:06:08  ENMP303I                    ELEMENT AND MEMBER VALIDATION SUCCESSFULLY
15:06:08  ENMP305I  PACKAGE INSPECT:  ELEMENT MOVE/TRANSFER SYNC CHECK STARTED
15:06:08  ENMP306I                    ELEMENT MOVE/TRANSFER SYNC CHECK SUCCESSFUL
15:06:08  ENMP308I  PACKAGE INSPECT:  ELEMENT CCID/COMMENTS CHECK STARTED
15:06:08  ENMP309I                    CCID/COMMENTS CHECK SUCCESSFULLY COMPLETED
****************************** Bottom of Data ******************************
```

5. Press F3 to return to the CAST panel where the message CAST SUCCESSFUL is displayed. If approver groups were found, the package status would be IN-APPROVAL, and the next step in the life cycle would be REVIEW. As this package has no approvers associated with it, the status has been updated to APPROVED. Press F3 to return to the Package Options Menu.

Executing a Package

Once a package is in approved status, the next step is the execution.

1. From the Package Foreground Options Menu select option 5, Execute. The Execute/Submit Package Panelopens. Here, you can choose to execute the package in Foreground (option E), or to submit it for batch execution (option S). To make the most efficient use of time and resources, we choose to Submit the package to batch. If there be a problem with the job execution, held output will be readily available and we can forwarded it to Technical Support for review, if needed. Select option S and press Enter.

```
EXECUTE ----------------  EXECUTE/SUBMIT PACKAGE  -------------
OPTION  ===>  S

   E - Execute Package                  S  - Submit Package


PACKAGE ID: SMPLPKG                      STATUS: APPROVED
DESCRIPTION:  sample lifecycle pkg
PACKAGE TYPE: STANDARD
SHARABLE PACKAGE: Y
EXECUTION WINDOW FROM ===> 11NOV02 00:00 TO ===> 31DEC79 00:00

              USER ID  DATE    TIME
CREATED:      TURKA02  11NOV02 16:56
LAST UPDATED: TURKA02  12NOV02 15:02
CAST:         TURKA02  12NOV02 15:06
APPROVED:              12NOV02 15:06
EXECUTED:                              ENDEVOR RC:
```

2. To submit the package, verify you have a valid job card, select option S for submit, and press Enter. The package is submitted for execution.

```
SUBMIT ---------------------  SUBMIT PACKAGE  -----------------
OPTION  ===>    S

   S - Submit job for Batch Processing
   E - Enter additional JCL to be included with the job

PACKAGE ID:   SMPLPKG                    STATUS:  APPROVED
DESCRIPTION:  sample lifecycle pkg
PACKAGE TYPE: STANDARD

INCLUDE JCL   ===> N (Y/N)

JOB STATEMENT INFORMATION:
  ===> //TURKA02G JOB (41300000),SHIP,MSGCLASS=X,NOTIFY=TURKA02
  ===>
  ===>
  ===>
```

3. Using SDSF, Unicenter CA-SYSVIEW Realtime Performance Management or other Output Management product, you can view the package execution report in the held output queue. The execution report will look similar to the messages displayed during the foreground move; that is, HEADER1 followed by FINARP01 and its move and delete processors. Performing the action/package in batch offers the added advantage of the Endevor Action Summary Report. This report provides a one line summary for each action in the batch job/package, its processor and Endevor return codes, and its source location information. This report appears at the end of the C1MSGS sysout, or if C1MSGS2 was coded, it is written there.

```
                        E N D E V O R   A C T I O N   S U M M A R Y   R E P O R T
REQUESTED BY: TURKA02G


                     PROC  NDVR   +--------------  FROM  INFORMATION  ---------
ACTION    ELEMENT     RC    RC    ENVIRONMENT   SYSTEM    SUBSYSTEM   TYPE
MOVE      HEADER1          0000   SMPLTEST      FINANCE   ACCTREC     COPY
MOVE      FINARP01    0000 0000   SMPLTEST      FINANCE   ACCTREC     COBOL
****************************** BOTTOM OF DATA ******************************
```

Once executed, a package can be Backed Out, Committed, Archived, Reset or Deleted. For detailed information on the impact of these package actions, see the *Packages Guide*.

Now that your changes have returned to PROD, your lifecycle is complete and the new versions of FINARP01 and HEADER1 are “signed in” and available for the next developer who needs to modify them.

Congratulations, you have successfully completed a day in the Endevor life cycle! Through our Endevor Sample Application scenario, you have retrieved elements, updated them, moved them through the lifecycle, and returned them to the Production environment. Now you are ready to apply what you have learned to your own site.

Appendix

A

Testing New Endevor Release Upgrades With Your Data

As an alternative to using the Endevor Sample application, you can use your existing production data for testing upgrades to new releases of Endevor. To do so, you must perform several post-installation steps to create a copy of the data on your test system. We recommend copying your largest and most active systems. If your site has many environments, you do not need to establish a test version of them all; however, you should use at least two mapped environments.

Using the steps outlined below, you can convert copies of your production MCFs and Package File, and make copies of all the base, delta, and output libraries associated with the systems you have chosen. If you are changing the names of these files on the test system, or if you need to establish a test bed on your production system, you must update Endevor with the new names.

The conversion steps, summarized below, are described in detail later in this Appendix.

- Create the Element Catalog
- Convert the MCF
- Convert the Package File
- Create C1DEFLTS Table
- Load the Element Catalog
- Allocate ACMQ files
- Synchronize the Element Catalog with MCFs

- Allocate and copy empty data files for base, delta, source/processor output libraries, processor loadlibs
- Copy Production data
- Update Endevor dataset names
- Test

Creating Your Element Catalog

Endevor incorporates an Element Catalog file to support long element names and to boost performance by reducing the volume of I/O operations. The Element Catalog file is required and only one is allowed per site. The Element Catalog is identified to Endevor via the ELMCATL field in the TYPE=MAIN section of the C1DEFLTS Table. To create your Element Catalog, run job BC1JJB07. This member can be found in the *iprfx.iqual.JCLLIB* that is delivered with the product.

Converting the Master Control Files (MCFs)

The MCF data set's maximum record length has changed. As a result, you need to redefine your MCFs. Use member BC1JXMCF located in *iprfx.iqual.JCLLIB*. This job will REPRO your pre-4.0 MCFs into sequential datasets, create your new 4.0 MCFs, and then REPRO the old data from the sequential files into your 4.0 files. For more information on this job see the *Release Summary*.

Converting the Package File

The Package data set's maximum record length has changed. If you want to test using a copy of your existing package file, use member BC1JXPCF from your Endevor JCL library. This job backs up your existing data set to a sequential file, deletes and redefines the package file, then populates the records back into the newly-defined VSAM package file. If you want to test with a new package file, create a new PDS member using only Step 3A and submit it for execution.

Updating Your C1DEFLTS Table

Before you can use any of your newly converted files, you must first identify them to Endevor via the C1DEFLTS table. The BC1JDEFT member, found in the *iprfx.iqual*.JCLLIB will assemble and link your Defaults Table. Make sure that you have updated the table with the new Element Catalog, MCF file names, Package Master File, and ACMQ files, if applicable. The updated table must reside in the authorized library established during the install process. Verify this information with the person who installed the product at your site. For detailed information on the C1DEFLTS Table, please refer to the *Installation Guide*.

Loading the Element Catalog

To load the element catalog with the existing data in your MCFs, run job BC1JXCNV found in the Endevor JCL library, '*iprfx.iqual*.JCLLIB'. This job constructs element catalog data and populates this information in the catalog and cross-reference files. Read carefully all the notes and comments in each step of the JCL before running this job.

Allocating ACMQ Files

If your site is licensed for Automated Configuration Manager (ACM), and you specify ASCM=Y in your C1DEFLTS Table, the ACM query files, ACMROOT and ACMXREF must be allocated. Use job BC1JACMD in the Endevor *iprfx.iqual*.JCLLIB to allocate them. If you want to make copes of your existing ACMQ files, you can use IDCAMS to REPRO them to a sequential file (attributes FB 4096) and then REPRO them back into your test files.

Submit Job BC1JACMO in the *iprfx.iqual*.JCLLIB to convert the ACMQ data to the new release format.

Synchronizing the Element Catalog with Your MCFs

To ensure the Element Catalog that you just loaded is properly referenced by the MCFs defined in your C1DEFLTS Table, run job BC1JXCNM. Make sure that you run it in UPDATE mode. In this mode, the program examines all the MCFs defined in the C1DEFLTS table and stamps the MCFs with the Element Catalog data set name. Use member BC1JXCNM in '*irpfx.iqual*.JCLLIB'.

Note: Be sure that you change the execution parameter to UPDATE before submitting the job.

Allocating Base, Delta, Source/Processor Output Libraries

After processing the MCFs, Package, ACMROOT, XREF and Element Catalog files, you need to allocate your data files. Using IEFBR14 or IDCAMS, allocate empty PDS, PDS/e or ELIBS, based on your site's implementation. Copy any base, deltas, source output libraries, processor output and load libraries associated with the system(s) and types you plan to use in your testing. Using IDCAMS, IEBCOPY, or BC1PNCPY, copy your production files over to the new empty test files. See the Utilities Guide for more information.

Identifying Test Libraries to Endevor

After you have copied all your files, you need to change the dataset names from their production name values to their new test name values. Update the dataset names in each stage for each system imported for testing under each Environment. Use option 8, DATA SET, on the ENVIRONMENT OPTIONS menu to rename your production file names to your test file names. Select option 4 (Environment) from the Primary Options Menu, then 8 for Dataset. The Type Data Set Request Panel is displayed.

```
-------------------------  TYPE DATA SET REQUEST  ----------------
OPTION  ===> U

   blank - Display type data sets
   U - Update type data sets

ENVIRONMENT ===> SMPLTEST

SYSTEM      ===> FINANCE

STAGE       ===> T       T - TEST      Q - QA
```

Specify U, on the Option line along with the name of the system and stage identifier you want to update. Press Enter.

```
UPDATE  ----------------------  TYPE DATA SETS  -------------- Row 1 to 9 of 9
COMMAND ===>                                                   SCROLL ===> PAGE

CURRENT ENV:  SMPLTEST   STAGE ID:  T   SYSTEM:  FINANCE
NEXT    ENV:  SMPLTEST   STAGE ID:  Q   SYSTEM:  FINANCE

   DATA SET NAME                              --- LAST MODIFIED ---
                                                USER    DATE   TIME  TYPE   MSG
BST.SUPP40.SMPL&C1ST..COPYLIB                 PILRO01C 19SEP02 11:23  ??
BST.SUPP40.SMPL&C1ST..DELTA                   PILRO01C 19SEP02 11:23  ??
BST.SUPP40.SMPL&C1ST..MACLIB                  PILRO01C 19SEP02 11:23  ??
BST.SUPP40.SMPL&C1ST..PARMLIB                 PILRO01C 19SEP02 11:23  ??
BST.SUPP40.SMPL&C1ST..BASE                    PILRO01C 19SEP02 11:23  ??
BST.SUPP40.SMPL&C1ST..SRCLIB                  PILRO01C 19SEP02 11:23  ??
BST.PAN40.BASE                                ONETH01  02OCT02 14:50  PV
BST.PAN40.DELTA                               ONETH01  02OCT02 14:50  PV
BST.PAN40.TEST                                ONETH01  10OCT02 11:28  PV
****************************** Bottom of data ******************************
```

The Type Data Sets Panel opens. To rename a data set, simply type over the existing name in the DATA SET NAME column. For more information, see the *Administration Guide*. Any error messages are displayed under the MSG column. Press F1 to get more information or refer to *Error Codes and Messages*. Repeat these steps for both stages of each Environment, for each system you copied.

Note: If you decide not to use your production data files for testing, you have the option of starting from scratch. To do this you need to create all new files, and run Batch Admin (ENBE1000) to build your new environment. See the *SCL Reference* for more information on the Batch Admin Utility.

Test

After you have updated all your systems and stages, you are ready to test! Be certain to test your Exits, making sure that you reassemble and re-link them first.

Appendix

B

Frequently Asked Questions

Questions and Answers

Question: How do I contact AllFusion Endevor Change Manager Technical Support?

Answer: Support is available 24 hours a day, seven days a week. Contact us at esupport.ca.com or call 508-628-8971.

Question: What do the LMP Key Codes stand for?

Answer:

- A8 – AllFusion Endevor Change Manager Parallel Development Option
- A9 – AllFusion Endevor Change Manager Extended Processors Option
- AK – AllFusion Endevor Change Manager
- AY – AllFusion Endevor Change Manager - Automated Configuration Option
- B4 – AllFusion Endevor Change Manager - Interface for CA-Roscoe
- BM – AllFusion Endevor Change Manager - Interface for External Security
- BN – AllFusion Endevor Change Manager - Interface for Tivoli Information Management for z/OS

- BO – AllFusion Endevor Change Manager - Interface for CA-Librarian
- BU – AllFusion Endevor Change Manager - Interface for CA-Panvalet
- BX – AllFusion Endevor Change Manager Quick Edit Option
- CB – AllFusion Endevor Footprint Synchronization Option
- CU – AllFusion Endevor Change Manager - Interface for CA-Netman
- EN – AllFusion Endevor Change Manager Option for DB2

Question: Is there a way that I can open an issue with Endevor Support without calling in on the hotline?

Answer: Go to ca.com and click on Support. This opens a menu of support options. From the list of support options, choose Technical Support. Click on the StarTCC link, then follow the on-screen instructions. Not only can you open an issue with Support from this panel, but you can also download PTFs and documentation, and place orders online.

Question: Where can I learn about the new features in Endevor 4.0?

Answer: The new features are documented in the *Release Summary*. You can also display a summary of the changes from the New Features panel.

Question: Can my ACMROOT, ACMXREF and ELEMENT Catalog be controlled by LSERV?

Answer: It is recommended that the Element Catalog be controlled by LSERV. ACMROOT and ACMXREF cannot be controlled by LSERV.

Question: In 3.9 and earlier releases, there was a PTF O000315 that would bypass approval/package processing. Does this still exist in 4.0?

Answer: Yes. This optional PTF is now part of the ENCOPTBL (Optional PTF Table). To find this PTF in the table, on the command line type f '315/' and press Enter. There are two parts to this PTF. Read them carefully. After you have updated the table, you need to reassemble and re-link, and, if applicable, refresh the linklist.

```
* O000315/A
* THE FOLLOWING OPTIONAL PTF WILL ALLOW TO RUN SPECIFIC ACTIONS
* WITHOUT USING A PACKAGE EVEN IF THE INVENTORY AREA WOULD NORMALLY
* REQUIRE PACKAGE USE
*
USAGE NOTES :
* REPLACE THE (ON,00) WITH THE CORRESPONDING VALUE(S) OF THE ACTION(S)
* YOU WISH TO ALLOW WITHOUT PACKAGES. IF YOU WISH TO ALLOW MORE THAN
* ONE ACTION, ADD THEIR CORRESPONDING VALUES AND REPLACE (ON,00) WITH
* THE TOTAL
*       ARCHIVE   -  01                    RESTORE   -  08
*       DELETE    -  02                    SIGNIN    -  16
*       MOVE      -  04                    TRANSFER  -  32
* EXAMPLES :
* TO ALLOW ARCHIVE ONLY        CODE (ON,01)
* TO ALLOW ARCHIVE AND SIGNIN, CODE (ON,17)
*-----------------------------------------------------------------------*
*        ENHOPT ALLOW_NON_PKG_ACTIONS=(ON,00)
************************************************************************
* O000315/B
* THE FOLLOWING OPTIONAL PTF WILL ALLOW TO RUN SPECIFIC ACTIONS IN
* A PACKAGE WITHOUT REQUIRING APPROVAL.
*
* USAGE NOTES :
* REPLACE THE (ON,00) WITH THE CORRESPONDING VALUE(S) OF THE ACTION(S)
* YOU WISH TO ALLOW WITHOUT PACKAGES. IF YOU WISH TO ALLOW MORE THAN
* ONE ACTION, ADD THEIR CORRESPONDING VALUES AND REPLACE (ON,00) WITH
* THE TOTAL
*       ARCHIVE   -  01                    SIGNIN    -  04
*       DELETE    -  02                    TRANSFER  -  08
* EXAMPLES :
* TO ALLOW ARCHIVE ONLY,       CODE (ON,01)
* TO ALLOW ARCHIVE AND SIGNIN, CODE (ON,05)
*-----------------------------------------------------------------------*
*        ENHOPT PKG_ACTIONS_NO_APPRVR=(ON,00)
```

Question: If I do a DISPLAY element, with the following options specified, CCID=, LIST=Y, and the SYSTEM and/or SUBSYSTEM is left blank or wildcarded (*), I am prompted to choose a SYSTEM and/or SUBSYSTEM. I want the list to include all SYSTEM and SUBSYSTEMS. How can I accomplish this?

Answer: By default, Endevor asks you to choose a specific SYSTEM/SUBSYSTEM combination prior to building an element selection list. To bypass this requirement, activate an option in the ENCOPTBL (Endevor Optional Features Table) for optional PTF O002553. The ENCOPTBL can be found in the *iprfx.iqual*.SOURCE library that is delivered with the product. To enable this option, you need to uncomment the line ENHOPT NO_SYS_SBS_SELECTION_LIST=ON, reassemble and re-link the ENCOPTBL. To find this optional PTF in the table on the command line type F 'O002553' and press Enter.

```
*********************************************************************************
* O002553
* THIS OPTIONAL PTF WILL SUPPRESS THE SYSTEM/SUBSYSTEM PROMPTS ON THE
* BATCH/FOREGROUND AND PKG ACTION PANELS, EVEN WHEN 'DISPLAY LIST' IS
* SET TO Y.
* THIS ALLOWS ELEMENTS SELECTION BY CCID/PROCESSOR GROUP ACROSS MORE
* THAN ONE SYSTEM/SUBSYSTEM.
*
*-------------------------------------------------------------------------------*
     ENHOPT NO_SYS_SBS_SELECTION_LIST=ON
*
```

Question: In earlier release of Endevor, I was able to change the severity of messages generated by Endevor with Optional PTFS; for example, IMGR009E, SMGR116E and C1G0507E. Where can I find those PTFS in 4.0?

Answer: The PTFS for these messages have been incorporated into the Optional PTF Table (ENCOPTBL). This member can be found in *iprfx.iqual*.SOURCE library delivered with the product. The following MSG SEVERITY TABLE has been added to the 4.0 ENCOPTBL.

```
*********************************************************************************
* MSGSEVERITY TABLE                                          *
*********************************************************************************
* THE FOLLOWING SECTION OF THE OPTIONS TABLE ALLOWS TO CHANGE THE
* MESSAGE SEVERITY FOR A NUMBER OF SPECIFIED MESSAGES.
* YOU CAN LEAVE THE OPTION UNSPECIFIED (COMMENTED OUT) WHICH MEANS THAT
* THE DEFAULT MESSAGE SEVERITY (AS SHOWN) WILL BE ISSUED.
*
* YOU CAN ALSO SELECT TO ISSUE THE OTHER ALLOWABLE MESSAGE SEVERITIES
* (AS SHOWN) ON THE SAMPLE STATEMENT.
* IF YOU SELECT A MESSAGE SEVERTY THAT IS NOT ALLOWED THE ASSEMBLY OF
* THE OPTIONS TABLE WILL FAIL.
*-------------------------------------------------------------------------------
*               MSGSEVERITY MSGID   DEFAULT   - POSSIBLE -  REFERENCE
*-------------------------------------------------------------------------------
*        ENHOPT MSGSEVERITY_C1G0295=C            W/C       O000716
```

```
*       ENHOPT MSGSEVERITY_SMGR116=C          W/C/E     O000240/557
*       ENHOPT MSGSEVERITY_C1X0103=E            W/E       O000501
*       ENHOPT MSGSEVERITY_IMGR009=E            W/E       O000503
*       ENHOPT MSGSEVERITY_C1G0119=E            W/E       O000520
*       ENHOPT MSGSEVERITY_C1G0272=E            C/E       O000649
*       ENHOPT MSGSEVERITY_C1G0231=E            W/E       O000893
*       ENHOPT MSGSEVERITY_ENBX043=C            W/C       O002162
*       ENHOPT MSGSEVERITY_FPVL004=E            W/E       O002213
*       ENHOPT MSGSEVERITY_C1G0507=C            W/C       O002677
*       ENHOPT MSGSEVERITY_C1G0507=C            C/E       O002833
*       ENHOPT MSGSEVERITY_ENBE027=W            I/W       O002329
*       ENHOPT MSGSEVERITY_C1G0336=E            W/E       O002816
```

Question: If I add a new environment to my C1DEFLTS Table and I am using the Element Catalog, what do I have to do to activate this environment?

Answer: You don't have to do anything. The first time that Endevor is invoked after you have added the environment to the C1DEFLTS Table, the MCFs will be updated with the Element Catalog name in the stage record. From that point on, Endevor will check to ensure that the Catalog name in the MCF and the C1DEFLTS are the same. If the names are not the same, the MCF open will fail. This check has been added to prevent MCFs from belonging to more than one catalog.

Question: How do I make sure all my changes are added in the right order? For example, how can I ensure that my copybooks are added before my source program?

Answer: If you are performing your actions in foreground, the order depends on you. You have to make sure you add all your copybooks first, then your subroutines, and finally your main program. In batch and in packages, Endevor follows the type sequence defined to each system/stage as established by your Endevor Administrator. By default, the type sequence is defined in numerical order based on the order in which the types were defined to Endevor. The first type to be defined is placed at the top of the list; the last type to be defined is placed at the bottom of the list. The Endevor Administrator has the ability to adjust the numerical assignments on the Type Sequence panel. For detailed information on how type sequencing works see the *SCL Reference*. For information on adjusting the Type Sequence order, see the *Administration Guide*.

Question: Are the Element Catalog and Element Registration the same thing?

Answer: No, the Element Catalog and Element Registration are not the same thing. In fact, they are not related. Element Registration is an optional feature of Endevor that enables you to choose whether you want to restrict the use of the same element name at the subsystem or processor group level. The Element Catalog, which is required when upgrading to Release 4.0, is a VSAM file that allows Endevor to support long element names (HFS/UNIX). For more information on these new features, please refer to the *Release Summary*.

Question: I just upgraded from 3.8. What is ENTERPRISE_PKG=A, in the ENDICNFG Table?

Answer: This field will be used by a new product called AllFusion Change Manager Enterprise Workbench. In the ENDICNFG table, which is located in the *iprfx.iqual*.SOURCE library delivered with Endevor, the option ENTERPRISE_PKG= has three possible values:

A: List all packages (Enterprise and Endevor)

E: List Enterprise packages only

X: Exclude Enterprise packages from list

Question: Do I need to reassemble my API and/or Exit programs written for earlier releases?

Answer: Yes, is it recommended that all exits and API Programs always be reassembled using the current release macros.

Question: Is the API upwardly compatible?

Answer: No, it must be reassembled using the current release macros.

Question: I am upgrading from 3.7.2. Do I need to make any changes to my security table?

Answer: The tables format1, format2, format3, format4 and format5 have changed. Please see the *Security Guide* for details.

Question: Is AllFusion Change Manager Enterprise Workbench required for long name elements?

Answer: Yes. USS long file names can be maintained via SCL. AllFusion Change Manager Enterprise Workbench is required for online list, display and action processing. For more information on this product please contact your CRM.