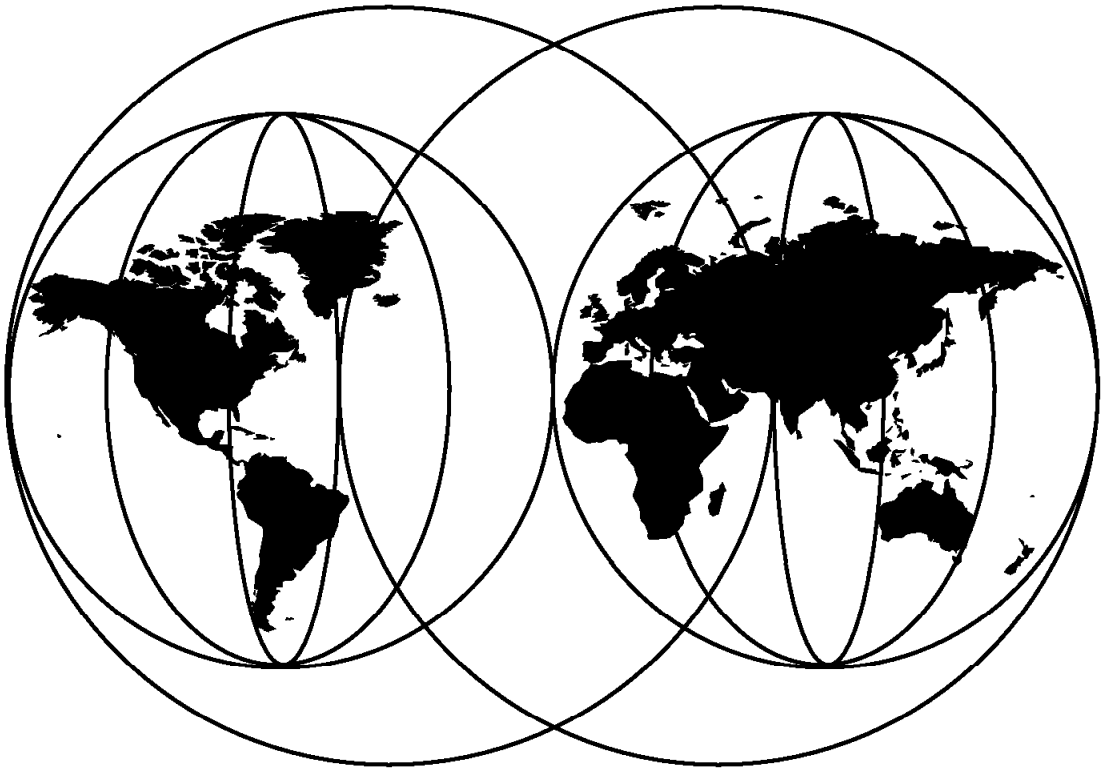
VisualAge 2000: Facilitating Find and Fix in an OS/390 Environment

Andrea Conzett, Eric Privette, Rina Sumiantoro



International Technical Support Organization

http://www.redbooks.ibm.com

SG24-5253-00

International Technical Support Organization

SG24-5253-00

VisualAge 2000: Facilitating Find and Fix in an OS/390 Environment

December 1998

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix B, “Special Notices” on page 253.

First Edition (December 1998)

This edition applies to:

- Version 1 Release 3 of Maintenance Tool Solution for Application Year 2000 Problem (MA2000, 5655-A33) for OS/390
- Version 2 Release 1 of COBOL and CICS Command Level Conversion Aid for OS/390 & MVS & VM (CCCA, 5648-B05)
- Version 2 Release 1 of IBM COBOL for OS/390 & VM (5648-A25) with VisualAge COBOL Millennium Language Extensions for OS/390 & VM (MLE, 5648-MLE)

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. QXXE Building 80-E2
650 Harry Road
San Jose, California 95120-6099

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Tables

Preface

Vast numbers of otherwise fully functional application programs are currently unable to interpret the abbreviated date values they contain as belonging to any century but the twentieth. The need to approach the daunting task of identifying, analyzing, and eliminating this limitation in a systematic and efficient manner has led IBM to offer a complementary assortment of products and services collectively known as VisualAge 2000. The guidelines, products, and services that constitute VisualAge 2000 are intended to support customers in their own efforts to tackle the problems posed by the use of abbreviated date fields in their application programs.

This book examines two of the products available in the VisualAge 2000 collection designed to address the requirements of system programmers and application developers engaged in identifying and correcting abridged date fields in their application programs in an OS/390 environment:

Maintenance 2000
Assists in Year 2000 impact analysis of host applications.

The Millennium Language Extensions to COBOL for OS/390 & VM
Provides automatic century windowing for specified date fields.

In addition to providing an introduction to the functionality available within Maintenance 2000, this book uses a sample application as the basis for a concrete demonstration of how the product can be used to perform the tasks of date identification, Year 2000 impact analysis, and result management. The actual results of the Year 2000 impact analysis process are then used as the basis for the correction of date-related problems in the sample application programs using the new facilities of the COBOL compiler known collectively as the Millennium Language Extensions. Three specific approaches to the solution of date-sensitive problems in the sample application programs are presented in order to show how the new enhancements to the COBOL compiler can be used to assist in a gradual and incremental process of date field expansion in your application inventory.

The process of taking the results provided by Maintenance 2000 and applying the corresponding fixes to the application programs by means of the Millennium Language Extensions can be further automated by using

COBOL and CICS Command Level Conversion Aid Version 2.1.

COBOL and CICS Command Level Conversion Aid is a tool that can assist with conversion from earlier levels of COBOL. Its primary function is to convert source programs from COBOL 68 or 74 Standard to COBOL 85 Standard. An additional function of COBOL and CICS Command Level Conversion Aid is the ability to add Millennium Language Extensions syntax to the source program. This function is known as the *DATE FORMAT conversion option* and requires specific input that can be provided by a tool like Maintenance 2000.

Because COBOL and CICS Command Level Conversion Aid represents an important link between Maintenance 2000 and the Millennium Language Extensions this book also provides an introduction to the DATE FORMAT conversion option. However, since the main functionality of the tool is not

specifically related to Year 2000, COBOL and CICS Command Level Conversion Aid is not being discussed in its entirety.

The Team That Wrote This Redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization San Jose Center.

Andrea Conzett is an Application Development Specialist at the International Technical Support Organization, San Jose Center. He writes extensively and teaches IBM classes worldwide on all areas of COBOL and PL/I application development as well as on all application development related issues of the Year 2000 challenge. He has 13 years of experience in application development from both IBM internal and customer projects. Before joining the ITSO, Andrea worked in IBM Switzerland as Product Manager for all application development tools.

Eric Privette is an application development specialist in IBM Global Services in Switzerland. He is currently engaged in Year 2000 customer projects. He has 12 years of experience in host application development.

Rina Sumiantoro is an application development specialist in IBM Global Services in Indonesia. She is currently engaged in Year 2000 customer projects. She has seven years of experience in host application development.

Thanks to the following people for their invaluable contributions to this project:

Tom Ross
IBM Santa Teresa Laboratory, COBOL Development

Nicholas Tindall
IBM Santa Teresa Laboratory, COBOL Development

Hiroko Fujita
IBM Yamato Laboratory, AD Solution Development

Keith Warltier
IBM EMEA Year 2000 Centre of Competence

Thanks also to **Merrill Bani**, IBM Australian Programming Center, who developed the sample application that we use in this book.

Comments Welcome

Your comments are important to us!

We want our redbooks to be as helpful as possible. Please send us your comments about this or other redbooks in one of the following ways:

- Fax the evaluation form found in “ITSO Redbook Evaluation” on page 263 to the fax number shown on the form.
- Use the online evaluation form found at `http://www.redbooks.ibm.com/`
- Send your comments in an Internet note to `redbook@us.ibm.com`

Chapter 1. An Overview of the Sample Application

A sample application comprising five COBOL programs forms the basis of all the Year 2000-related activities documented in this book. This application performs the simple task of maintaining a master file and is designed to be relatively transparent and straightforward so that attention can be focused on the products and activities described rather than on the mechanics of the application itself.

See Appendix A, “Files Used in This Book” on page 249 for a detailed list of all the code samples used in the book and for information about how you can get the sample code for your reference or in order to install it on your system.

The centerpiece of the application is an indexed VSAM file containing employee data which contains the following information about each employee:

- Employee identification number which serves as the record key.
- Department code
- Name
- Address line 1
- Address line 2
- Address line 3
- Zip code
- Date on which the employee joined the organization (format YYDDD)
- Date on which the employee left the organization (format YYMMDD)
- Date on which the master file was last modified (format YYDDD)
- Date of birth (format YYDDD)
- Date on which the employee’s security pass expires (format YYDDD packed decimal)

In addition to the employee master file, a file containing departmental information is provided. This file is used for reference and is not maintained by the sample application. This VSAM file includes the following information:

- Department code
- Department description
- Date on which information was last updated (format YYDDD packed decimal)

These files are processed by the following programs:

TARMU3
: This interactive CICS program is used to maintain the employee master file. This program performs validation of on-line input and uses it to modify existing employee records or to add new ones. Subprogram TARDTE3 is invoked to translate the date formats used interactively and those stored in the master file.

TARMU3B
: This batch program performs the same tasks as those performed by TARMU3 but uses a sequential input file as the source of the insertions or updates to the employee master file.

TARMDTE3

This subprogram is used to translate between the date patterns stored on the master file and those used on-line or in the two batch reports produced by programs TARMU5 and TARMU6.

TARMU5

This batch program reads the employee master file, invokes subprogram TARDTE3 to translate the date formats of the dates stored on the employee record into different formats, and produces a report file containing employee data.

TARMU6

This batch program reads the employee master file and examines the security expiration date of all active employees to determine if their security pass has expired. If the expiration date on the employee record antedates the current date, the expiration date is automatically extended by 1 year and the master file is appropriately updated. A report file containing information about employees whose security expiration date was updated is created as well. Subprogram TARDTE3 is also called in this instance to translate between the date formats stored on the employee file and those required in the report.

Chapter 2. Maintenance 2000 in the VisualAge 2000 Product Suite

The IBM VisualAge 2000 offerings (VA2000) are focused on assisting customers who are interested in implementing their own solutions to the Year 2000 challenge. VA2000 consists of a suite of products from both IBM as well as carefully chosen vendors, coupled with education and technical support, for the Year 2000 solution. VA2000 is populated with complementary cross-platform, language-independent tools intended to address and facilitate all phases of the Year 2000, which include:

1. Inventory and assessment
2. Analysis
3. Finding and fixing date exposures
4. Testing the modified data and application logic
5. Integrating them into a production environment

IBM Maintenance 2000 Version 1.3 (MA2000), intended to address the requirements of system programmers and application developers on the OS/390 platform, is included in VA2000 in order to expedite and accelerate the date-impact analysis process, which represents a critical element of the FIND and FIX phase of the Year 2000 solution.

2.1 Maintenance 2000 System Requirements

This section describes the hard-ware and software requirements for MA2000.

2.1.1 Hardware Requirements

MA2000 operates on any processor supported by MVS/ESA or OS/390.

It has special requirements regarding the availability of sufficient space on direct access storage devices. Refer to 2.3.1.4, “Physical Resource Allocation Planning and Allocation” on page 14 for a further discussion of this subject.

2.1.2 Software Requirements

MA2000 operates on **one** of the following operating systems. Unless explicitly stated otherwise, later releases are also supported.

- MVS/ESA System Product-JES2 Version 4 Release 1.0 (5695-047)
- MVS/ESA System Product-JES3 Version 4 Release 1.0 (5695-048)
- MVS/ESA System Product-JES2 Version 5 Release 1.0 (5655-068)
- MVS/ESA System Product-JES3 Version 5 Release 1.0 (5655-069)
- OS/390 Version 1 Release 1.0 (5645-001)
- OS/390 Version 2 Release 4.0 (5647-A01)

MA2000 requires the following software products. Unless explicitly stated otherwise, any of their later releases is also supported.

- System Modification Program Extended (SMP/E) Version 1 Release 8.0 (5668-949)
- TSO Extensions (TSO/E) Version 2 Release 3.0 (5685-025)

- Interactive System Productivity Facility (ISPF) Version 2 Release 3.0 (5665-319) **and** ISPF/Program Development Facility (ISPF/PDF) Version 2 Release 3.0 (5665-317)

  or

  Interactive System Productivity Facility (ISPF) for MVS Version 3 Release 2.0 (5685-054) **and** ISPF/Program Development Facility (ISPF/PDF) for MVS Version 3 Release 3.0 (5665-402)

  or

  Interactive System Productivity Facility (ISPF) Version 4 Release 1.0 (5655-042)
- Assembler H Version 2.1 or Hi-level Assembler Version 1.1
- IBM C/370 Runtime Library Version 2 Release 1.0 (5688-188) **and** OS PL/I Library Version 2 Release 3.0 (5668-911/5669-909/5668-910)

  or

  IBM SAA AD/Cycle Language Environment/370 (LE/370) Version 1 Release 1.0 (5688-198)
- Data Facility Sort (DFSORT) Version 1 Release 11.0 (5740-SM1)
- If you have a PL/I application source to be analyzed, any release of the OS PL/I Optimizing Compiler able to compile this source is required.

Prolog is also a software prerequisite. But it is shipped with MA2000 and need not be ordered separately. When you install MA2000, Prolog is installed automatically with MA2000 product data set names. However, if you already have a version of Prolog installed (for example AD/Cycle Prolog Version 1 Release 2) you can use it.

2.2 Functional Overview of Maintenance 2000

MA2000 is a TSO application whose primary processes are executed as batch jobs. The principal batch processes that compose MA2000 include:

- Analysis and data dictionary creation functions
- Retrieval functions
- Retrieval result management functions

The analysis and data dictionary creation functions involve using program source and Job Control Language (JCL) code stored in partitioned data sets as input to a series of sequential batch jobs. These batch jobs create intermediate analysis data that is subsequently used as input to the respective data dictionary creation process. The data dictionaries in turn serve as the basis for a wide variety of batch impact analysis and data retrieval functions submitted by the MA2000 user in the on-line ISPF environment. The results of the impact analysis and data retrieval functions are conveniently accessible by means of on-line display functions, as host output files or as export files suitable for processing on a workstation for coding and test management purposes.

2.2.1 MA2000 Analysis and Data Dictionary Creation

The batch analysis and data dictionary creation functions are available for the following types of source code:

- COBOL source programs and copybooks
  - VS COBOL II plus the OS/VS COBOL compiler-level EXAMINE and TRANSFORM statements; the COBOL intrinsic functions as well as the DATE FORMAT clause available with the current IBM COBOL compilers (IBM COBOL for MVS & VM, IBM COBOL for OS/390 & VM) are also supported.
  - Supported COBOL statements include:
    - Data definition statements (length, offset, and redefinition statements)
    - Data assignment statements (MOVE, READ, and CALL)
    - Data reference statements (IF and DO)
  - EXEC SQL, EXEC CICS, EXEC DLI and CALL CBLTDLI statements
- PL/I source programs (after %INCLUDE statements and macros have been subjected to preprocessing)
  - OS PL/I V1R5M1 or OS PL/I V2 or later, presuming that new functions in Version 2 have not been used
  - Supported PL/I statements include:
    - Declare statements (length, offset, based and defined items)
    - Data assignment statements (A=B, READ, and CALL)
    - Data reference statements (IF and DO)
  - EXEC SQL, EXEC CICS, EXEC DLI, and CALL PLITDLI statements
- CA-Easytrieve Plus programs and macros
  - CA-Easytrieve Plus R6.2
  - Supported statements include:
    - Declare statements
    - Data assignment statements
    - Data reference statements
- JCL and cataloged procedures
  - MVS/ESA SP R4
  - Supported statements include:
    - JOB
    - DD
    - EXEC
    - PEND
    - Cataloged and in-stream PROC statements
  - The standard utilities listed below are automatically analyzed during JCL analysis:
    - SORT
    - DFSORT

- ICEMAN
- IEBCOPY
- IEBGENER
- ICEGENER
- IDCAMS

- For a complete list of COBOL, PL/I, CA-Easytrieve and JCL elements which are not supported by MA2000 Version 1.3, please refer to the *Maintenance 2000 Guide and Reference Version 1, Release 3*.

2.2.2 MA2000 Retrieval Functions

MA2000 provides an extensive array of data retrieval functions which can be classified according to the number of starting-points used as input to the retrieval process. The two types of retrieval functions provided by MA2000 can be described as follows:

Individual Retrieval Retrieval functions that restrict themselves to investigating the impact of one individual item

Multiple Retrieval Retrieval functions that base their analysis on more than one starting-point item

2.2.2.1 Individual Retrieval Functions

On the program level, the individual retrieval functions investigate the impact of a single starting-point. In addition, there are some cross-reference functions on the application level.

The individual retrieval function consists of the following components:

- The search function
- The program correlation chart
- The cross-reference list
- The individual impact analysis

The search function: The search function is one of the most powerful features available in MA2000 for data retrieval. Depending upon the number and precision of the arguments supplied to the search function, a large volume of data can be returned and used subsequently to construct a starting-point file to be used in turn as input to a multiple retrieval function such as the multiple impact analysis function or the Year 2000 analysis function. The scope of the retrieval to be performed in the search function can be limited by the specification of a domain, to indicate what programs and jobs are to be the targets of the search. The following types of searches can be performed by the search function:

- The *similar item* search function performs a search of COBOL, PL/I, and CA-Easytrieve source programs or JCL for data items with a name similar to patterns specified by the user in the form of wild cards.
- The *synonym* retrieval component of the search function is used to retrieve data definitions of items that overlay or redefine data items supplied by the user as input and therefore have the same content.

- The *DATE function* retrieval option automatically returns program data items whose value is set directly by the use of a date-related function in a program.

The program correlation chart: The program correlation chart function is used to graphically illustrate the relationship between program modules. Retrieval options allow the user to specify whether the hierarchical relationship between calling, called, or both types of modules should be displayed in the output report.

The cross-reference list: The cross-reference list function produces a cross-reference list of related external resources accessed by any or all programs belonging to an MA2000 subsystem or where any or all includes, copybooks, macros, input and output segments, data sets, DB2 tables, or CICS components such as maps, files, transactions, and queues within a subsystem are used.

The individual impact analysis: The individual impact analysis function analyzes the impact of individually specified starting-point items in order to help the user understand existing application programs. This function focuses on data flow by analyzing program and JCL source files and identifying the operations performed upon the specified data as well as identifying other data that is synonymous with it. One of the following items may be specified as the starting-point option of the individual impact analysis function, either by typing it in directly or by choosing it from an on-line display:

- An individual data item name in a specified program
- An individual data item selected from one of the major structures of a specified program
- An individual data item name selected from one of the file input/output major structures of a specified program
- An external call parameter contained in a specified program
- An individual data item which is the target of a date-related function in a specified program
- A data set name
- An IMS DB segment name
- The name of a DB2 table or view column
- A CICS file name

After having chosen the item to be used as the starting-point for the individual impact analysis function, the user must decide which of four available analysis methods to apply to that starting-point. The analysis methods available for selection are:

Cause analysis Identifies data items that may cause the value of the specified starting-point item to change.

Influence analysis Identifies data items whose value may be influenced by the value of the specified starting-point item.

Equivalency analysis Retrieves data that is synonymous with the specified starting-point item, thereby highlighting elements that may require modification if the starting-point item itself is modified. While the cause analysis and influence analysis monitor the flow of a specific data item (value oriented),

the equivalency analysis also identifies other variables that are not directly impacted but might contain the same type of data as the starting-point item (type oriented).

Year 2000 analysis Performs an equivalency analysis. In addition to the starting-point item provided by the user, the year 2000 analysis automatically adds those data items to the list of starting-points that have been the target of an assignment statement involving the value returned by a program date-related function.

Figure 1 on page 9 explains what items will be retrieved by what type of analysis:

- **SP** is the starting-point item. It is always retrieved.
- **Cn** are items retrieved by the cause analysis as well as the equivalency or Year 2000 analysis.
- **In** represent items retrieved by the influence analysis. as well as the equivalency or year 2000 analysis.
- **En** are items retrieved by the equivalency or Year 2000 analysis. The difference between the equivalency and the Year 2000 analysis lies in the starting-point items and not in the way the analysis is performed.

The arrows can represent any kind of assignment. This can be a MOVE statement in COBOL, a READ or WRITE operation, or a database access.

The items in Figure 1 on page 9 do not have to be in the same program. The arrows can also represent a link through a data set, which is reflected in the JCL. This would be the case for example if program A writes from item **E3** to position n in data set D, and program B reads from position n in data set D into item **C2**.

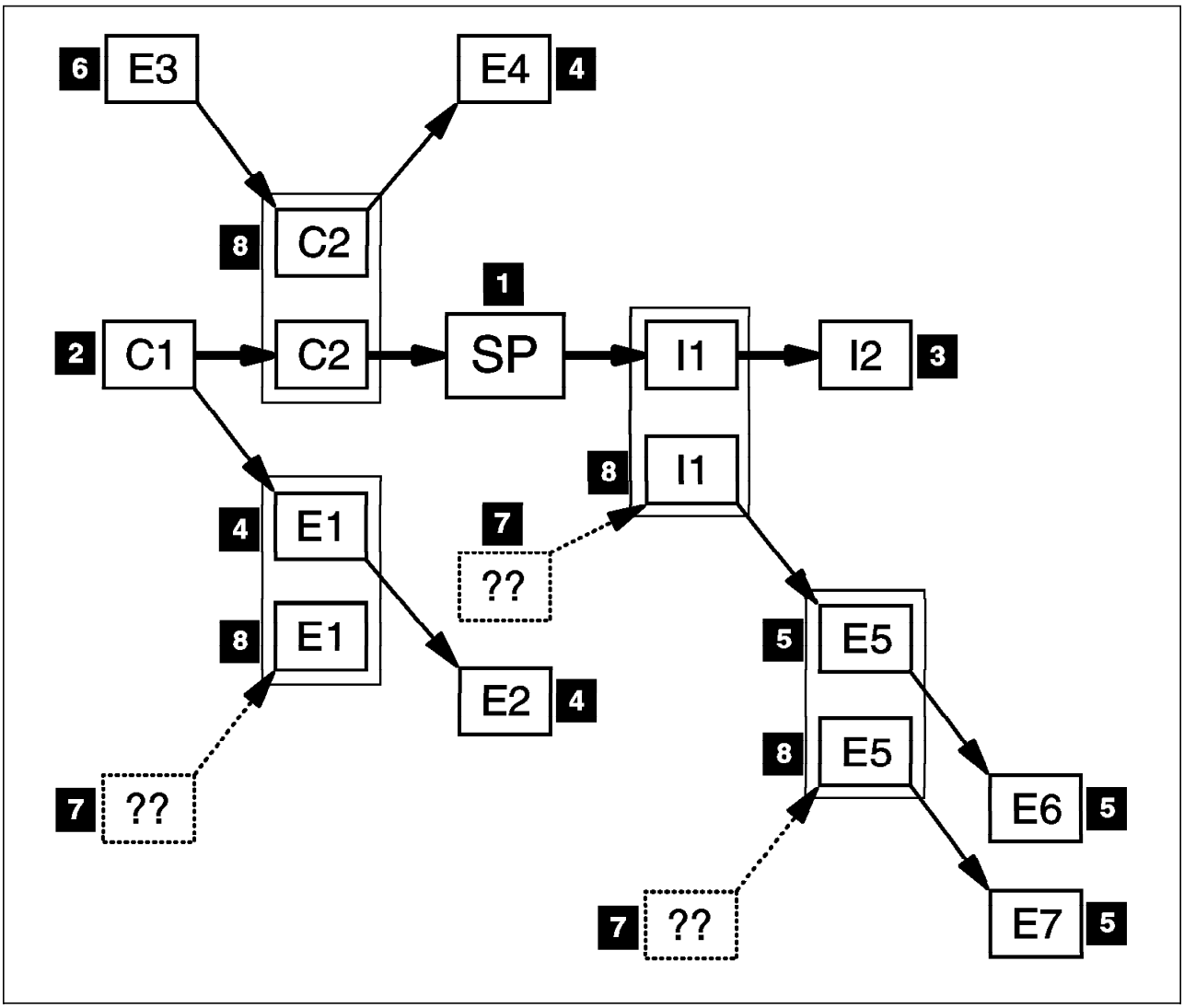


Figure 1. Maintenance 2000 Impact Analysis

1 The starting-point is either a basic program variable or a specific position within an input/output element such as a file record, DB2 table row, or IMS segment. If it is not a program variable, the impact analysis process first maps the position of the input/output element to a program variable and takes this as the starting-point. The retrieved items are always program variables or literals.

The Year 2000 analysis may add additional variables as starting-points. It actually performs a multiple impact analysis (see 2.2.2.2, “Multiple Retrieval Functions” on page 10).

2 A change of C1 and/or C2 directly causes the starting-point to change. C1, C2, and SP form a causal chain. In the next explanatory items we use the term *causing items* for items on the starting-point side of an arrow to another item.

3 A change of the starting-point has a direct influence on I1 and I2. SP, I1, and I2 form a causal chain. In the next explanatory items, we use the term *influenced items* for items on the ending point side of an arrow from another item.

4. The equivalency analysis retrieves items influenced by items that are causing items in regard to the starting-point, even if these items are not part of the causal chain.

5. The equivalency analysis retrieves items influenced by items that in turn are influenced items in regard to the starting-point even if these items are not part of the causal chain.

6. The equivalency analysis retrieves items that are causal items for other items, which in turn are causal items in regard to the starting-point even if these items are not part of the causal chain.

7. The equivalency analysis does not retrieve items that are causal items to items that in turn are influenced items in regard to the starting-point.

8. Items that are part of a causal chain can be used in a different context of the program in a way that interrupts the causal chain.

   In the following COBOL example, after execution of line 000313 the variable INFLU-1 no longer contains the same value as the starting-point START-P. Therefore, INFLU-3 and INFLU-4 are no longer in the same causal chain with the starting-point.

   If the value of the starting-point changes, INFLU-1 is impacted but INFLU-3 and INFLU-4 are not. But if the format or data type of the starting-point changes, then not only the items in the causal chain but also INFLU-3 and INFLU-4 may have to be modified. For example, if the starting-point is a date that is being expanded to a four digit-year, INFLU-3 and INFLU-4 probably also contain date data and need to be analyzed for Year 2000 impact.

```
  ⋮
000307     MOVE CAUSE-1 TO CAUSE-2.
000308     MOVE CAUSE-2 TO START-P.
000309
000310     MOVE START-P TO INFLU-1.
000311     MOVE INFLU-1 TO INFLU-2.
000312
000313     MOVE "981010" TO INFLU-1.
000314     MOVE INFLU-1 TO INFLU-3.
000315     MOVE INFLU-3 TO INFLU-4.
  ⋮
```

For a more detailed discussion of the four impact analysis methods available in MA2000, please see Chapter 5.6, “Impact Analysis,” in the *Maintenance 2000 Guide and Reference Version 1, Release 3*.

2.2.2.2 Multiple Retrieval Functions

In contrast to the individual retrieval functions, the multiple retrieval functions investigate the impact of multiple starting-point items. These retrieval functions require the use of a starting-point file, which may contain numerous starting-point items of various types. The starting-point file may be created by using a combination of the results obtained by the use of the individual retrieval search function or by invoking the starting-point file creation function in the on-line MA2000 environment. No matter which approach is taken, however, the initial contents of the starting-point file will require close examination and editing before a multiple retrieval function can be invoked in order to prevent unwanted or duplicate items being used as starting-points.

The multiple retrieval functions include:

- Multiple impact analysis
- Year 2000 analysis

Multiple impact analysis: The multiple impact-analysis function applies the same analysis methods of cause, influence, and equivalency analysis to multiple starting-points as the individual impact analysis function applies to individual starting points. Please refer to 2.2.2.1, “Individual Retrieval Functions” on page 6 for more detailed information.

Year 2000 analysis: The Year 2000 analysis function investigates the impact of multiple date-related starting-point items. The method of tracing is essentially the same as that performed by the equivalency impact analysis function (see 2.2.2.1, “Individual Retrieval Functions” on page 6), with the addition of an automatic DATE function recognition feature which implicitly recognizes those items involved in program date-related functions as starting-point for analysis in addition to any starting-points which the user may have explicitly supplied.

In addition to the usual summary and detail reports, the Year 2000 analysis function generates a Date Identification File (DIF) if the corresponding option is selected. This DIF is used as input to the CCCA tool which automatically adds Millennium Language Extensions syntax to the program source.

2.2.3 MA2000 Retrieval Result Management Functions

The retrieval result management function displays, prints, or exports the results of the individual and multiple retrieval functions in various forms as well as providing batch job management functions. Retrieval management functions include:

- Deferred job management functions which allow deferred jobs to be submitted for processing or deleted. A deferred job is one for which the execution JCL has been generated by MA2000, but which has not yet been submitted by the user.
- Both summary- and detail-level reports of the results of individual and multiple retrieval functions can be displayed on-line or printed by means of batch jobs generated automatically for this purpose.
- Files suitably formatted for subsequent processing by a spread sheet program on the workstation can be produced by means of a batch export file creation process.
- The retrieval results of the Year 2000 analysis process executed by individual users can be amalgamated by means of an accumulation function which merges them into a system-wide result data set. These accumulated results may then be printed or exported to various types of file formats selectively or collectively by component type.

2.3 Outline of the Maintenance 2000 Task Flow

This section describes the sequence of steps required to plan for, install, customize, and use MA2000 effectively. The steps to be performed reflect the three basic types of tasks involved in the operation of MA2000:

- Manual tasks performed outside the scope of the MA2000 environment
- Batch processes executed to establish the MA2000 environment

- Iterative interactive tasks performed within the MA2000 on-line environment

See Figure 2 on page 13 for a more detailed outline of the activities which compose each of these three basic categories of tasks. This figure presents a broad outline of the procedure which we recommend should be followed when using MA2000. These diverse tasks are obviously not intended to be the responsibility of any one individual, but rather represent a broad spectrum of roles, ranging from that of the system administrator to application developers in an OS/390 environment.

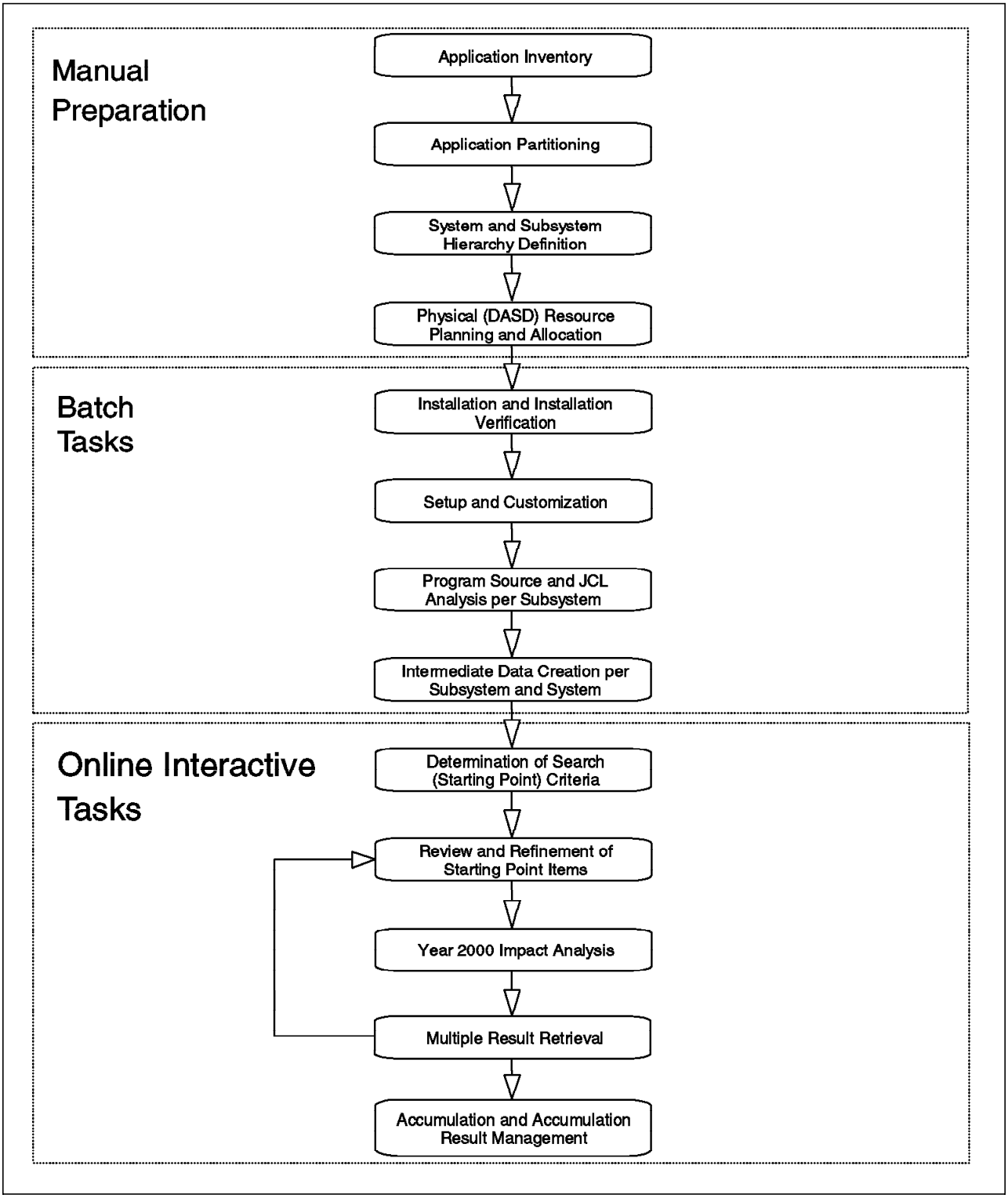


Figure 2. The Maintenance 2000 Task Flow

2.3.1 Manual Preparatory Tasks

This section provides a brief explanation of the tasks illustrated in Figure 2 on page 13. While we in no way intend to minimize the importance of the vital manual preparatory work performed outside the scope of the MA2000 environment, we do not treat these tasks comprehensively here as most are only marginally related to the specific operation of MA2000. For a general description of the VA2000 methodology and its phases, we invite you to consult URL http://www.software.ibm.com/ad/va2000/y2k.

2.3.1.1 Application Inventory

In relation to MA2000, we understand the application inventory task not to be the preliminary one of identifying what applications are currently running in a production environment, but rather one of identifying those relevant applications that contain potential date exposures and have therefore been earmarked as candidates for detailed date impact analysis. The applications identified by what we term the *application inventory process* therefore represent a subset of those applications that might normally result from an initial application inventory process.

2.3.1.2 Application Partitioning

Application partitioning involves performing a further segmentation of the body of applications collectively identified in the previous step into groups whose size does not exceed a recommended total of 2,000 programs and 1,200 JCL members. As the execution time required for MA2000 analysis and retrieval functions is directly proportional to the number of input programs and related resources in a single system, it is advisable to restrict the maximum number of components per system to the suggested perimeters. For more information regarding systems and subsystems within MA2000, please refer to 2.3.1.3, “System and Subsystem Hierarchy Definition.”

2.3.1.3 System and Subsystem Hierarchy Definition

This task represents the further subdivision of each aggregate of programs and JCL/procedure members which resulted from the application partitioning process into what is known in MA2000 terminology as a *system*. A system, which represents nothing other than a cohesive group of user software resources, acts as the basic operational unit of MA2000. Systems can be composed of one to three subsystems, among which there exists a hierarchical relationship. For further information, see 3.1, “System and Subsystem Hierarchy Definition” on page 19.

2.3.1.4 Physical Resource Allocation Planning and Allocation

Resource planning and allocation deals primarily with the estimation of direct access storage device (DASD) resources required for the following three items:

- The MA2000 product itself, together with the installation verification program
- The intermediate data produced by the program source and JCL analysis process
- The retrieval results produced by the various retrieval functions

The following approximations of required resources are intended to serve only as guidelines to aid the system administrator in the estimation of amount of DASD space necessary. Please note that these are intended to serve only as a general rule of thumb and that the actual amount of space required will differ

according to your own environment. The DASD space requirements can be estimated as follows:

- Approximately 400 tracks of DASD space are required for the MA2000 product and the installation verification program.
- For the intermediate data produced by the program source and JCL analysis process the following approximations apply:
  - For the data produced by the analysis of COBOL, PL/I, and CA-Easytrieve source, the DASD requirements are five to eight times as large as the original program source data sets.
  - For the data produced by the analysis of JCL and cataloged procedures, the DASD requirements are one to two times as large as the original source data sets.
- The DASD requirements for the retrieval results data depends upon the retrieval function which was performed and the size of the retrieval target.

2.3.2 Batch Tasks

This section describes the batch jobs which are a prerequisite for the use of the retrieval functions in the on-line MA2000 ISPF environment. With the exception of the tasks connected with the installation of the MA2000 product itself, all of the activities here presuppose the definition of the system and subsystem hierarchies discussed briefly in 2.3.1.3, “System and Subsystem Hierarchy Definition” on page 14, testifying to the critical role of this activity in the effective use of MA2000.

2.3.2.1 Installation and Installation Verification

The installation of the MA2000 product is performed by means of SMP/E. For more information regarding the installation and installation verification procedure, please refer to the documentation shipped with the product.

2.3.2.2 Setup and Customization

The setup and customization process involves the definition of parameters which are used to establish the two primary functional environments of MA2000:

- **The analysis environment** — Parameters required to build the MA2000 analysis environment are defined according to your operational system environment and apply to every instance of COBOL, PL/I, CA-Easytrieve, and JCL analysis. While some parameters, such as that defining the severity level of messages to be displayed, are shared by all the analysis processes, others are unique to the analysis function being performed. These system-wide parameters are to be found in the environment parameter IMX.V1R3M0.SIMXENV data set, as well as in the system parameter IMX.V1R3M0.SIMXPRM data set created during the installation process.

  Note

  ″IMX.V1R3M0″ represents the high level and second level qualifiers we used for the MA2000 system data sets when we installed the product. Depending on the naming conventions in your installation, these qualifiers might be different. However throughout this redbook ″IMX.V1R3M0″ always refers to the MA2000 product data sets.

- **The retrieval environment** — Parameters required to set up the MA2000 retrieval environment can be categorized according to their scope:

- Parameters specified in the environment parameter data set and in the system parameter data set are effective for all systems defined during the system and subsystem hierarchy definition process.
- Parameters specified in a member of the system identification data set IMX.V1R3M0.SIMXID are applicable to only the system and subordinate subsystems in question.

For more detailed information regarding the specification of parameters for the analysis and retrieval environments, please see 3.2, “MA2000 System and System ID Parameter Definitions” on page 22.

2.3.2.3 Program Source and JCL Analysis per Subsystem

Program source code and JCL analysis consists of a series of batch jobs and is performed for each system and subsystem ID combination. Separate JCL to execute the analysis process is provided for those instances when a particular system has been defined as having one, two, or the maximum of three subsystems.

COBOL and PL/I source code analysis automatically includes the analysis of EXEC SQL, EXEC CICS, EXEC DLI and CALL CBLTDLI and CALL PLITDLI in the respective source programs. If a standard utility step is encountered during JCL analysis, the utility analysis function is automatically invoked to track data flow from an input to an output record. For more information regarding these automatic analysis functions, please see Chapter 3, “Using MA2000 to Analyze Program and JCL” in the manual *Maintenance 2000 Guide and Reference Version 1, Release 3*.

The output of the analysis process is placed in the intermediate analysis buffer for further processing by the respective intermediate data management function.

2.3.2.4 Intermediate Data Creation for a Subsystem and System

This series of batch jobs begins with the data that resulted from the successful analysis of program source code and JCL, and concludes with the creation of the program source and JCL dictionaries, which are the prerequisites for the retrieval functions. Analysis output data at the subsystem level is first stored in intermediate data sets which are in turn used to create the dictionaries for intermediate data at the system level. If more than one subsystem is defined as belonging to the system in question, then the hierarchical priority of the individual subsystems must be specified before executing the JCL that creates a dictionary.

2.3.3 On-Line Interactive Tasks

This section describes those tasks performed within the context of the on-line ISPF environment of MA2000. All the input options available on the MA2000 main menu panel represent functions performed by batch jobs activated to process user input. It should come as no surprise that no data can be returned by the on-line retrieval functions without the specification of a retrieval view, representing a system and subsystem ID pair. This attests once again to the pivotal role played by the system-subsystem relationship within MA2000 processing.

2.3.3.1 Determination of Search Criteria

Input for the starting-point file for the Year 2000 analysis can be created by invoking the starting-point file creation option on the MA2000 main menu panel, or by repeatedly invoking the assorted types of search functions available within the individual retrieval search function. When determining the initial search criteria to be used as input to the Year 2000 analysis of our sample application, we performed the following steps:

1. Wild-card search using the similar item search function
2. Date item search using the DATE function search option
3. Synonym search using the output of the previous two steps after it has been edited to eliminate duplicate data items
4. Data set name search using wild cards constructed on the basis of the output resulting from the invocation of the data set option of the create starting-point file main menu option.

2.3.3.2 Review and Refinement of Starting-Point Items

The amalgamated results of the various individual types of searches performed in order to construct a starting-point file must be carefully reviewed before undertaking an initial Year 2000 analysis. Do this in order to discard any unwanted or inappropriate data items or data set names returned by the search functions. This is particularly true in cases where wild cards were specified as search criteria for pattern matching.

2.3.3.3 Year 2000 Impact Analysis

Year 2000 impact analysis is performed using the multiple starting-point file created by editing the combined results of the various individual searches.

The output of the Year 2000 Impact Analysis consists of:

- **Summary report** — A list of all the starting-point items and all the impacted or equivalent items, optionally with the date formats of these items added.
- **Detailed report** — A list of all the starting-point items and all the impacted or equivalent items, saying where these items are defined and where and how they are used. The lines of source code and JCL are also included.
- **Date Identification File** — A list of all the starting-point items and all the impacted or equivalent items with the date formats of these items added. The Date Identification File (DIF) is created with the format required by CCCA as input for its DATE FORMAT conversion option.

2.3.3.4 Multiple Result Retrieval

The results of the initial Year 2000 analysis must be reviewed in order to determine if starting-point items originally used as input should be eliminated and the Year 2000 analysis process repeated. This sequence of retrieval, result review, and refinement of the starting-point criteria may require multiple iterations until the results of the Year 2000 impact analysis accurately reflect the date exposures in the system and subsystems concerned. The scope of the Year 2000 impact analysis may also be restricted or expanded as required by the specification of an appropriate subsystem. The results of the Year 2000 analysis can be displayed on-line or printed in the form of summary or detailed reports. The detailed report may also be written to a sequential file if required. The results file may also be exported to a file suitably formatted for further processing in the workstation environment.

2.3.3.5 Accumulation and Management of Accumulation Results

A retrieval-result management function unique to the results of multiple result retrieval is the *accumulation function*. In the case of the Year 2000 impact analysis, the result of each execution of the function is maintained as an individual user′s Year 2000 work result. This result can apply to one or more of the subsystems assigned to a system, or, even more specifically, to a particular subsystem domain. The accumulation function is performed when the user wishes to integrate a specific retrieval result with the Year 2000 system result data set. Once the individual results of Year 2000 analysis have been accumulated to the system result data set, the process cannot be reversed.

Note

The prefix of all the retrieval result data sets is set by the “#RETPFIX” parameter found in member SYS@ of the environment parameter data set IMX.V1R3M0.SIMXENV. If this parameter is left blank, the user ID is taken as the default for the highest-level qualifier of the retrieval result data sets. Otherwise, the #RETPFIX parameter is taken as the highest-level qualifier and the user ID is added as second-level qualifier. This construct prohibits retrieval result data sets from being shared among on-line users since the individual user ID always plays a role in the generation of data set names. The accumulation function is the only means by which an individual user′s work results can be made available to others investigating the system in question. Please note that only one user can execute the accumulation retrieval management function at any given time.

After the results have been accumulated, information about system resources such as programs, copybooks, and batch jobs can be obtained on an individual or collective basis and the results written to a sequential host file, to a workstation export file, or directly to a host printer.

Chapter 3. Maintenance 2000 Preparation Tasks

This chapter describes in detail the tasks we performed in order to set up the MA2000 environment against which we ran the on-line analysis jobs. The tasks include:

- Defining the MA2000 System and Subsystems
- Customizing MA2000 system parameters
- Creating the program and JCL dictionaries for our sample application (retrieval environment)
- Establishing the TSO and ISPF on-line environment

3.1 System and Subsystem Hierarchy Definition

This section provides a more detailed discussion of the system and subsystem relationship within MA2000. Systems and subsystems represent the basic operational units within MA2000 and, as such, form the basis for the analysis and retrieval functions. The relationship is as follows:

- Seven of the eight batch jobs that must be executed to establish the COBOL analysis and retrieval environment cannot be performed without the specification of both a system ID and at least one subsystem ID. Even the one job that does not require a subsystem ID as a parameter must nevertheless be supplied with a system ID in order to execute successfully.
- None of the four batch jobs that must be executed in order to establish the analysis and retrieval environments for JCL can be executed without the provision of a system ID and at least one subsystem ID.
- None of the input options available on the main menu in the MA2000 on-line environment can be selected for further processing without the simultaneous specification of a valid combination of system and subsystem ID.

3.1.1 Systems and Subsystems

In MA2000, the term *system* represents a cohesive collection of user software resources. A system must conform to the following requirements:

- A system must be defined as containing at least one subsystem. When a system has only one subsystem, the boundaries of the subsystem and system are identical.
- A system cannot contain more than three subsystems.
- One subsystem cannot belong to more than one system. Because a system is the basic operational unit of MA2000, no search can be performed across multiple systems.

3.1.2 The System and Subsystem Hierarchy

A hierarchical relationship exists among the subsystems defined as belonging to a system. This is best illustrated by Figure 3 on page 20. The system defined as “SYSITSO” is specified as the system to be searched and represents the object of the retrieval process. A subsystem ID representing the retrieval view specifies where the search is to be performed within the specified system. A maximum of

three different views of a system are provided according to the number of subsystems defined.

```
IMXP001------------------------------  Main Menu  ------------------------------
 Menu ===>
                                                              (USERID: ITSORS3 )
       Select an item from the menu, and then specify
       the system ID and the retrieval view:

     Individual retrieval:             Retrieval result management
      1  Impact analysis (individual)   7  Individual retrieval (1 - 4)
      2  Search                         8  Multiple retrieval   (5, 6)
      3  Program correlation chart      9  Accumulative result
      4  Cross-reference list
                                       Others
     Multiple retrieval                 A  Starting point file creation
      5  Impact analysis (multiple)     B  Deferred job
      6  Year 2000 analysis             X  Exit

     System ID       ==> SYSITS0
     Retrieval view  ==> SUBSYS1   (See note.)

     Note: Specify one of the subsystem IDs for the retrieval view.
```

Figure 3. The MA2000 Main Menu - System and Subsystem

The retrieval view is a reflection of the hierarchical relationship among subsystems. Given that the highest-level subsystem ID is SUBSYS1 as pictured in Figure 3, the second-highest subsystem ID is SUBSYS2, and the lowest ID is SUBSYS3, the following retrieval rules apply:

- If SUBSYS1 is specified as the retrieval view, as in Figure 3, the retrieval view represents the concatenation of SUBSYS1, SUBSYS2, and SUBSYS3.
- If SUBSYS2 is indicated as the subsystem-ID, then the retrieval view is the concatenation of subsystems SUBSYS2 and SUBSYS3.
- If SUBSYS3 is supplied as input, the retrieval view is SUBSYS3 only.

3.1.3 System and Subsystem Definition

This section provides an outline of the basic approach to be followed when defining systems and their subsystems. We discuss the restrictions that must be observed when defining systems as well as the sequence of activities that must be performed when processing systems and subsystems within the MA2000 environment.

3.1.3.1 System Constraints

Because a system represents the fundamental organizational and operational entity within MA2000, the total number of resources that ultimately constitute a single system has an impact on the performance and is therefore a matter for careful consideration when determining system boundaries. Table 1 on page 21 shows the recommended maximum allowable values to be observed when defining a system.

Table 1. Limits on the Size of a System

Program to be Analyzed	Total Number of Programs	Average Number of Programs per System	Maximum Number of Programs per System	Average Number of Lines per Member	Maximum Number of Lines per Member
COBOL,PL/I,CA-Easytrieve (after macro expansion)	40,000	500	2,000	500	3,000
JCL (after macro expansion)	60,000	70	1,200	100	4,200

3.1.3.2 Defining Systems and Subsystems within the MA2000 Environment

This section provides an overview of the steps involved in defining a system and its subsystems to MA2000 and discusses the information required for both batch and on-line processing of a system.

Overview: The sequence of steps to be performed when defining and processing systems and subsystems in MA2000 is as follows:

1. Gather your software resources into as many systems as required while observing the limits imposed by MA2000 shown in Table 1.
2. Group the resources of any given system into subsystems where appropriate. A system must have at least one subsystem.
3. Assign each subsystem a subsystem ID.
4. Assign a group of related subsystems a system ID.
5. Define the order in which the subsystems are to be concatenated.
6. Execute the analysis process for each subsystem ID to create an analysis buffer for every subsystem.
7. Run the storing process to create the subsystem intermediate data.
8. Create the system intermediate data using the subsystem intermediate data as input.
9. Run the individual or multiple retrieval process against each system using the desired retrieval view.
10. Delete the intermediate data for each system as necessary, to conserve DASD space.

Input for Batch Processing: Sample JCL has been provided for the batch jobs necessary to establish MA2000 analysis and retrieval environments for each system. Select the appropriate JCL according to the number of subsystems defined for each system as illustrated in Table 2 on page 22. Specify the names of the system and subsystems to be analyzed by supplying an appropriate value for the required input parameters #SYSID and #SUBIDn and execute the JCL against the system to which the subsystems belong.

Table 2. MA2000 Execution JCL Libraries	
Number of Subsystems per System	**Execution JCL Library**
1	IMX.V1R3M0.SIMXLIB1
2	IMX.V1R3M0.SIMXLIB2
3	IMX.V1R3M0.SIMXLIB3

Use the following input parameters to specify the priority of the subsystems that make up each system. For a discussion of the hierarchical relationship among the subsystems of a system, see 3.1.2, “The System and Subsystem Hierarchy” on page 19. The parameters are these:

1. IMX.V1R3M0.SIMXLIB1
   - Specify the subsystem ID with the subparameter SUB1.
2. IMX.V1R3M0.SIMXLIB2
   - Specify the upper-level subsystem ID with the subparameter SUB1.
   - Specify the lower-level subsystem ID with the subparameter SUB2.
3. IMX.V1R3M0.SIMXLIB3
   - Specify the highest-level subsystem ID with the subparameter SUB1.
   - Specify the second-highest subsystem ID with the subparameter SUB2.
   - Specify the lowest with the subparameter SUB3.

Input for On-line Processing: Before any of the individual or multiple retrieval processes can be activated in the on-line environment, a member with a name identical to the value of the system ID input parameter used in batch processing must exist in the system-identification file IMX.V1R3M0.SIMXID. This member contains parameters effective for each individual system. For more detailed information regarding the values of the system parameters supplied here, see 3.2, “MA2000 System and System ID Parameter Definitions.”

Note

The existence of an appropriate member in the system ID file is a prerequisite for any on-line processing.

3.2 MA2000 System and System ID Parameter Definitions

This section examines the various types of parameters that must be specified to construct the MA2000 system analysis and retrieval environments as well as those parameters required at the level of an individual system ID. A clear distinction must be drawn between system and system ID parameters:

System parameters

System parameters are parameters whose values are valid for one or more systems being processed within the batch and on-line environments of MA2000. System parameters may be understood as a convenient method of defining parameters simultaneously for multiple systems, thereby alleviating the necessity for the user to specify the same parameters repeatedly at the individual system level. Although the effectiveness of system parameters may extend beyond the boundaries of

individual systems, MA2000 also offers the flexibility of allowing a user to define a set of system parameters for a single system being analyzed. The relationship between MA2000 system parameters (as defined by a SYSPRM file) and a system (as defined by a SYSID file) can be characterized as follows:

- One system parameter file can be defined for every system in the MA2000 environment.
- One system parameter file can be defined for each system in the MA2000 environment.
- One system parameter file can be defined for a group of individual systems in the MA2000 environment.

System parameters are stored in members of a partitioned data set and apply to the following MA2000 components:

- PL/I analysis
- COBOL analysis
- CA-Easytrieve Plus analysis
- JCL analysis
- Retrieval environment libraries
- Retrieval environment output data sets and print functions
- Individual result retrieval search function
- Program correlation chart formats
- All types of impact analysis

A special case of system parameters are the *environment parameters*. Environment parameters are defined in one member of the environment parameter data set IMX.V1R3M0.SIMXENV. The environment parameters should apply to all the MA2000 systems in the entire environment. Through these parameters, you specify the high-level qualifiers of the MA2000 product data sets as well as the data set names of your run-time libraries and the ISPF system libraries. If you want to share the MA2000 analysis and retrieval data among various users, you can specify as environment parameters a common first-level qualifier to be used for the analysis data and one to be used for the retrieval data.

System ID Parameters

These are parameters that are effective for one system ID. These values are supplied in the appropriately named member of the system ID data set (IMX.V1R3M0.SIMXID).

Because our sample scenario is composed exclusively of COBOL source programs and JCL members, this discussion of system and system ID parameters is limited to the parameters required to establish the analysis and retrieval environments. The parameters you need to modify depend to some degree upon the objects of your analysis. For a more detailed examination of the system and system ID parameters for other analysis and retrieval environments, please see Chapter 6, ″System Parameters″ in the manual *Maintenance 2000 Guide and Reference Version 1, Release 3*.

Note

Please note that any modified parameter values in the following discussion are installation specific and are intended only as guidelines as to what values may require change in your installation. The values you supply for any of these parameters depend entirely upon your environment. Please consult with your system administrator if necessary.

3.2.1 Standard and Nonstandard System Parameters

There is no doubt that the system parameter file as it exists at the completion of the installation process requires modification in order to conform to the environment in which MA2000 is running. It may, however, be necessary to further alter parameters for a specific system or a group of systems. For example,

- Every result retrieval output data set for a system or systems may need to share a common first level qualifier for identification purposes.
- For a specific system or group of systems, it may be desirable for MA2000 to recognize data items containing both two- and four-digit dates when performing the date-related search function.

Such individual requirements can be accommodated by the creation of a nonstandard system parameter file. A nonstandard system parameter file modifies the corresponding MA2000 system parameters for one or more systems.

3.2.1.1 Creating a Nonstandard System Parameter File

A nonstandard system parameter file should be created by allocating a new partitioned data set with attributes identical to those of the original system parameter data set and containing only those system parameter members whose contents have been manually modified. Thus, a complete standardized system parameter file will always exist for every MA2000 installation rather than a multitude of divergent copies.

There is one exception: you must always copy member IMXR1 to the nonstandard system parameter data set even if you don't change anything in it.

3.2.1.2 Specifying a Nonstandard System Parameter File

When a nonstandard system parameter file is to be used in a batch job or for the duration of an on-line TSO session during which MA2000 retrieval functions are invoked, the name of the nonstandard system parameter file must be supplied in the following items:

- If a different system parameter file is to be used during the batch analysis process, then the SYSPRM= parameter in the corresponding JCL member must be appropriately specified. One or all of the following members of the appropriate SIMXLIBn library will require modification depending upon the type of source code being analyzed:
  - IMX3ANC
  - IMX3ANP
  - IMX3ANJ

Note

If you wish to use your standard system parameter file, delete the DD statement where &SYSPRM is specified in the jobs listed, or allocate a dummy data set and specify it for the &SYSPRM parameter.

- Only those systems that share the same system parameter file can be processed in the same on-line session. If result retrieval is to be performed against a system that needs a nonstandard system parameter file, then specify the name of the data set where the nonstandard system parameter members reside as the #SYSPRM parameter in the corresponding SYSID member of the system ID parameters data set (see 3.2.3, “System ID Parameters” on page 33).

  The system parameter data sets do not have to be allocated to the TSO session by the user. The MA2000 on-line procedures make these allocations. The standard system parameter data set name is given by the #IMXPFIX parameter in the SYS@ member of the environment parameter data set with a last level qualifier of SIMXPRM. The nonstandard system parameter data set name is specified in the system ID parameter #SYSPRM. The system ID parameter data set name is given by the #IMXPFIX parameter in the SYS@ member of the environment parameter data set with a last-level qualifier of SIMXID.

  Therefore, the only parameter data set that must be allocated to the TSO session by the TSO user is the environment parameter data set. The TSO user must first terminate the current TSO session and reestablish a session using a logon CLIST containing the appropriate data-set allocations and concatenation sequence. Figure 4 shows an example of the logon CLIST used in our sample environment.

```
CONTROL NOMSG NOFLUSH NOLIST
SE '***** STARTING DEFAULT CLIST FOR' USER(*)
ALLOC F(LOADLIB) DA('IMX.V1R3M0.SELBMOD0' -
                    'CEEV1R50.SCEERUN'    -
                    'IMX.V1R3M0.SIMXLOAD') SHR REUSE
STEPLIB SET(LOADLIB)
CONCATD F(ELBWSLIB) DA('IMX.V1R3M0.SELBWS') SHR
CONCATD F(ELBBLLIB) DA('IMX.V1R3M0.SIMXBLK') SHR
CONCATD F(SYSEXEC) DA('IMX.V1R3M0.SIMXEXEC') SHR
CONCATD F(SYSEXEC) DA('IMX.V1R3M0.SELBEXEC') SHR
/* The following data set is the Environment Parameter file */
CONCATD F(SYSEXEC) DA('IMX.V1R3M0.SIMXENV') SHR
CONCATD F(ISPPLIB) DA('IMX.V1R3M0.SIMXPENU') SHR
CONCATD F(ISPMLIB) DA('IMX.V1R3M0.SIMXMENU') SHR
CONCATD F(ISPSLIB) DA('IMX.V1R3M0.SIMXSLIB') SHR
```

Figure 4. Sample TSO logon CLIST for Allocation and Concatenation of MA2000 System Files

- The #SYSPRM parameter in the appropriate SYSID member of the SYSID data set must be set to the name of the nonstandard system parameter file. This parameter is also used to concatenate the nonstandard system parameter data set name before that of the standard parameter data set name in all the batch JCL generated by the on-line result retrieval options. Consequently, if a nonstandard SYSPRM file is to be used during the result retrieval impact or Year 2000 analysis process, the JCL generated by MA2000 does not require any manual modification for the changes made in the nonstandard system parameter data set to take effect.

Figure 5 on page 26 illustrates the order of precedence among nonstandard and standard system parameter files and their relationship to both the MA2000 environment as well as to individual systems and the subsystems of which they are composed. The boxes in the diagram represent hierarchical relationships: a larger box always takes precedence over a smaller one.

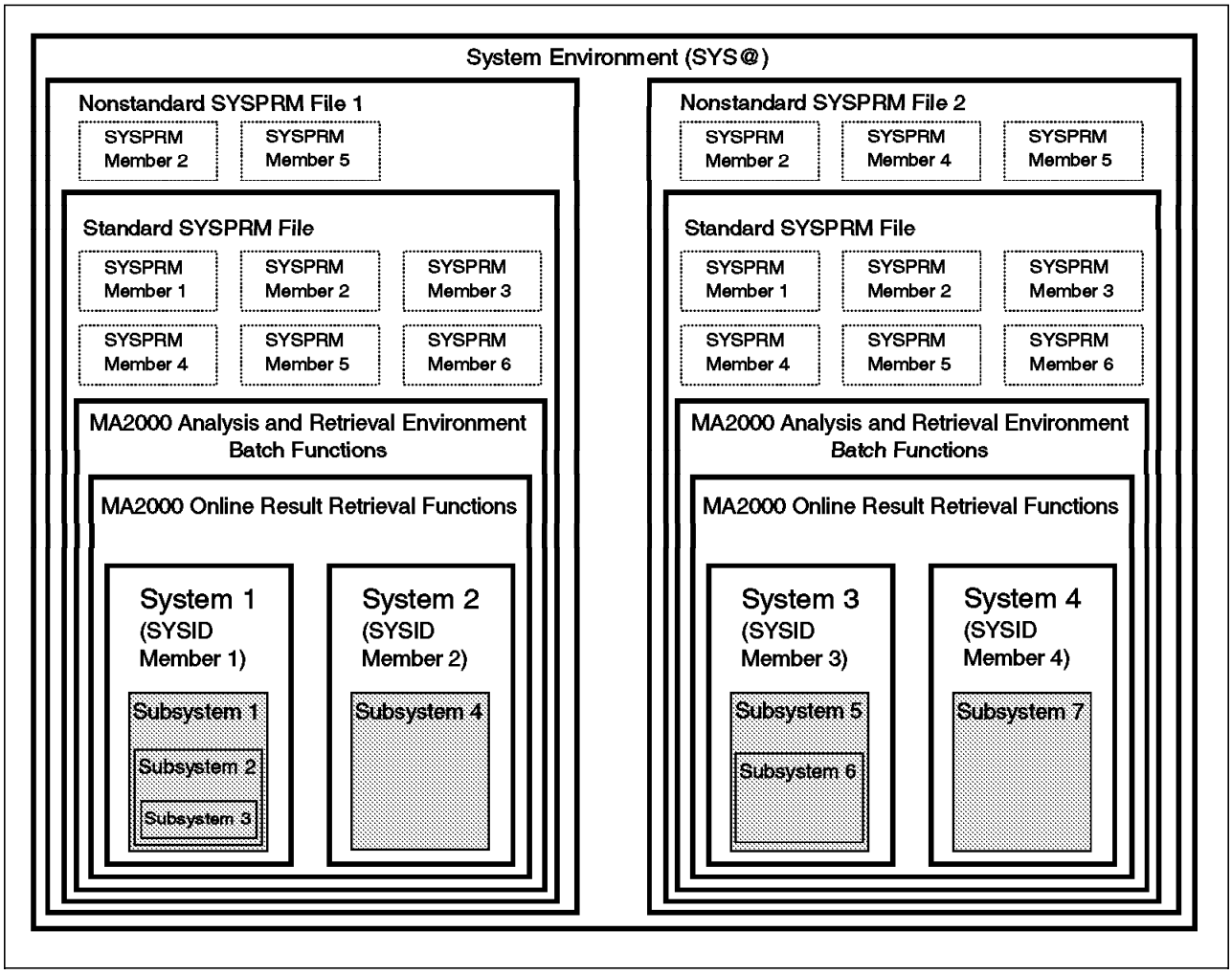


Figure 5. System Parameter Files and the MA2000 Environment

3.2.2 System Parameter Types

Only one member is needed in the environment parameter data set.

The naming conventions for the parameter members found in the system parameter data set (SYSPRM) indicate the type and scope of the parameters they contain:

- **COMMON** — This customizable member contains cross-functional system parameters.
- **IMXxx** — "xx" varies according to function. All these customizable members contain parameters related to analysis or retrieval functions.
- **ZIMXxx** — Again, "xx" varies for each analysis or retrieval function. These members are usually not customizable.

3.2.2.1 Environment Parameter

To create the valid environment parameters for your installation:

1. Create a member SYS@ in the environment parameter data set IMX.V1R3M0.SIMXENV.
2. Copy member IMXSYS00 of the environment parameter data set to member SYS@.
3. Change all the parameter values that start with @ to a blank (' ').
4. Specify the necessary parameters for your environment. Table 3 explains the environment variables.

Table 3. Environment Parameters

Parameter	Description	Our Value
#TBLVOL	This parameter specifies the volume serial of the DASD where MA2000 will allocate ISPF tables for your on-line environment. This parameter is mandatory.	SYS153
#DISK	This parameter specifies the unit name of your DASD. This parameter is mandatory.	SYSDA
#IMXPFIX	This parameter specifies the high-level qualifiers of the MA2000 product data sets in your installation. This parameter is mandatory.	IMX.V1R3M0
#ELBPFIX	This parameter specifies the high-level qualifiers of the Prolog product data sets in your installation if you don't use Prolog that comes with MA2000. Otherwise, the Prolog data sets have the same high-level qualifiers as MA2000. This parameter is mandatory.	IMX.V1R3M0
#SRTTAG	This parameter specifies the program name of the DFSORT program in your installation. This parameter is mandatory.	ICEMAN
#SRTLIB	This parameter specifies the data set name of the SORTLIB library in your installation. You don't have to specify it, if it is in the link list. This parameter can be blank.	''
#RUNLIB1 #RUNLIB2 #RUNLIB3	These parameters specify the SEDCLINK, SIBMLINK, and PLILINK run-time libraries in your installation. You don't have to specify them if they are in the link list or if your system is running under Language Environment. These parameters can be blank.	''
#ISPLIB #ISPLPA #ISRLIB #ISRLPA	These parameters specify the ISPF and PDF load libraries in your installation. You don't have to specify them if they are in the link list. These parameters can be blank.	''
#ANAPFIX	This parameter specifies the first-level qualifier of the analysis data sets to be used in your MA2000 analysis environment. If you leave it blank, the ID of the MA2000 user is substituted. This parameter can be blank.	''
#RETPFIX	This parameter specifies the first level qualifier of the result data sets to be used in your MA2000 retrieval environment. If you leave it blank the ID of the MA2000 user is substituted. This parameter can be blank.	''

Note

If no value is specified for the retrieval result data set prefix specified in the #RETPFIX, the user's TSO user ID is supplied as a default. If this parameter is supplied, the user's retrieval result data sets have a first-level qualifier consisting of the value specified for the #RETPFIX parameter but the second-level qualifier is the same as the user's TSO user ID. This construct prohibits retrieval result data sets from being shared among on-line users since the individual user ID always plays a role in the generation of data set names.

The #RETPFIX parameter can be specified in order to group all retrieval result data sets produced by various users together under one common first level to make them more easily identifiable.

3.2.2.2 Common Parameters

We did not need to modify any of the default values initially supplied for these parameters when we analyzed our sample application with MA2000. We recommend, however, that you verify the parameters involving the date formats of the DB2 special register and the CICS system initialization table against those used in your own environment.

3.2.2.3 Parameter Members for the Analysis Environment

This section describes the various parameter members used to construct the analysis environment and relates them, where appropriate, to our sample scenario. For a list of all the parameters contained in these members and their initial values, see Chapter 9, "Setting Up the Analysis Environment" in the manual *Maintenance 2000 Guide and Reference Version 1, Release 3*. The parameter members are these:

- **IMXAA** (JCL analysis parameters) — We did not find it necessary to change any of the initially supplied values.
- **IMXAB** (PL/I analysis parameters) — PL/I analysis is outside the scope of the sample scenario.
- **IMXEB** (CA-Easytrieve analysis parameters) — CA-Easytrieve analysis is outside the scope of the sample scenario.
- **IMXDA** (COBOL analysis parameters) — Table 4 explains the parameter that required modification for the sample scenario. All other parameters retain their initial values.

Table 4. COBOL Analysis Parameters Requiring Modification

Parameter	Description	Initial Value	Modified Value
literal_delimiter	This parameter specifies the delimiter character for literals in COBOL source programs.	'APOST'	'QUOTE'

A nonnumeric literal in a COBOL program is a character string enclosed in quotation (") marks. As an IBM COBOL extension, you can use apostrophes (') as the literal delimiters instead of quotation marks. Please note that the default value of the literal delimiter COBOL analysis parameter is set to 'APOST'. Quite inexplicable error messages result from the COBOL analysis

process if the value of this parameter does not reflect your environment's COBOL programming convention, as Figure 6 on page 29 illustrates.

```
TARMU3B          IMXAC003I  COBOL analysis member: TARMU3B
TARMU3B          IMXAP052E  The variable '23' is not defined.  Line: 20
TARMU3B          IMXAC011I  Analysis result: ADD,  Number of lines: 541
```

Figure 6. Error Messages Resulting from an Inconsistent Literal Delimiter Parameter

For more information on nonnumeric literals, see Chapter 1, "COBOL Language Structure" in the manual *COBOL for OS/390 & VM Language Reference*.

Figure 7 shows the contents of the COBOL analysis parameter member used in the analysis of our sample scenario source code.

```
%************************************************************************%
%* Maintenance 2000 Version 1 Release 3 Modification Level 0.          *%
%* 5655-A33 (C) Copyright IBM Corp. 1996, 1998.                        *%
%* All Rights Reserved.                                                *%
%* Licensed Materials - Property of IBM.                               *%
%* See Copyright Instructions, G120-2083.                              *%
%************************************************************************%
%************************************************************************%
%* Before working your Maintenance 2000 environment                    *%
%*      you will have to make sure of the following:                   *%
%* -    If you plan to change                                          *%
%*                     any of the preset values,                       *%
%*                  or add                                             *%
%*                     any of the values on the omitted parameters:    *%
%*      Change the preset values                                       *%
%*             to your desired ones,                                   *%
%*      or add the omitted parameters                                  *%
%*             specifying your desired values,                         *%
%*                by consulting                                        *%
%*                      the Maintenance 2000                           *%
%*                          Guide and Reference                        *%
%*                      for the corresponding parameters.              *%
%************************************************************************%
   dup_update(on) .
   literal_delimiter("QUOTE") .
   var_delete(yes) .
```

Figure 7. Sample Scenario COBOL Analysis Parameters

3.2.2.4 Parameter Members for the Retrieval Environment

This section describes the various parameter members used to construct the retrieval environment and relates them, where appropriate, to our sample scenario. For a list of all the parameters in these members and their initial values, see Chapter 11, "Setting Up the Retrieval Environment" in the manual *Maintenance 2000 Guide and Reference Version 1, Release 3*. The parameter members are these:

- **IMXR1** (Output parameters) — Table 5 on page 30 explains the retrieval environment output parameters that required modification for the sample scenario. All other parameters retained their initial values.

Table 5. Retrieval Environment Output Parameters Requiring Modification

Parameter	Description	Initial Value	Modified Value
#OUTPAR 1-5	Specifies a maximum of five lines of the print output definition statement.	'//'#OUTPUT' OUTPUT COPIES=1'	See Figure 8 on page 31.
#OUTPUT	Specifies the reference name of the print output definition statement.	None	'OUT1'
#OUTUSE	Specifies whether to use the output statement or not.	'N'	'Y'
#PRTJCL 1-9,A-C	Specifies the print JCL statements.	None	See Figure 8 on page 31.
#OUTDCB	Specifies the DCB of the output DD statement used by the print statement.	None	See Figure 8 on page 31.

Figure 8 on page 31 shows the contents of the retrieval environment output parameter member used in the analysis of our sample scenario source code.

```
/*********************************************************************/
/*                                                                   */
/*  Maintenance 2000 Version 1 Release 3 Modification Level 0.       */
/*  5655-A33 (C) Copyright IBM Corp. 1996, 1998.                     */
/*  All Rights Reserved.                                             */
/*  Licensed Materials - Property of IBM.                            */
/*  See Copyright Instructions, G120-2083.                           */
/*                                                                   */
/*********************************************************************/

/*********************************************************************/
/*                                                                   */
/*  Before working your Maintenance 2000 environment                 */
/*       you will have to make sure of the following:                */
/*                                                                   */
/*  -    If you plan to change                                       */
/*                      any of the preset values:                    */
/*       Change the preset values                                    */
/*              to your desired ones                                 */
/*                 by consulting                                     */
/*                       the Maintenance 2000                        */
/*                           Guide and Reference                     */
/*                       for the corresponding parameters.           */
/*                                                                   */
/*********************************************************************/

    #VERSION  = 130
    #PRTOPT1  = 1
    #PRTLIN   = 50
    #OUTUSE   = 'Y'
    #OUTPUT   = 'OUT1'
    #OUTPAR1  = '//'#OUTPUT' OUTPUT DEFAULT=YES,            '
    #OUTPAR2  = '//            PAGEDEF=V06481,PRMODE=PAGE,  '
    #OUTPAR3  = '//            CHARS=GT20,DEST=ST35G2A,     '
    #OUTPAR4  = '//            COPIES=1                     '
    #OUTPAR5  = '//*                                        '
    #INDD     = 'SYSUT1'
    #OUTDD    = 'SYSUT2'
    #OUTDCB   = 'DCB=(RECFM=FBA,LRECL=133,BLKSIZE=3192)'
    #PRTJCL1  = '//        EXEC PGM=IEBGENER                '
    #PRTJCL2  = '//SYSPRINT  DD DUMMY                       '
    #PRTJCL3  = '//SYSIN     DD DUMMY                       '
    #PRTJCL4  = '//*                                        '
    #PRTJCL5  = '//*                                        '
    #PRTJCL6  = '//*                                        '
    #PRTJCL7  = '//*                                        '
    #PRTJCL8  = '//*                                        '
    #PRTJCL9  = '//*                                        '
    #PRTJCLA  = '//*                                        '
    #PRTJCLB  = '//*                                        '
    #PRTJCLC  = '//*                                        '
```

Figure 8. Sample Scenario Retrieval Environment Output Parameters

- **IMXBE** (Program correlation chart parameters) — The initial value of this parameter, which specifies that members occuring more than once within the same tree should be indicated in the program correlation chart, did not require modification. You should set this parameter to ″off″ if you wish to indicate those members that occur more than once in recursive calls in the program correlation chart.
- **IMXBG** (Impact analysis parameters) — Although these impact analysis parameters do not normally require any modification, two of them have a direct impact on the Year 2000 impact analysis process and might therefore theoretically require customization depending on the data you are analyzing:
  - sys_start(date,OnOff) — This parameter specifies whether or not MA2000 automatically recognizes the data items in all programs in a domain which are involved in date functions as starting-points for the Year 2000

impact analysis. This flag is initially turned on. If this flag is turned off, only those date-related data items explicitly specified by the user are designated as starting-points for the Year 2000 impact analysis. This suspension of the automated date recognition function might be advantageous when analyzing particularly large amounts of data by helping to limit the initial number of starting-point items.

- sys_start(yyyy,OnOff) — This parameter is effective only when ″sys_start(date,on)″ is specified. This parameter indicates whether or not MA2000 is to automatically treat data items involved in date functions that support four-digit dates as starting-points. Normally, MA2000 automatically designates only those items in which two-digit year data is directly set by means of a date-related function as a starting-point.

Figure 9 shows the contents of the impact analysis parameter member used in the analysis of our sample scenario source code. Items preceded by the '%' sign are considered as comments.

```
%************************************************************************%
%* Maintenance 2000 Version 1 Release 3 Modification Level 0.         *%
%* 5655-A33 (C) Copyright IBM Corp. 1996, 1998.                       *%
%* All Rights Reserved.                                               *%
%* Licensed Materials - Property of IBM.                              *%
%* See Copyright Instructions, G120-2083.                             *%
%************************************************************************%
%************************************************************************%
%* Before working your Maintenance 2000 environment                   *%
%*      you will have to make sure of the following:                  *%
%* -    If you plan to change                                         *%
%*                     any of the preset values,                      *%
%*                  or add                                            *%
%*                     any of the values on the omitted parameters:   *%
%*      Change the preset values                                      *%
%*             to your desired ones,                                  *%
%*      or add the omitted parameters                                 *%
%*             specifying your desired values,                        *%
%*                by consulting                                       *%
%*                      the Maintenance 2000                          *%
%*                          Guide and Reference                       *%
%*                      for the corresponding parameters.             *%
%************************************************************************%
   sys_impact_rule(comparison,on) .
   sys_multi_chk(bndry,on) .
   sys_output_form(batch) .
   sys_start(date,on) .
   sys_start(yyyy,on) .
```

Figure 9. Sample Scenario Impact Analysis Parameters

- **IMXBH** (Search parameters) — Table 6 on page 33 explains the retrieval environment search parameters that required modification for the sample scenario. All other parameters retained their initial values.

Table 6. Retrieval Environment Search Parameters Requiring Modification			
Parameter	**Description**	**Initial Value**	**Modified Value**
sys_start	Specifies whether the date-related function search option should retrieve data items which are the targets of an assignment statement involving program date functions that support four-digit years.	sys_start (yyyy,off)	sys_start (yyyy,on)

Figure 10 shows the contents of the retrieval environment search parameter member used in the analysis of our sample scenario source code.

```
%************************************************************************%
%* Maintenance 2000 Version 1 Release 3 Modification Level 0.         *%
%* 5655-A33 (C) Copyright IBM Corp. 1996, 1998.                       *%
%* All Rights Reserved.                                               *%
%* Licensed Materials - Property of IBM.                              *%
%* See Copyright Instructions, G120-2083.                             *%
%************************************************************************%
%************************************************************************%
%* Before working your Maintenance 2000 environment                   *%
%*      you will have to make sure of the following:                  *%
%* -    If you plan to change                                         *%
%*                     any of the preset values,                      *%
%*                  or add                                            *%
%*                     any of the values on the omitted parameters:   *%
%*      Change the preset values                                      *%
%*             to your desired ones,                                  *%
%*      or add the omitted parameters                                 *%
%*             specifying your desired values,                        *%
%*                by consulting                                       *%
%*                      the Maintenance 2000                          *%
%*                          Guide and Reference                       *%
%*                      for the corresponding parameters.             *%
%************************************************************************%
   sys_start(yyyy,on) .
   syslib(yes) .
```

Figure 10. Sample Scenario Retrieval Environment Search Parameters

3.2.3 System ID Parameters

A member must exist in the partitioned system ID data set (IMX.V1R3M0.SIMXID) for each system to be processed in an MA2000 installation. The member name must be identical to the name of the system being defined. The parameters defined in a member of the system ID data set are valid for only one system. The parameters specified here include the system and subsystem names and the names of cataloged procedure libraries, COBOL copybook libraries, and PL/I and COBOL SQL include libraries belonging to the system. The concatenation of multiple libraries is possible.

One of the most important parameters to be supplied in the system ID member of the system ID file is the name of the nonstandard system parameter data set, if one exists. The value of this parameter is used to construct the concatenation sequence of your nonstandard and standard system parameter libraries in the batch JCL generated for result retrieval functions.

Table 7 on page 34 explains the system ID parameters that required modification for the sample scenario.

Table 7. System ID Parameters Requiring Modification

Parameter	Description	Initial Value	Modified Value
#SYSID	System ID	'SYS1'	'SYSITSO'
#SUBID1	Subsystem ID	'NONE'	'SUBSITSO'
#CATPRO1	Cataloged procedure library name	''	'ITSORS2.SAMPAPPL.JCLPRC'
#CBLCPY1	COBOL copybook library name	''	'ITSORS2.SAMPAPPL.COPYBOOK'
#VOLWRK	Volume where workstation export files should reside	'VVVVVV'	'SYS155'
#VOLSER	Volume where all other newly created files should reside	'VVVVVV'	'SYS155'
#SYSPRM	Nonstandard system parameter library name	''	'ITSORS3.MA2000.SYSPRM'

Note

A value must be supplied for the two volume parameters #VOLSER and #VOLWRK. If these values are missing, the JCL generated by the retrieval result options will fail as a result of JCL errors.

Figure 11 on page 35 shows the contents of system ID member SYSITSO used in the sample scenario.

```
/*********************************************************************/
/*                                                                   */
/*  Maintenance 2000 Version 1 Release 3 Modification Level 0.       */
/*  5655-A33 (C) Copyright IBM Corp. 1996, 1998.                     */
/*  All Rights Reserved.                                             */
/*  Licensed Materials - Property of IBM.                            */
/*  See Copyright Instructions, G120-2083.                           */
/*                                                                   */
/*********************************************************************/

    #SYSID   = 'SYSITSO'
    #SUBID1  = 'SUBSITSO'
    #SUBID2  = 'NONE'
    #SUBID3  = 'NONE'
    #CATPRO1 = 'ITSORS2.SAMPAPPL.JCLPRC'
    #CATPRO2 = ''
    #CATPRO3 = ''
    #CATPRO4 = ''
    #CATPRO5 = ''
    #CBLCPY1 = 'ITSORS2.SAMPAPPL.COPYBOOK'
    #CBLCPY2 = ''
    #CBLCPY3 = ''
    #CBLCPY4 = ''
    #CBLCPY5 = ''
    #CBLCPY6 = ''
    #CBLCPY7 = ''
    #CBLCPY8 = ''
    #CBLCPY9 = ''
    #CBLCPYA = ''
    #SQLINC1 = ''
    #SQLINC2 = ''
    #SQLINC3 = ''
    #SQLCPY1 = ''
    #SQLCPY2 = ''
    #SQLCPY3 = ''
    #VOLWRK  = 'SYS155'
    #VOLSER  = 'SYS155'
    #SYSPRM  = 'ITSORS3.MA2000.SYSPRM'
    IMXSYSA  = '#SYSID #SUBID1 #SUBID2 #SUBID3'
    IMXSYSB  = '#CBLCPY1 #CBLCPY2 #CBLCPY3 #CBLCPY4 #CBLCPY5'
    IMXSYSC  = '#CBLCPY6 #CBLCPY7 #CBLCPY8 #CBLCPY9 #CBLCPYA'
    IMXSYSD  = '#SQLINC1 #SQLINC2 #SQLINC3'
    IMXSYSE  = '#SQLCPY1 #SQLCPY2 #SQLCPY3'
    IMXSYSF  = '#CATPRO1 #CATPRO2 #CATPRO3 #CATPRO4 #CATPRO5'
    IMXSYSG  = '#VOLWRK #VOLSER #SYSPRM'

    address ispexec "VPUT ("IMXSYSA IMXSYSB IMXSYSC,
                           IMXSYSD IMXSYSE IMXSYSF,
                           IMXSYSG") PROFILE"
    return rc
```

Figure 11. Contents of the Sample System ID Member

3.3 Overview of Analysis and Retrieval Environment Creation

This section provides an overview of the tasks involved in the creation of the analysis and retrieval environments for a single system in preparation for result retrieval in the MA2000 on-line environment. We outline the steps to be followed for each language supported by MA2000 as well as the order in which they are to be performed when a system comprises a mixture of the supported languages. We also discuss the steps that must be taken when source statements are modified after the completion of MA2000 analysis. For a detailed examination of the steps we undertook to create the analysis and retrieval environments for our sample scenario COBOL source and JCL statements, see 3.4, “COBOL Processing” on page 41 and 3.5, “JCL Processing” on page 69.

3.3.1 Execution JCL

Table 8 provides a brief description of the JCL members supplied to create the MA2000 analysis and retrieval environments and specifies the order in which they are normally to be executed.

Table 8. MA2000 Execution JCL for Analysis and Retrieval Environment Creation

Member	Description	Target Level	JCL	PL/I	COBOL	EASYTR
IMX0PRE	PL/I precompilation	Subsystem		1		
IMX1ALC	Allocation of intermediate data sets	Subsystem	1	2	1	1
IMX2CDJ	JCL analysis card creation	Subsystem	2			
IMX2CDP	PL/I analysis card creation	Subsystem		3		
IMX2CDC	COBOL analysis card creation	Subsystem			2	
IMX2CDE	CA-Easytrieve Plus analysis card creation	Subsystem				2
IMX3ANJ	JCL analysis	Subsystem	3			
IMX3ANP	PL/I analysis	Subsystem		4		
IMX3ANC	COBOL analysis	Subsystem			3	
IMX3ANE	CA-Easytrieve Plus analysis	Subsystem				3
IMX4SBJ	JCL storing of subsystem intermediate data	Subsystem	4			
IMX4SBP	PL/I storing of subsystem intermediate data	Subsystem		5		
IMX4SBC	COBOL storing of subsystem intermediate data	Subsystem			4	
IMX4SBE	CA-Easytrieve Plus storing of subsystem intermediate data	Subsystem				4
IMX5DCJ	JCL-related dictionary creation	Subsystem and System	5			
IMX5DCP	PL/I-related dictionary creation	Subsystem and System		6		
IMX5DCC	COBOL-related dictionary creation	Subsystem and System			5	
IMX5DCE	CA-Easytrieve Plus-related dictionary creation	Subsystem and System				5
IMX6TBL	Table registration	System		7	6	6
IMX7CKM	Unregistered member check	Subsystem		8	7	7
IMX7CKD	Unspecified DL/I call check	Subsystem		9	8	8
IMX9DEL	Deletion of intermediate data sets	Subsystem	6	10	9	9

When analyzing a system whose subsystems are composed of a mixture of the supported languages, perform the steps outlined here for each language up to and including the point of dictionary creation after which the remainder of the steps can be performed all together.

Note

- All members with the exception of IMX1ALC can be rerun if the job ends in error. If IMX1ALC is to be rerun, job IMX9DEL must first execute.
- When rerunning any of the jobs, pay particular attention to the status of data sets that are created and subsequently deleted during the course of the job. MA2000 jobs normally begin with a preliminary deletion of all data sets allocated during the course of job execution.
  - If you are submitting a job for the first time or resubmitting a job that completed successfully at the time of its last execution, place all the preliminary deletion steps performed by PGM=IEFBR14 in commentary prior to execution to avoid unnecessary JCL errors.
  - If you are resubmitting a job that did not complete successfully at the time of its last execution, check the status of the data sets deleted at the outset of the job to see which ones still exist and which ones may already have been deleted prior to the abnormal termination of the job and place the unnecessary deletion steps in commentary.

3.3.2 Exceptional Cases

If a program must be modified in order to correct an analysis error such as incompatible member and program names, the following steps must be performed after you complete the appropriate action to correct the error:

- For COBOL programs, the steps from analysis card creation (IMX2CDC) through COBOL-related dictionary creation must be repeated.
- For PL/I programs, repeat the steps from precompilation (IMX0PRE) to PL/I-related dictionary creation if the original source code has been modified or repeat the steps from analysis card creation (IMX2CDP) to PL/I-related dictionary creation if the modification was performed on the precompiled source code.
- For CA-Easytrieve Plus programs, repeat the steps from analysis card creation (IMX2CDE) to CA-Easytrieve Plus-related dictionary creation (IMX5DCE).
- Save the program to be added in the input library, which is used during precompilation, analysis card creation, source analysis, and the storing process. During the storing process, the saved program is copied to the #ANAPFIX.System_ID.Subsystem_ID.SOURCE library, which is used for saving source retrieval members against which analysis was successfully completed.

  Note

  Data set #ANAPFIX.System_ID.Subsystem_ID.SOURCE is used to store a copy of successfully analyzed source programs for later use by the retrieval process. You should therefore ensure that your original program source library does not have an identical name.

- If unregistered members are detected by job IMX7CKM, indicating references to programs for which no source code exists, the same procedure just described for COBOL and CA-Easytrieve Plus programs should be followed. For PL/I programs, the steps from precompilaton to dictionary creation should be performed.

3.3.3 Maintenance 2000 Data Sets

This section provides a list of data sets allocated by MA2000 during the creation of the analysis and retrieval environments, the on-line retrieval result and retrieval result management functions, and the retrieval result accumulation process.

3.3.3.1 Analysis and Retrieval Environment Data Sets

Table 9 describes the data sets which are generated temporarily or permanently during the analysis and retrieval environment creation process.

Note

- "#ANAPFIX" refers to the prefix for analysis data sets which is defined in the environment parameter file member SYS@.
- "#SYSID" and "#SUBIDn" refer to the system ID and subsystem ID specified in the appropriate member of the SYSID data set.

Table 9 (Page 1 of 2). MA2000 Analysis and Retrieval Environment Data Sets

Data Set Name	Description
#ANAPFIX.#SYSID.#SUBIDn.INJCL01	JCL analysis card (output of IMX2CDJ)
#ANAPFIX.#SYSID.#SUBIDn.INPLI01	PL/I analysis card (output of IMX2CDP)
#ANAPFIX.#SYSID.#SUBIDn.INCBL01	COBOL analysis card (output of IMX2CDC)
#ANAPFIX.#SYSID.#SUBIDn.INESY01	CA-Easytrieve Plus analysis card (output of IMX2CDE)
#ANAPFIX.#SYSID.#SUBIDn.DICJ	JCL member list
#ANAPFIX.#SYSID.#SUBIDn.DICP	PL/I, COBOL or CA-Easytrieve Plus member list (input for IMX5DCP, IMX5DCC, or IMX5DCE, respectively)
#ANAPFIX.#SYSID.#SUBIDn.PI01	PL/I, COBOL or CA-Easytrieve Plus program information
#ANAPFIX.#SYSID.#SUBIDn.SOURCE	PL/I, COBOL or CA-Easytrieve source (master) file generated during the storing process
#ANAPFIX.#SYSID.#SUBIDn.PX	Program analysis transaction file
#ANAPFIX.#SYSID.#SUBIDn.JI	JCL information
#ANAPFIX.#SYSID.#SUBIDn.JCL	JCL source file
#ANAPFIX.#SYSID.#SUBIDn.JX	JCL analysis transaction file
#ANAPFIX.#SYSID.#SUBIDn.PMSG	PL/I analysis message file
#ANAPFIX.#SYSID.#SUBIDn.CMSG	COBOL analysis message file
#ANAPFIX.#SYSID.#SUBIDn.EMSG	CA-Easytrieve Plus analysis message file
#ANAPFIX.#SYSID.#SUBIDn.JMSG	JCL analysis message file
#ANAPFIX.#SYSID.#SUBIDn.PLIDL	Misregistered PL/I program deletion card
#ANAPFIX.#SYSID.#SUBIDn.CBLDL	Misregistered COBOL program deletion card
#ANAPFIX.#SYSID.#SUBIDn.ESYDL	MIsregistered CA-Easytrieve Plus program deletion card
#ANAPFIX.#SYSID.#SUBIDn.JCLDL	Misregistered JCL deletion card
#ANAPFIX.#SYSID.#SUBIDn.XJ	JCL-related dictionary (cross-reference)
#ANAPFIX.#SYSID.#SUBIDn.XP	PL/I, COBOL or CA-Easytrieve program-related dictionary (cross-reference)

Table 9 (Page 2 of 2). MA2000 Analysis and Retrieval Environment Data Sets	
Data Set Name	**Description**
#ANAPFIX.#SYSID.#SUBIDn.CP	• COBOL copybook dictionary • PL/I INCLUDE dictionary • CA-Easytrieve Plus macro dictionary
#ANAPFIX.#SYSID.#SUBIDn.FI	Input/Output file dictionary
#ANAPFIX.#SYSID.#SUBIDn.DS	Input/Output data set dictionary
#ANAPFIX.#SYSID.#SUBIDn.SG	Input/Output segment dictionary
#ANAPFIX.#SYSID.#SUBIDn.TB	DB/2 table dictionary
#ANAPFIX.#SYSID.#SUBIDn.CF	CICS file dictionary
#ANAPFIX.#SYSID.#SUBIDn.CQ	CICS queue dictionary
#ANAPFIX.#SYSID.#SUBIDn.CT	CICS transaction dictionary
#ANAPFIX.#SYSID.#SUBIDn.CM	CICS map dictionary
#ANAPFIX.#SYSID.#SUBIDn.HASHJI	JCL hash table
#ANAPFIX.#SYSID.#SUBIDn.HASHPI	Program hash table
#ANAPFIX.#SYSID.#SUBIDn.DUMMYPO	Dummy file
#ANAPFIX.#SYSID.#SUBIDn.SOURCE.B	• Post-precompilation PL/I source • CA-Easytrieve Plus macro expanded source
#ANAPFIX.#SYSID.#SUBIDn.PI.B	PL/I, COBOL or CA-Easytrieve Plus information (buffer) (input for IMX4SBP, IMX4SBC or IMX4SBE, respectively)
#ANAPFIX.#SYSID.#SUBIDn.PX.B	PL/I, COBOL or CA-Easytrieve Plus program analysis transaction (buffer) (input for IMX4SBP, IMX4SBC or IMX4SBE, respectively)
#ANAPFIX.#SYSID.#SUBIDn.JI.B	JCL analysis intermediate data buffer (input for IMX4SBJ)
#ANAPFIX.#SYSID.#SUBIDn.JX.B	JCL analysis transaction (buffer) (input for IMX4SBJ)

3.3.3.2 Retrieval Result Data Sets

Table 10 describes the data sets created to hold the results of the individual or multiple result retrieval process.

Note

"#RETPFIX" is assigned a value in member SYS@ of the environment parameter data set (SYSENV). If no value is specified for this parameter, the user's TSO user ID is used to construct a high-level qualifier for the retrieval result data sets. When a value is supplied, it is used together with the user's TSO user ID to construct the first two qualifiers of the retrieval result data sets. The parenthesis which enclose this parameter in Table 10 indicate its possible absence. "#userid" is intended to represent the user's TSO user ID.

Table 10 (Page 1 of 2). MA2000 Retrieval Result Data Sets	
Data Set Name	**Description**
(#RETPFIX.)#userid.DOMAN.#yymmdd.Thhmmss	Domain
(#RETPFIX.)#userid.DETAL2.#yymmdd.Thhmmss	Retrieval result (detailed)
(#RETPFIX.)#userid.SUMRY.#yymmdd.Thhmmss	Retrieval result (summary)
(#RETPFIX.)#userid.CSVWK.#yymmdd.Thhmmss	CSV (Line data count) file
(#RETPFIX.)#userid.PRGIN.#yymmdd.Thhmmss	Execution statement

Table 10 (Page 2 of 2). MA2000 Retrieval Result Data Sets	
Data Set Name	**Description**
(#RETPFIX.)#userid.RESLT.#yymmdd.Thhmmss	Execution status
(#RETPFIX.)#userid.RESLT2.#yymmdd.Thhmmss	Retrieval result (for accumulating)
(#RETPFIX.)#userid.JFILE.#yymmdd.Thhmmss	Deferred JCL awaiting processing

3.3.3.3 Accumulative Data

Table 11 describes the data sets which are generated to contain the accumulated system results of Year 2000 impact analysis.

Note

- "#ANAPFIX" refers to the prefix for analysis data sets which is defined in the environment parameter file member SYS@.
- "#SYSID" refers to the value of the system ID parameter specified in the appropriately named member of the system ID data set (SYSID).

Table 11. MA2000 Year 2000 Accumulative Result Data Sets	
Data Set Name	**Description**
#ANAPFIX.SYS2000.#SYSID.CNTL	Year 2000 control file
#ANAPFIX.SYS2000.#SYSID.MSGNEW	Year 2000 result message file
#ANAPFIX.SYS2000.#SYSID.PGMNEW	New Year 2000 accumulated program results
#ANAPFIX.SYS2000.#SYSID.PGMOLD	Previous Year 2000 accumulated program results
#ANAPFIX.SYS2000.#SYSID.JCLNEW	New Year 2000 accumulated JCL results
#ANAPFIX.SYS2000.#SYSID.JCLOLD	Previous Year 2000 accumulated JCL results
#ANAPFIX.SYS2000.#SYSID.CPYNEW	New Year 2000 accumulated copybook results
#ANAPFIX.SYS2000.#SYSID.CPYOLD	Previous Year 2000 accumulated copybook results
#ANAPFIX.SYS2000.#SYSID.INCNEW	New Year 2000 accumulated include results
#ANAPFIX.SYS2000.#SYSID.INCOLD	Previous Year 2000 accumulated include results
#ANAPFIX.SYS2000.#SYSID.SQCNEW	New Year 2000 accumulated COBOL SQL results
#ANAPFIX.SYS2000.#SYSID.SQCOLD	Previous Year 2000 accumulated COBOL SQL results
#ANAPFIX.SYS2000.#SYSID.SQINEW	New Year 2000 accumulated PL/I SQL results
#ANAPFIX.SYS2000.#SYSID.SQIOLD	Previous Year 2000 accumulated PL/I SQL results
#ANAPFIX.SYS2000.#SYSID.ESYNEW	New Year 2000 accumulated CA-Easytrieve macro results
#ANAPFIX.SYS2000.#SYSID.ESYOLD	Previous Year 2000 accumulated CA-Easytrieve macro results

3.3.3.4 System Data Sets

Table 12 on page 41 describes the system data set used during the table registration process of retrieval environment creation.

Note

1. "#ANAPFIX" refers to the prefix for analysis data sets which is defined in the environment parameter file member SYS@.
2. "#SYSID" refers to the value of the system ID parameter specified in the appropriately named member of the system ID data set (SYSID).

Table 12. MA2000 System Data Sets

Data Set Name	Description
#ANAPFIX.#SYSID.TBL	Program registration table to register programs in unsupported languages called externally by programs being analyzed.

3.4 COBOL Processing

This section reviews in detail the activities we performed in order to establish the MA2000 analysis and retrieval environments for the COBOL source code from our sample application. Each COBOL-related batch job outlined in 3.3.1, "Execution JCL" on page 36 is examined more closely to indicate what input parameters could require adaptation in your environment and show where the supplied JCL must be modified prior to execution. In addition, a sample of the execution JCL used process our sample application code is provided to illustrate of the customization process.

As our sample application was composed entirely of COBOL source code, our discussion here is limited to COBOL-related tasks. For an examination of the activities associated with other supported languages, please consult Chapter 3, *"*Using MA2000 to Analyze Program and JCL*"* in the manual *Maintenance 2000 Guide and Reference Version 1, Release 3*.

Note

Before submitting a job, you may have to eliminate the initial data set deletion steps from the JCL, See 3.3.1, "Execution JCL" on page 36 for a discussion of how to eliminate potential JCL errors.

3.4.1 Creating the COBOL Analysis Environment

COBOL analysis is performed for each subsystem within the system being processed. The activities that make up the COBOL analysis function are prerequisites for the subsequent creation of the COBOL-related program dictionaries in the retrieval environment. The establishment of the dictionaries is, in turn, a prerequisite for the invocation of the on-line program-related retrieval result functions.

The JCL used for the creation of the analysis environment for our sample application was taken from the SIMXLIB1 data set, as our sample consisted of only one subsystem. If the system you are processing comprises more than one subsystem, use the members located in SIMXLIB2 or SIMXLIB3, as appropriate. We performed the following three jobs to establish the COBOL analysis environment for our system:

1. IMX1ALC

2. IMX2CDC
3. IMX3ANC

3.4.1.1 IMX1ALC: Intermediate Data Set Allocation

Job IMX1ALC allocates and catalogs the intermediate data sets required by MA2000. For a list of MA2000 data sets, see 3.3.3.1, “Analysis and Retrieval Environment Data Sets” on page 38. Job IMX1ALC must be executed only once per system. If you have already submitted this job in conjunction with other analysis activities, it is not necessary for you to resubmit it at this point.

Input Parameters: In addition to supplying a job card in conformity with our installation standards, we found it necessary to change the parameters listed in Table 13 before executing job IMX1ALC.

Table 13. IMX1ALC Input Parameters Requiring Modification for the Sample Application

Parameter	Description	Initial Value	Modified Value
UNIT	Unit name of DASD	#DISK	SYSDA
VOLSER	Volume to which data set allocated	#VOLSER	None
ANAPFIX	Primary qualifier of data set name	#ANAHLQ	ITSORS3
SYSID	System ID	#SYSID	SYSITSO
SUBID	Subsystem ID	#SUBID1	SUBSITSO
UA	Unit of allocation for PL/I source and other data sets	CYL	TRK
PI*,SR*,JI*,JC*,PX*, JX*,XP*,XJ*,DS*,FI*, SG*,CP*,TB*,CM*, CF*,CT*,CQ*	Primary allocation quantities for respective data sets	> 1	= 1

Adaptation: We found it necessary to adapt the JCL supplied for job IMX1ALC by eliminating all the initial deletions performed on the data sets to be allocated. Before the initial execution of IMX1ALC, we removed the following steps to avoid the JCL errors that otherwise would have resulted:

- //PI@DEL EXEC PGM=IEFBR14
- //SR@DEL EXEC PGM=IEFBR14
- //JI@DEL EXEC PGM=IEFBR14
- //JC@DEL EXEC PGM=IEFBR14
- //PX@DEL EXEC PGM=IEFBR14
- //JX@DEL EXEC PGM=IEFBR14
- //PD@DEL EXEC PGM=IEFBR14
- //CD@DEL EXEC PGM=IEFBR14
- //JD@DEL EXEC PGM=IEFBR14
- //ED@DEL EXEC PGM=IEFBR14
- //XP@DEL EXEC PGM=IEFBR14
- //XJ@DEL EXEC PGM=IEFBR14
- //DS@DEL EXEC PGM=IEFBR14

- //FI@DEL EXEC PGM=IEFBR14
- //SG@DEL EXEC PGM=IEFBR14
- //CP@DEL EXEC PGM=IEFBR14
- //TB@DEL EXEC PGM=IEFBR14
- //CM@DEL EXEC PGM=IEFBR14
- //CF@DEL EXEC PGM=IEFBR14
- //CT@DEL EXEC PGM=IEFBR14
- //CQ@DEL EXEC PGM=IEFBR14
- //DU@DEL EXEC PGM=IEFBR14
- //HP@DEL EXEC PGM=IEFBR14
- //HJ@DEL EXEC PGM=IEFBR14

Execution JCL: The JCL shown in Figure 12 is an abbreviated sample of the JCL we executed in order to allocate the MA2000 intermediate data sets. It illustrates the necessary modifications to the input parameters as well as the elimination of initial data set deletions required prior to execution.

```
//ITSORS3# JOB ,
//             MSGLEVEL=(1,1),MSGCLASS=H,TIME=(,59),
//             NOTIFY=&SYSUID,USER=&SYSUID
//*
//*******************************************************************
//*                                                                 *
//*       Maintenance 2000 Version 1 Release 3 Modification Level 0. *
//*       5655-A33 (C) Copyright IBM Corp. 1996, 1998.              *
//*       All Rights Reserved.                                      *
//*       Licensed Materials - Property of IBM.                     *
//*       See Copyright Instructions, G120-2083.                    *
//*                                                                 *
//*******************************************************************
//*
//*-----------------------------------------------------------------*
//* DELETE/ALLOCATE/UNCATALOG/CATALOG MA/2000 INTERMEDIATE DATA SETS *
//*-----------------------------------------------------------------*
//*
//IMXALC  PROC UNIT=SYSDA,                    <= (IF NECESSARY) CHANGE
//*            VOLSER=#VOLSER,                <================ CHANGE
//             ANAPFIX=ITSORS3,               <= (IF NECESSARY) CHANGE
//             SYSID=SYSITSO,
//             SUBID=SUBSITSO,
//*
//     UA=TRK, PI/SOURCE.....................................
//*            JI/JCL.....................................
//*            PX/JX/XP/XJ.....................................
//*            DS/FI/SG.....................................
//*            CP.....................................
//*            TB/CM/CF/CT/CQ.....................................
//     UB=TRK, PLIDL/CBLDL/JCLDL.....................................
//*            DUMMYPO.....................................
//*            HASHPI/HASHJI.....................................
//*
```

Figure 12 (Part 1 of 5). Sample JCL for Intermediate Data Set Allocation

```
//* <QUANTITY> PI           PI1=PRIMARY  PI2=SECONDARY  PIM=DIRECTORY
//             PI1=0029,
//             PI2=0001,
//             PIM=0001,
//*
//* <QUANTITY> SOURCE       SR1=PRIMARY  SR2=SECONDARY  SRM=DIRECTORY
//             SR1=0015,
//             SR2=0001,
//             SRM=0001,
//*
//* <QUANTITY> JI           JI1=PRIMARY  JI2=SECONDARY  JIM=DIRECTORY
//             JI1=0001,
//             JI2=0001,
//             JIM=0001,
//*
//* <QUANTITY> JCL          JC1=PRIMARY  JC2=SECONDARY  JCM=DIRECTORY
//             JC1=0001,
//             JC2=0001,
//             JCM=0001,
//*
//* <QUANTITY> PX           PX1=PRIMARY  PX2=SECONDARY  <- SEQUENTIAL
//             PX1=0001,
//             PX2=0001,
//*
//* <QUANTITY> JX           JX1=PRIMARY  JX2=SECONDARY  <- SEQUENTIAL
//             JX1=0001,
//             JX2=0001,
//*
//* <QUANTITY> PLIDL        PD1=PRIMARY  PD2=SECONDARY  <- SEQUENTIAL
//             PD1=0001,
//             PD2=0001,
//*
//* <QUANTITY> CBLDL        CD1=PRIMARY  CD2=SECONDARY  <- SEQUENTIAL
//             CD1=0001,
//             CD2=0001,
//*
//* <QUANTITY> JCLDL        JD1=PRIMARY  JD2=SECONDARY  <- SEQUENTIAL
//             JD1=0001,
//             JD2=0001,
//*
//* <QUANTITY> XP           XP1=PRIMARY  XP2=SECONDARY  XPM=DIRECTORY
//             XP1=0001,
//             XP2=0001,
//             XPM=0001,
//*
//* <QUANTITY> XJ           XJ1=PRIMARY  XJ2=SECONDARY  XJM=DIRECTORY
//             XJ1=0001,
//             XJ2=0001,
//             XJM=0001,
//*
//* <QUANTITY> DS           DS1=PRIMARY  DS2=SECONDARY  DSM=DIRECTORY
//             DS1=0001,
//             DS2=0001,
//             DSM=0001,
//*
//* <QUANTITY> FI           FI1=PRIMARY  FI2=SECONDARY  FIM=DIRECTORY
//             FI1=0001,
//             FI2=0001,
//             FIM=0001,
//*
//* <QUANTITY> SG           SG1=PRIMARY  SG2=SECONDARY  SGM=DIRECTORY
//             SG1=0001,
//             SG2=0001,
//             SGM=0001,
//*
```

Figure 12 (Part 2 of 5). Sample JCL for Intermediate Data Set Allocation

```
//* <QUANTITY> CP           CP1=PRIMARY  CP2=SECONDARY  CPM=DIRECTORY
//             CP1=0001,
//             CP2=0001,
//             CPM=0001,
//*
//* <QUANTITY> TB           TB1=PRIMARY  TB2=SECONDARY  TBM=DIRECTORY
//             TB1=0001,
//             TB2=0001,
//             TBM=0001,
//*
//* <QUANTITY> CM           CM1=PRIMARY  CM2=SECONDARY  CMM=DIRECTORY
//             CM1=0001,
//             CM2=0001,
//             CMM=0001,
//*
//* <QUANTITY> CF           CF1=PRIMARY  CF2=SECONDARY  CFM=DIRECTORY
//             CF1=0001,
//             CF2=0001,
//             CFM=0001,
//*
//* <QUANTITY> CT           CT1=PRIMARY  CT2=SECONDARY  CTM=DIRECTORY
//             CT1=0001,
//             CT2=0001,
//             CTM=0001,
//*
//* <QUANTITY> CQ           CQ1=PRIMARY  CQ2=SECONDARY  CQM=DIRECTORY
//             CQ1=0001,
//             CQ2=0001,
//             CQM=0001,
//*
//* <QUANTITY> DUMMYPO      DP1=PRIMARY  DP2=SECONDARY  DPM=DIRECTORY
//             DP1=0001,
//             DP2=0001,
//             DPM=0001,
//*
//* <QUANTITY> HASHPI       HP1=PRIMARY  HP2=SECONDARY  <- SEQUENTIAL
//             HP1=0001,
//             HP2=0001,
//*
//* <QUANTITY> HASHJI       HJ1=PRIMARY  HJ2=SECONDARY  <- SEQUENTIAL
//             HJ1=0001,
//             HJ2=0001
//*
//*PI@DEL  EXEC PGM=IEFBR14
//*PI        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PIO1,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*SR@DEL  EXEC PGM=IEFBR14
//*SR        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..SOURCE,
//*           UNIT=&UNIT,
//*           VOLUME=SER=&VOLSER,
//*           DISP=(OLD,DELETE)
⋮
//*JI@DEL  EXEC PGM=IEFBR14
//*JI        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JI,
//*           UNIT=&UNIT,
//*           VOLUME=SER=&VOLSER,
//*           DISP=(OLD,DELETE)
⋮
```

Figure 12 (Part 3 of 5). Sample JCL for Intermediate Data Set Allocation

```
//*JC@DEL  EXEC PGM=IEFBR14
//*JC        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JCL,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*PX@DEL  EXEC PGM=IEFBR14
//*PX        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PX,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*JX@DEL  EXEC PGM=IEFBR14
//*JX        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JX,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*PD@DEL  EXEC PGM=IEFBR14
//*PLIDL     DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PLIDL,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*CD@DEL  EXEC PGM=IEFBR14
//*CBLDL     DD DSNAME=&ANAPFIX..&SYSID..&SUBID..CBLDL,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*JD@DEL  EXEC PGM=IEFBR14
//*JCLDL     DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JCLDL,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*ED@DEL  EXEC PGM=IEFBR14
//*ESYDL     DD DSNAME=&ANAPFIX..&SYSID..&SUBID..ESYDL,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*XP@DEL  EXEC PGM=IEFBR14
//*XP        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..XP,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*XJ@DEL  EXEC PGM=IEFBR14
//*XJ        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..XJ,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*DS@DEL  EXEC PGM=IEFBR14
//*DS        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DS,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*FI@DEL  EXEC PGM=IEFBR14
//*FI        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..FI,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
```

Figure 12 (Part 4 of 5). Sample JCL for Intermediate Data Set Allocation

```
//*SG@DEL  EXEC PGM=IEFBR14
//*SG        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..SG,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*CP@DEL  EXEC PGM=IEFBR14
//*CP        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..CP,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*TB@DEL  EXEC PGM=IEFBR14
//*TB        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..TB,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*CM@DEL  EXEC PGM=IEFBR14
//*CM        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..CM,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*CF@DEL  EXEC PGM=IEFBR14
//*CF        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..CF,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*CT@DEL  EXEC PGM=IEFBR14
//*CT        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..CT,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*CQ@DEL  EXEC PGM=IEFBR14
//*CQ        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..CQ,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*DU@DEL  EXEC PGM=IEFBR14
//*DUMMYPO   DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DUMMYPO,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*HP@DEL  EXEC PGM=IEFBR14
//*HASHPI    DD DSNAME=&ANAPFIX..&SYSID..&SUBID..HASHPI,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//*HJ@DEL  EXEC PGM=IEFBR14
//*HASHJI    DD DSNAME=&ANAPFIX..&SYSID..&SUBID..HASHJI,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
⋮
//        PEND
//*
//IMXALC1 EXEC IMXALC
//
```

Figure 12 (Part 5 of 5). Sample JCL for Intermediate Data Set Allocation

3.4.1.2 IMX2CDC: COBOL Analysis Input Card Creation

This job uses a partitioned data set containing COBOL source programs as input and generates analysis input cards for COBOL analysis job IMX3ANC.

Input Parameters: Table 14 illustrates the parameters that required modification in order for us to create the input analysis cards for our sample scenario. All other parameters retained their initial values.

Table 14. IMX2CDC Input Parameters Requiring Modification for the Sample Application

Parameter	Description	Initial Value	Modified Value
RUNLIB1	IBM C/370 (SEDCLINK) JOBLIB library name	#RUNLIB1	We deleted this JOBLIB DD statement because our system is running under Language Environment.
RUNLIB2	IBM SIBMLINK JOBLIB library name	#RUNLIB2	We deleted this JOBLIB DD statement because our system is running under Language Environment.
UNIT	Unit name of DASD	#DISK	SYSDA
VOLSER	Volume on which data sets are to be allocated	#VOLSER	None
ANAPFIX	Primary analysis data set name qualifier	#ANAHLQ	ITSORS3
SYSID	System ID	#SYSID	SYSITSO
SUBID	Subsystem ID	#SUBID1	SUBSITSO
IMXPFIX	High level qualifiers of the MA2000 product data sets	#IMXHLQ	IMX.V1R3M0
ELBPFIX	High level qualifiers of the Prolog product data sets	#ELBHLQ	IMX.V1R3M0
CBLSRC	COBOL source library	#CBLSRC1	ITSORS2.SAMPAPPL.COBOL
IC*	Primary and secondary allocation quantities for COBOL analysis card	> 1	= 1

Note

The SYSTSIN input card allows you to change the number of programs for which analysis cards are created. If you wish to create analysis cards for more than 5,000 programs, change the processing count of 5,000 which is specified after the job name IMXRR01 in the IC@CRE.SYSTSIN procedure override statement.

Adaptation: We found it necessary to adapt the JCL supplied for job IMX2CDC by eliminating all the initial deletions performed on the data sets allocated in the job. Before the initial execution of IMX2CDC, we removed the following steps at

the outset of the job in order to avoid the JCL error that otherwise would have resulted:

- //IC@DEL EXEC PGM=IEFBR14

Execution JCL: The JCL shown in Figure 13 is an illustration of the JCL we executed to create the COBOL analysis input cards for our sample application.

```
//ITSORS3B JOB ,
//             MSGLEVEL=(1,1),MSGCLASS=H,TIME=(,59),
//             NOTIFY=&SYSUID,USER=&SYSUID
//JOBLIB    DD DSNAME=IMX.V1R3MO.SELBMODO,    <= (IF NECESSARY) CHANGE
//             DISP=(SHR,KEEP)
//*
//*         DD DSNAME=#RUNLIB1,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*         DD DSNAME=#RUNLIB2,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*
//IMXCDC  PROC UNIT=SYSDA,                    <= (IF NECESSARY) CHANGE
//*            VOLSER=#VOLSER,                <================ CHANGE
//             ANAPFIX=ITSORS3,               <= (IF NECESSARY) CHANGE
//             SYSID=SYSITSO,
//             SUBID=SUBSITSO,
//*
//             IMXPFIX='IMX.V1R3MO',          <= (IF NECESSARY) CHANGE
//             ELBPFIX='IMX.V1R3MO',          <= (IF NECESSARY) CHANGE
//*
//             CBLSRC='ITSORS2.SAMPAPPL.COBOL', <= CHANGE
//*
//             UI=TRK,  INCBLO1.....................................
//*
//* <QUANTITY> INCBLO1       IC1=PRIMARY  IC2=SECONDARY  <- SEQUENTIAL
//             IC1=0001,
//             IC2=0001
//*
//* PUT IN COMMENTARY PRIOR TO FIRST EXECUTION
//* REMOVE COMMENTARY FOR SUBSEQUENT EXECUTIONS
//*IC@DEL  EXEC PGM=IEFBR14
//*INCBLO1   DD DSNAME=&ANAPFIX..&SYSID..&SUBID..INCBLO1,
//*             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
//IC@ALC  EXEC PGM=IEFBR14
//INCBLO1   DD DSNAME=&ANAPFIX..&SYSID..&SUBID..INCBLO1,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB),
//             SPACE=(TRK,(0001,0001,0000)),
//             DISP=(NEW,KEEP)
//IC@UCT  EXEC PGM=IEFBR14
//INCBLO1   DD DSNAME=&ANAPFIX..&SYSID..&SUBID..INCBLO1,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,UNCATLG)
```

Figure 13 (Part 1 of 2). Sample JCL for COBOL Analysis Input Card Creation

```
//IC@DEL  EXEC PGM=IEFBR14
//INCBL01   DD DSNAME=&ANAPFIX..&SYSID..&SUBID..INCBL01,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,DELETE)
//*
//IC@CRE  EXEC PGM=IKJEFT01
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//ELBBLLIB  DD DSNAME=&IMXPFIX..SIMXBLK,
//             DISP=(SHR,KEEP)
//SYSENV    DD DSNAME=&IMXPFIX..SIMXENV(SYS@),
//             DISP=(SHR,KEEP)
//SYSPRM    DD DSNAME=&IMXPFIX..SIMXPRM,
//             DISP=(SHR,KEEP)
//INCARD    DD DSNAME=&IMXPFIX..SIMXCARD(IMXAF#2C),
//             DISP=(SHR,KEEP)
//INPDS     DD DSNAME=&CBLSRC,
//             DISP=(SHR,KEEP)
//INCRD01   DD DSNAME=&ANAPFIX..&SYSID..&SUBID..INCBL01,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB),
//             SPACE=(&UI,(&IC1,&IC2,0000)),
//             DISP=(NEW,CATLG)
//INCRD02   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//INCRD03   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//INCRD04   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//INCRD05   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//INCRD06   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//INCRD07   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//INCRD08   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//INCRD09   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//INCRD10   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//*
//        PEND
//*
//IMXCDC1 EXEC IMXCDC
//*
//*
//*
//IC@CRE.SYSTSIN DD *
  IMXRR01      5000
/*
//
```

Figure 13 (Part 2 of 2). Sample JCL for COBOL Analysis Input Card Creation

3.4.1.3 IMX3ANC: COBOL Analysis

This function analyzes COBOL source programs and creates intermediate data in the analysis buffer for subsequent processing by the storing process performed in job IMX4SBC. The SQL, CICS and DL/I analysis functions are automatically invoked during analysis when any corresponding statements are encountered in the source programs.

Note

The following restrictions apply to the COBOL analysis function:

- COBOL source programs must contain no compilation errors when compiled with the VS COBOL II compiler.
- COBOL source containing multiple programs in a source member or in nested programs is not supported.
- The program name must be specified in the PROGRAM-ID paragraph and be identical to the source member name.

Input Parameters: Table 15 shows the parameters that required modification for us to successfully execute the COBOL analysis function against the programs in our sample application. All other parameters retained their original values.

Table 15 (Page 1 of 2). IMX3ANC Input Parameters Requiring Modification for the Sample Application

Parameter	Description	Initial Value	Modified Value
RUNLIB1	IBM C/370 (SEDCLINK) JOBLIB library name	#RUNLIB1	We deleted this JOBLIB DD statement because our system is running under Language Environment.
RUNLIB2	IBM SIBMLINK JOBLIB library name	#RUNLIB2	We deleted this JOBLIB DD statement because our system is running under Language Environment.
UNIT	Unit name of DASD	#DISK	SYSDA
VOLSER	Volume on which data sets are to be allocated	#VOLSER	None
ANAPFIX	Primary analysis data set name qualifier	#ANAHLQ	ITSORS3
SYSID	System ID	#SYSID	SYSITSO
SUBID	Subsystem ID	#SUBID1	SUBSITSO
IMXPFIX	High-level qualifiers of the MA2000 product data sets	#IMXHLQ	IMX.V1R3M0
ELBPFIX	High-level qualifiers of the Prolog product data sets	#ELBHLQ	IMX.V1R3M0
CBLSRC	COBOL source library	#CBLSRC1	ITSORS2.SAMPAPPL.COBOL
CBLCPY	COBOL copybook library	#CBLCPY1	ITSORS2.SAMPAPPL. COPYBOOK

Table 15 (Page 2 of 2). IMX3ANC Input Parameters Requiring Modification for the Sample Application

Parameter	Description	Initial Value	Modified Value
CBLSQL	COBOL DB2 SQL INCLUDE library	#CBLSQL1	None
SYSPRM	Nonstandard system parameter library	#SYSPRM	ITSORS3.MA2000. SYSPRM
UC	Unit of space allocation for PI.B,PX.B,CMSG	CYL	TRK
PI1	Primary allocation quantity for COBOL analysis intermediate buffer	600	14
PI2,PIM, PX*,CM*	Primary and secondary allocation quantities for various libraries	> 1	= 1

Adaptation: We found it necessary to adapt the JCL supplied for job IMX3ANC by eliminating all the initial deletions performed on data sets allocated during the job. Before the initial execution of IMX3ANC, we removed the following steps at the outset of the job in order to avoid the JCL errors that would have resulted otherwise:

- //PI@DEL EXEC PGM=IEFBR14
- //PX@DEL EXEC PGM=IEFBR14
- //CM@DEL EXEC PGM=IEFBR14

Execution JCL: The JCL shown in Figure 14 typifies the JCL we executed to analyze the COBOL source programs of our sample application.

```
//ITSORS3C JOB ,
//            MSGLEVEL=(1,1),MSGCLASS=H,TIME=(,59),
//            NOTIFY=&SYSUID,USER=&SYSUID
//JOBLIB    DD DSNAME=IMX.V1R3M0.SELBMODO,    <= (IF NECESSARY) CHANGE
//            DISP=(SHR,KEEP)
//*
//*         DD DSNAME=#RUNLIB1,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*         DD DSNAME=#RUNLIB2,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*
//*
//          DD DSNAME=IMX.V1R3M0.SIMXLOAD,    <= (IF NECESSARY) CHANGE
//            DISP=(SHR,KEEP)
//*
```

Figure 14 (Part 1 of 4). Sample JCL for COBOL Analysis

```
//IMXANC  PROC UNIT=SYSDA,                    <= (IF NECESSARY) CHANGE
//*            VOLSER=#VOLSER,                <================ CHANGE
//             ANAPFIX=ITSORS3,               <= (IF NECESSARY) CHANGE
//             SYSID=SYSITSO,
//             SUBID=SUBSITSO,
//*
//             IMXPFIX='IMX.V1R3MO',          <= (IF NECESSARY) CHANGE
//             ELBPFIX='IMX.V1R3MO',          <= (IF NECESSARY) CHANGE
//*
//             CBLSRC='ITSORS2.SAMPAPPL.COBOL',   <= CHANGE
//*
//             CBLCPY='ITSORS2.SAMPAPPL.COPYBOOK',   <= CHANGE
//*
//*            CBLSQL='#CBLSQL1',             <= (IF NECESSARY) CHANGE
//*
//             SYSPRM='ITSORS3.MA2000.SYSPRM',
//*
//     UC=TRK, PI.B/PX.B/CMSG.................<= (IF NECESSARY) CHANGE
//*
//*            PI1 ---> <UNUSED PRIMARY QUANTITY WILL NOT BE RELEASED>
//*            PX1 ---> <UNUSED PRIMARY QUANTITY WILL NOT BE RELEASED>
//*            CM1 ---> <UNUSED PRIMARY QUANTITY WILL NOT BE RELEASED>
//*
//             PI1=0014,
//             PI2=0001,
//             PIM=0001,
//*
//             PX1=0001,
//             PX2=0001,
//*
//             CM1=0001,
//             CM2=0001
//*
//*PI@DEL  EXEC PGM=IEFBR14
//*PI@B      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PI.B,
//*            UNIT=&UNIT,
//***          VOLUME=SER=&VOLSER,
//*            DISP=(OLD,DELETE)
//PI@ALC  EXEC PGM=IEFBR14
//PI@B      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PI.B,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PO,LRECL=04096,BLKSIZE=20480,RECFM=FB),
//             SPACE=(TRK,(0001,0000,0001)),
//             DISP=(NEW,KEEP)
//PI@UCT  EXEC PGM=IEFBR14
//PI@B      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PI.B,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,UNCATLG)
//PI@DEL  EXEC PGM=IEFBR14
//PI@B      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PI.B,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,DELETE)
//*
//*PX@DEL  EXEC PGM=IEFBR14
//*PX@B      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PX.B,
//*            UNIT=&UNIT,
//***           VOLUME=SER=&VOLSER,
//*            DISP=(OLD,DELETE)
```

Figure 14 (Part 2 of 4). Sample JCL for COBOL Analysis

```
//PX@ALC  EXEC PGM=IEFBR14
//PX@B      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PX.B,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PS,LRECL=00512,BLKSIZE=23476,RECFM=VB),
//             SPACE=(TRK,(0001,0000,0000)),
//             DISP=(NEW,KEEP)
//PX@UCT  EXEC PGM=IEFBR14
//PX@B      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PX.B,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,UNCATLG)
//PX@DEL  EXEC PGM=IEFBR14
//PX@B      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PX.B,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,DELETE)
//*
//*CM@DEL  EXEC PGM=IEFBR14
//*CMSG      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..CMSG,
//*            UNIT=&UNIT,
//***         VOLUME=SER=&VOLSER,
//*            DISP=(OLD,DELETE)
//CM@ALC  EXEC PGM=IEFBR14
//CMSG      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..CMSG,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PS,LRECL=00512,BLKSIZE=23476,RECFM=VB),
//             SPACE=(TRK,(0001,0000,0000)),
//             DISP=(NEW,KEEP)
//CM@UCT  EXEC PGM=IEFBR14
//CMSG      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..CMSG,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,UNCATLG)
//CM@DEL  EXEC PGM=IEFBR14
//CMSG      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..CMSG,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,DELETE)
//*
//CBLANA  EXEC PGM=IKJEFT01
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//ELBBLLIB  DD DSNAME=&IMXPFIX..SIMXBLK,
//             DISP=(SHR,KEEP)
//SYSENV    DD DSNAME=&IMXPFIX..SIMXENV(SYS@),
//             DISP=(SHR,KEEP)
//SYSPRM    DD DSNAME=&SYSPRM,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&IMXPFIX..SIMXPRM,
//             DISP=(SHR,KEEP)
//INCARD    DD DSNAME=&ANAPFIX..&SYSID..&SUBID..INCBL01,
//             DISP=(SHR,KEEP)
//CBL       DD DSNAME=&CBLSRC,
//             DISP=(SHR,KEEP)
//CPY       DD DSNAME=&CBLCPY,
//             DISP=(SHR,KEEP)
```

Figure 14 (Part 3 of 4). Sample JCL for COBOL Analysis

```
//*SQLINC    DD DSNAME=&CBLSQL,
//*             DISP=(SHR,KEEP)
//PI         DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PI.B,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PO,LRECL=04096,BLKSIZE=20480,RECFM=FB),
//             SPACE=(&UC,(&PI1,&PI2,&PIM)),
//             DISP=(NEW,CATLG)
//PIM        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PI01,
//             DISP=(SHR,KEEP)
//PX         DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PX.B,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PS,LRECL=00512,BLKSIZE=23476,RECFM=VB),
//             SPACE=(&UC,(&PX1,&PX2,0000)),
//             DISP=(NEW,CATLG)
//MSG        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..CMSG,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PS,LRECL=00512,BLKSIZE=23476,RECFM=VB),
//             SPACE=(&UC,(&CM1,&CM2,0000)),
//             DISP=(NEW,CATLG)
//*
//         PEND
//*
//IMXANC1 EXEC IMXANC
//*
//CBLANA.SYSTSIN DD *
  IMXRR01      APEN
/*
//CBLANA.CPY     DD
//*
//*CBLANA.CPY  DD DSNAME=CBLCPY1A,
//*               DISP=(SHR,KEEP)
//*          DD DSNAME=CBLCPY1A,
//*             DISP=(SHR,KEEP)
//*          DD DSNAME=CBLCPY1C,              <= (IF NECESSARY)
//*             DISP=(SHR,KEEP)                  CONCATENATE DATA SET
//*
//CBLANA.SQLINC  DD
//*
//*          DD DSNAME=CBLSQL1A,              <= (IF NECESSARY)
//*             DISP=(SHR,KEEP)                   CONCATENATE DATA SET
//*          DD DSNAME=CBLSQL1B,              <= (IF NECESSARY)
//*             DISP=(SHR,KEEP)                   CONCATENATE DATA SET
//*          DD DSNAME=CBLSQL1C,              <= (IF NECESSARY)
//*             DISP=(SHR,KEEP)                   CONCATENATE DATA SET
//*
//
```

Figure 14 (Part 4 of 4). Sample JCL for COBOL Analysis

3.4.2 Creating the COBOL Retrieval Environment

A COBOL-related retrieval environment is initially created for each individual subsystem within the system being processed. In a subsequent step, the subsystem intermediate data resulting from this preparatory process is used to create COBOL-related intermediate data at the system level. Intermediate data management functions include the generation of a copy of the original source code library and program-related dictionaries as well as validation tasks that help to ensure that no system components have been omitted from processing. The activities making up the retrieval environment creation procedure are prerequisites for the invocation of the individual and multiple retrieval functions available in the MA2000 on-line environment.

The JCL we used for the establishment of the retrieval environment for our application was taken from the SIMXLIB1 library, since only one retrieval view

(= subsystem ID) of the sample was required. If, however, the system you are processing comprises more than one subsystem, use the members supplied in SIMXLIB2 or SIMXLIB3 as appropriate.

We performed the following three jobs to establish the retrieval environment for our sample application.

1. IMX4SBC
2. IMX5DCC
3. IMX6TBL

3.4.2.1 IMX4SBC: Storing Subsystem COBOL Intermediate Data

Job IMX4SBC creates intermediate COBOL-related data for a subsystem. During this process, data is copied from the buffer data sets created during the analysis process to subsystem intermediate data sets, where it is stored in preparation for the system intermediate data set creation process that follows. Any data that caused an error during the analysis phase is not stored in the subsystem intermediate data sets. As this intermediate data is created at the subsystem level, the JCL to execute this process must be obtained from the sample JCL library appropriate to the number of subsystems in your concatenation sequence. You must specify the priority of the subsystems within a system by means of the input parameters provided. For an explanation of these parameters, see 3.1.3.2, “Defining Systems and Subsystems within the MA2000 Environment” on page 21.

Job IMX4SBC also allows you to delete members erroneously registered in a subsystem. For details about eliminating members that have been mistakenly registered, see Chapter 4, ″Preparing to Retrieve Analysis Data″ in the manual *Maintenance 2000 Guide and Reference Version 1, Release 3*.

Input Parameters: Table 16 illustrates the parameters that required modification in order for us to successfully execute the subsystem intermediate data storage process for the COBOL programs in our sample application. All other parameters retained their original values.

Table 16 (Page 1 of 2). IMX4SBC Input Parameters Requiring Modification for the Sample Application

Parameter	Description	Initial Value	Modified Value
RUNLIB1	IBM C/370 (SEDCLINK) JOBLIB library name	#RUNLIB1	We deleted this JOBLIB DD statement because our system is running under Language Environment.
RUNLIB2	IBM SIBMLINK JOBLIB library name	#RUNLIB2	We deleted this JOBLIB DD statement because our system is running under Language Environment.
UNIT	Unit name of DASD	#DISK	SYSDA
VOLSER	Volume on which data sets are to be allocated	#VOLSER	None
ANAPFIX	Primary data set name qualifier	#ANAHLQ	ITSORS3
SYSID	System ID	#SYSID	SYSITSO
SUBID	Subsystem ID	#SUBID1	SUBSITSO

Table 16 (Page 2 of 2). IMX4SBC Input Parameters Requiring Modification for the Sample Application

Parameter	Description	Initial Value	Modified Value
IMXPFIX	High-level qualifiers of the MA2000 product data sets	#IMXHLQ	IMX.V1R3M0
ELBPFIX	High-level qualifiers of the Prolog product data sets	#ELBHLQ	IMX.V1R3M0
CBLSRC	COBOL source library	#CBLSRC1	ITSORS2.SAMPAPPL.COBOL
UD	Unit of space allocation for program-related dictionary	CYL	TRK
UW	Unit of space allocation for buffer	CYL	TRK
PI1,CS1	Primary allocation quantity for PI01 and SOURCE data buffer	30	15
PI2,CS2,	Secondary allocation quantities for PI01 and SOURCE data buffer	> 1	= 1
PIM,CSM	PI01 and SOURCE data buffer directory allocation	> 1	= 1

Adaptation: We found it necessary to adapt the JCL supplied for job IMX4SBC by eliminating all the initial deletions performed on data sets allocated during the job. Before the initial execution of IMX4SBC, we removed the following step at the outset of the job in order to avoid the JCL error that otherwise would have resulted:

- //DP@DEL EXEC PGM=IEFBR14

Execution JCL: Figure 15 on page 58 shows the JCL which we executed in order to store the subsystem intermediate COBOL data for our sample application COBOL programs.

```
//ITSORS3D JOB ,
//            MSGLEVEL=(1,1),MSGCLASS=H,TIME=(,59),
//            NOTIFY=&SYSUID,USER=&SYSUID
//JOBLIB    DD DSNAME=IMX.V1R3M0.SELBMODO,    <= (IF NECESSARY) CHANGE
//            DISP=(SHR,KEEP)
//*
//*         DD DSNAME=#RUNLIB1,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*         DD DSNAME=#RUNLIB2,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*
//*
//          DD DSNAME=IMX.V1R3M0.SIMXLOAD,    <= (IF NECESSARY) CHANGE
//             DISP=(SHR,KEEP)
//*
//*
//IMXSBC  PROC UNIT=SYSDA,                     <= (IF NECESSARY) CHANGE
//*            VOLSER=#VOLSER,                 <================ CHANGE
//             ANAPFIX=ITSORS3,                <= (IF NECESSARY) CHANGE
//             SYSID=SYSITSO,
//             SUBID=SUBSITSO,
//*
//             IMXPFIX='IMX.V1R3M0',           <= (IF NECESSARY) CHANGE
//             ELBPFIX='IMX.V1R3M0',           <= (IF NECESSARY) CHANGE
//*
//             CBLSRC='ITSORS2.SAMPAPPL.COBOL',   <= CHANGE
//*
//             PIVOLS=01,                      NUMBER OF PDS OF DATA
//*
//     UD=TRK, DICP.........................<= (IF NECESSARY) CHANGE
//*
//* <QUANTITY> DICP          DP1=PRIMARY  DP2=SECONDARY  <- SEQUENTIAL
//             DP1=0001,
//             DP2=0001,
//*
//     UW=TRK, SpaceForCBLSRCWorkingDataSet...<= (IF NECESSARY) CHANGE
//*            SpaceFor<<PI>>WorkingDataSet.....................
//*
//* <QUANTITY> ********      CS1=PRIMARY  CS2=SECONDARY  CSM=DIRECTORY
//             CS1=0015,                       <= (IF NECESSARY) CHANGE
//             CS2=0001,                       <= (IF NECESSARY) CHANGE
//             CSM=0001,                       <= (IF NECESSARY) CHANGE
//*
//* <QUANTITY> ********      PI1=PRIMARY  PI2=SECONDARY  PIM=DIRECTORY
//             PI1=0015,                       <= (IF NECESSARY) CHANGE
//             PI2=0001,                       <= (IF NECESSARY) CHANGE
//             PIM=0001,                       <= (IF NECESSARY) CHANGE
//*
//*DP@DEL  EXEC PGM=IEFBR14
//*DICP      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DICP,
//*            UNIT=&UNIT,
//***           VOLUME=SER=&VOLSER,
//*            DISP=(OLD,DELETE)
//DP@ALC  EXEC PGM=IEFBR14
//DICP      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DICP,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PS,LRECL=00080,BLKSIZE=03200,RECFM=FB),
//             SPACE=(TRK,(0001,0001,0000)),
//             DISP=(NEW,KEEP)
```

Figure 15 (Part 1 of 7). Sample JCL for Subsystem Intermediate COBOL Data Storage Process

```
//DP@UCT  EXEC PGM=IEFBR14
//DICP      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DICP,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,UNCATLG)
//DP@DEL  EXEC PGM=IEFBR14
//DICP      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DICP,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,DELETE)
//*
//PX@SRT  EXEC PGM=ICEMAN
//SYSIN     DD DUMMY
//SORTIN    DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PX.B,
//             DISP=(SHR,KEEP)
//SORTOUT   DD DSNAME=&&PX1B,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00512,BLKSIZE=23476,RECFM=VB),
//             SPACE=(CYL,(0010,0010,0000),RLSE),
//             DISP=(NEW,PASS)
//SORTWK01  DD UNIT=&UNIT,
//             SPACE=(CYL,(0010,0010,0000))
//SORTWK02  DD UNIT=&UNIT,
//             SPACE=(CYL,(0010,0010,0000))
//SORTWK03  DD UNIT=&UNIT,
//             SPACE=(CYL,(0010,0010,0000))
//SYSOUT    DD SYSOUT=*
//*
//PX@RAS  EXEC PGM=IKJEFT01,
//             COND=((0,NE,PX@SRT))
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//ELBBLLIB  DD DSNAME=&IMXPFIX..SIMXBLK,
//             DISP=(SHR,KEEP)
//SYSENV    DD DSNAME=&IMXPFIX..SIMXENV(SYS@),
//             DISP=(SHR,KEEP)
//SYSPRM    DD DSNAME=&IMXPFIX..SIMXPRM,
//             DISP=(SHR,KEEP)
//INCARD    DD DSNAME=&IMXPFIX..SIMXCARD(IMXAN#01),
//             DISP=(SHR,KEEP)
//DXBT1     DD DSNAME=&&PX1B,
//             DISP=(OLD,DELETE)
//DXBT2     DD DSNAME=&&PX2B,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00512,BLKSIZE=23476,RECFM=VB),
//             SPACE=(CYL,(0010,0010,0000),RLSE),
//             DISP=(NEW,PASS)
//DL        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..CBLDL,
//             DISP=(SHR,KEEP)
//COPYI     DD DSNAME=&&CP@I,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00080,BLKSIZE=03200,RECFM=FB),
//             SPACE=(CYL,(0010,0010,0000),RLSE),
//             DISP=(NEW,PASS)
//*
```

Figure 15 (Part 2 of 7). Sample JCL for Subsystem Intermediate COBOL Data Storage Process

```
//PX@UPD  EXEC PGM=IKJEFT01,
//             COND=((0,NE,PX@SRT),(4,LT,PX@RAS))
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//ELBBLLIB  DD DSNAME=&IMXPFIX..SIMXBLK,
//             DISP=(SHR,KEEP)
//SYSENV    DD DSNAME=&IMXPFIX..SIMXENV(SYS@),
//             DISP=(SHR,KEEP)
//SYSPRM    DD DSNAME=&IMXPFIX..SIMXPRM,
//             DISP=(SHR,KEEP)
//INCARD    DD DSNAME=&IMXPFIX..SIMXCARD(IMXAN#02),
//             DISP=(SHR,KEEP)
//DXBT2     DD DSNAME=&&PX2B,
//             DISP=(OLD,DELETE)
//DX        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PX,
//             DISP=(SHR,KEEP)
//DL        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..CBLDL,
//             DISP=(SHR,KEEP)
//DXT       DD DSNAME=&&PX3B,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00512,BLKSIZE=23476,RECFM=VB),
//             SPACE=(CYL,(0010,0010,0000),RLSE),
//             DISP=(NEW,PASS)
//DIC       DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DICP,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PS,LRECL=00080,BLKSIZE=03200,RECFM=FB),
//             SPACE=(&UD,(&DP1,&DP2,0000)),
//             DISP=(NEW,CATLG)
//*
//HP@UPD  EXEC PGM=IMXBN01,
//             PARM='&PIVOLS.',
//             REGION=1024K,
//             COND=((0,NE,PX@SRT),(4,LT,PX@RAS))
//HASHIDD   DD DSNAME=&ANAPFIX..&SYSID..&SUBID..HASHPI,
//             DISP=(SHR,KEEP)
//COPYIDD   DD DSNAME=&&CP@I,
//             DISP=(OLD,DELETE)
//HASHODD   DD DSNAME=&&HP@B,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00080,BLKSIZE=03200,RECFM=FB),
//             SPACE=(CYL,(0010,0010,0000),RLSE),
//             DISP=(NEW,PASS)
//COPYODD   DD DSNAME=&&CP@O,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00080,BLKSIZE=03200,RECFM=FB),
//             SPACE=(CYL,(0010,0010,0000),RLSE),
//             DISP=(NEW,PASS)
//DELIDD    DD DUMMY
//DELODD    DD DUMMY
//MMSGIDD   DD DUMMY
//SYSPRINT  DD SYSOUT=*
//*
```

Figure 15 (Part 3 of 7). Sample JCL for Subsystem Intermediate COBOL Data Storage Process

```
//PITCPY  EXEC PGM=IEBCOPY,
//             COND=((0,NE,PX@SRT),(4,LT,PX@RAS),(0,NE,HP@UPD))
//SYSIN     DD DSNAME=&&CP@O,
//             DISP=(OLD,PASS)
//XIB       DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PI.B,
//             DISP=(SHR,KEEP)
//XIO1      DD DSNAME=&&PI@B,
//             UNIT=&UNIT,
//             SPACE=(&UW,(&PI1,&PI2,&PIM),RLSE),
//             DISP=(NEW,PASS)
//SYSPRINT  DD SYSOUT=*
//*
//PI@RAS  EXEC PGM=IKJEFT01,
//             COND=((0,NE,PX@SRT),(4,LT,PX@RAS),(0,NE,HP@UPD),
//            (0,NE,PITCPY))
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//XIO1      DD DSNAME=&&PI@B,
//             DISP=(OLD,PASS)
//MCARD     DD DSNAME=&ANAPFIX..&SYSID..&SUBID..CBLDL,
//             DISP=(SHR,KEEP)
//*
//PI@CPY  EXEC PGM=IEBCOPY,
//             COND=((0,NE,PX@SRT),(4,LT,PX@RAS),(0,NE,HP@UPD),
//            (0,NE,PITCPY),(0,NE,PI@RAS))
//SYSIN     DD DUMMY
//XIO1      DD DSNAME=&&PI@B,
//             DISP=(OLD,DELETE)
//XI        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PIO1,
//             DISP=(SHR,KEEP)
//SYSPRINT  DD SYSOUT=*
//*
//PI@CPS  EXEC PGM=IEBCOPY,
//             COND=((0,NE,PX@SRT),(4,LT,PX@RAS),(0,NE,HP@UPD),
//            (0,NE,PITCPY),(0,NE,PI@RAS),
//            (0,NE,PI@CPY))
//SYSIN     DD DUMMY
//XI        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PIO1,
//             DISP=(SHR,KEEP)
//SYSPRINT  DD SYSOUT=*
//*
//SRTCPY  EXEC PGM=IEBCOPY,
//             COND=((0,NE,PX@SRT),(4,LT,PX@RAS),(0,NE,HP@UPD))
//SYSIN     DD DSNAME=&&CP@O,
//             DISP=(OLD,DELETE)
//XIB       DD DSNAME=&CBLSRC,
//             DISP=(SHR,KEEP)
//XIO1      DD DSNAME=&&SR@B,
//             UNIT=&UNIT,
//             SPACE=(&UW,(&CS1,&CS2,&CSM),RLSE),
//             DISP=(NEW,PASS)
//SYSPRINT  DD SYSOUT=*
//*
```

Figure 15 (Part 4 of 7). Sample JCL for Subsystem Intermediate COBOL Data Storage Process

```
//SR@RAS  EXEC PGM=IKJEFT01,
//             COND=((0,NE,PX@SRT),(4,LT,PX@RAS),(0,NE,HP@UPD),
//           (0,NE,SRTCPY))
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//XIO1      DD DSNAME=&&SR@B,
//             DISP=(OLD,PASS)
//MCARD     DD DSNAME=&ANAPFIX..&SYSID..&SUBID..CBLDL,
//             DISP=(SHR,KEEP)
//*
//SR@CPY  EXEC PGM=IEBCOPY,
//             COND=((0,NE,PX@SRT),(4,LT,PX@RAS),(0,NE,HP@UPD),
//           (0,NE,SRTCPY),(0,NE,SR@RAS))
//SYSIN     DD DUMMY
//XIO1      DD DSNAME=&&SR@B,
//             DISP=(OLD,DELETE)
//XI        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..SOURCE,
//             DISP=(SHR,KEEP)
//SYSPRINT  DD SYSOUT=*
//*
//SR@CPS  EXEC PGM=IEBCOPY,
//             COND=((0,NE,PX@SRT),(4,LT,PX@RAS),(0,NE,HP@UPD),
//           (0,NE,SRTCPY),(0,NE,SR@RAS),
//           (0,NE,SR@CPY))
//SYSIN     DD DUMMY
//XI        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..SOURCE,
//             DISP=(SHR,KEEP)
//SYSPRINT  DD SYSOUT=*
//*
//HP@CPY  EXEC PGM=IEBGENER,
//             COND=((0,NE,PX@SRT),(4,LT,PX@RAS),(0,NE,HP@UPD))
//SYSIN     DD DUMMY
//SYSUT1    DD DSNAME=&&HP@B,
//             DISP=(OLD,DELETE)
//SYSUT2    DD DSNAME=&ANAPFIX..&SYSID..&SUBID..HASHPI,
//             DISP=(OLD,KEEP)
//SYSPRINT  DD SYSOUT=*
//*
//PX3SRT  EXEC PGM=ICEMAN,
//             COND=((0,NE,PX@SRT),(4,LT,PX@RAS),(0,NE,PX@UPD))
//SYSIN     DD DUMMY
//SORTIN    DD DSNAME=&&PX3B,
//             DISP=(OLD,DELETE)
//SORTOUT   DD DSNAME=&&PX@B,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00512,BLKSIZE=23476,RECFM=VB),
//             SPACE=(CYL,(0010,0010,0000),RLSE),
//             DISP=(NEW,PASS)
//SORTWK01  DD UNIT=&UNIT,
//             SPACE=(CYL,(0010,0010,0000))
//SORTWK02  DD UNIT=&UNIT,
//             SPACE=(CYL,(0010,0010,0000))
//SORTWK03  DD UNIT=&UNIT,
//             SPACE=(CYL,(0010,0010,0000))
//SYSOUT    DD SYSOUT=*
//*
```

Figure 15 (Part 5 of 7). Sample JCL for Subsystem Intermediate COBOL Data Storage Process

```
//PX@CPY  EXEC PGM=IEBGENER,
//             COND=((0,NE,PX@SRT),(4,LT,PX@RAS),(0,NE,PX@UPD),
//            (0,NE,PX3SRT))
//SYSIN     DD DUMMY
//SYSUT1    DD DSNAME=&&PX@B,
//             DISP=(OLD,DELETE)
//SYSUT2    DD DSNAME=&ANAPFIX..&SYSID..&SUBID..PX,
//             DISP=(SHR,KEEP)
//SYSPRINT  DD SYSOUT=*
//*
//        PEND
//*
//IMXSBC1 EXEC IMXSBC
//*
//PX@SRT.SYSIN   DD *
  SORT         FIELDS=(11,8,CH,A),EQUALS
/*
//PX@RAS.SYSTSIN DD *
  IMXRRO1
/*
//PX@UPD.SYSTSIN DD *
  IMXRRO1
/*
//*
//PI@RAS.SYSTSIN DD *
  IMXANO1
/*
//*
//PI@CPY.SYSIN   DD *
  COPY         INDD=((XIO1,R)),OUTDD=XI
/*
//PI@CPS.SYSIN   DD *
  COPY         INDD=XI,OUTDD=XI
/*
//*
//SR@RAS.SYSTSIN DD *
  IMXANO1
/*
//*
//SR@CPY.SYSIN   DD *
  COPY         INDD=((XIO1,R)),OUTDD=XI
/*
//SR@CPS.SYSIN   DD *
  COPY         INDD=XI,OUTDD=XI
/*
//PX3SRT.SYSIN   DD *
  SORT         FIELDS=(11,8,CH,A),EQUALS
/*
//
```

Figure 15 (Part 6 of 7). Sample JCL for Subsystem Intermediate COBOL Data Storage Process

3.4.2.2 IMX5DCC: COBOL Program Dictionary Creation

Job IMX5DCC creates the COBOL-related dictionaries that constitute a portion of the system intermediate data. As the program dictionary creation process takes place at the system rather than at the individual subsystem level, job IMX5DCC must only be executed once per system. After obtaining the JCL for this job from the sample JCL library for the number of subsystems in the subsystem concatenation sequence, you must specify the priority of the subsystems being processed by means of the input parameters provided. Dictionaries are then created in accordance with the retrieval views you have defined by means of the subsystem ID input parameters. For an explanation of how to define the priority of subsystems in the subsystem concatenation sequence, see 3.1.3.2, “Defining Systems and Subsystems within the MA2000 Environment” on page 21.

The following types of dictionaries are created and updated by this process:

- Program-related dictionary
- Input-output segment dictionary
- Input-output file dictionary
- COBOL copybook dictionary
- CICS file dictionary
- CICS map dictionary
- CICS transaction dictionary
- CICS queue dictionary
- DB2 table dictionary

Input Parameters: Table 17 shows the parameters that required modification in order for us to successfully create the COBOL-related dictionaries for our sample application programs. All other parameters remained unchanged.

Table 17. IMX5DCC Input Parameters Requiring Modification for the Sample Application

Parameter	Description	Initial Value	Modified Value
RUNLIB1	IBM C/370 (SEDCLINK) JOBLIB library name	#RUNLIB1	We deleted this JOBLIB DD statement because our system is running under Language Environment.
RUNLIB2	IBM SIBMLINK JOBLIB library name	#RUNLIB2	We deleted this JOBLIB DD statement because our system is running under Language Environment.
UNIT	Unit name of DASD	#DISK	SYSDA
ANAPFIX	Primary data set name qualifier	#ANAHLQ	ITSORS3
SYSID	System ID	#SYSID	SYSITSO
SUBID	Subsystem ID	#SUBID1	SUBSITSO
IMXPFIX	High-level qualifiers of the MA2000 product data sets	#IMXHLQ	IMX.V1R3M0
ELBPFIX	High-level qualifiers of the Prolog product data sets	#ELBHLQ	IMX.V1R3M0

Adaptation: We did not find it necessary to adapt the JCL supplied for job IMX5DCC as no initial delete operations were performed on data sets at the outset of the job.

Execution JCL: As the JCL for job IMX5DCC required no modifications, the JCL shown in Figure 16 on page 65 illustrates only that portion of the JCL in which we modified the necessary input parameters for procedure IMXDCC, which is executed in job IMX5DCC.

```
//ITSORS3E JOB ,
//            MSGLEVEL=(1,1),MSGCLASS=H,TIME=(,59),
//            NOTIFY=&SYSUID,USER=&SYSUID
//JOBLIB    DD DSNAME=IMX.V1R3M0.SELBMODO,    <= (IF NECESSARY) CHANGE
//            DISP=(SHR,KEEP)
//*
//*         DD DSNAME=#RUNLIB1,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*         DD DSNAME=#RUNLIB2,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*
//          DD DSNAME=IMX.V1R3M0.SIMXLOAD,    <= (IF NECESSARY) CHANGE
//            DISP=(SHR,KEEP)
//*
//IMXDCC  PROC UNIT=SYSDA,                    <= (IF NECESSARY) CHANGE
//*
//            ANAPFIX=ITSORS3,                <= (IF NECESSARY) CHANGE
//            SYSID=SYSITSO,
//            SUBID=SUBSITSO,
//*
//            IMXPFIX='IMX.V1R3M0',           <= (IF NECESSARY) CHANGE
//            ELBPFIX='IMX.V1R3M0'            <= (IF NECESSARY) CHANGE
//*
```

Figure 16. Sample JCL for Program Dictionary Creation

3.4.2.3 IMX6TBL: Table Registration

Job IMX6TBL is used to identify a program written in a language not supported by MA2000 to the impact analysis process by registering information about it in a table. Generally, if a program to be analyzed contains an external call to another program written in an unsupported language, MA2000 processing is discontinued at the point the external call is performed. The program information in this table is used to enable MA2000 to continue the impact analysis process beyond the point of the call. The information found here is also referred to during the result retrieval process. For details regarding what type of information can be specified to MA2000 for each registration unit, see Chapter 4.4, *"*Creating the Retrieval Environment - Table Registration*"* in the manual *Maintenance 2000 Guide and Reference Version 1 Release 3*.

Note

Job IMX6TBL must be executed even if there is no data to be registered.

Input Parameters: Table 18 shows the parameters that required modification in order for us to successfully complete the table registration process. All other input parameters retained their original values.

Table 18 (Page 1 of 2). IMX6TBL Input Parameters Requiring Modification for the Sample Application

Parameter	Description	Initial Value	Modified Value
RUNLIB1	IBM C/370 (SEDCLINK) JOBLIB library name	#RUNLIB1	We deleted this JOBLIB DD statement because our system is running under Language Environment.
RUNLIB2	IBM SIBMLINK JOBLIB library name	#RUNLIB2	We deleted this JOBLIB DD statement because our system is running under Language Environment.

Table 18 (Page 2 of 2). IMX6TBL Input Parameters Requiring Modification for the Sample Application

Parameter	Description	Initial Value	Modified Value
RUNLIB3	PLILINK JOBLIB library name	#RUNLIB3	We deleted this JOBLIB DD statement because our system is running under Language Environment.
UNIT	Unit name of DASD	#DISK	SYSDA
VOLSER	Volume on which data sets are to be allocated	#VOLSER	None
ANAPFIX	Primary data set name qualifier	#ANAHLQ	ITSORS3
SYSID	System ID	#SYSID	SYSITSO
UT	Unit of allocation for table data set	CYL	TRK
TB*	Allocation quantities for table data set	> 1	= 1

Adaptation: We found it necessary to adapt the JCL supplied for job IMX6TBL by eliminating the initial deletion performed on the table data set allocated during the job. Before the initial execution of IMX6TBL, we removed the following step at the outset of the job in order to avoid the JCL error that otherwise would have resulted from its execution:

- //TB@DEL EXEC PGM=IEFBR14

Execution JCL: Figure 17 on page 67 shows the JCL we executed in order to perform the table registration process for our sample application. Input data should be supplied in the SYSIN DD card.

```
//ITSORS3F JOB ,
//             MSGLEVEL=(1,1),MSGCLASS=H,TIME=(,59),
//             NOTIFY=&SYSUID,USER=&SYSUID
//JOBLIB    DD DSNAME=IMX.V1R3M0.SELBMODO,    <= (IF NECESSARY) CHANGE
//             DISP=(SHR,KEEP)
//*
//*         DD DSNAME=#RUNLIB1,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*         DD DSNAME=#RUNLIB2,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*
//          DD DSNAME=IMX.V1R3M0.SIMXLOAD,    <= (IF NECESSARY) CHANGE
//             DISP=(SHR,KEEP)
//*
//*         DD DSNAME=#RUNLIB3,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*
//IMXTBL  PROC UNIT=SYSDA,                     <= (IF NECESSARY) CHANGE
//*            VOLSER=#VOLSER,
//             ANAPFIX=ITSORS3,                <= (IF NECESSARY) CHANGE
//*
//             SYSID=SYSITSO,
//*
//             IMXPFIX='IMX.V1R3M0',           <= (IF NECESSARY) CHANGE
//             ELBPFIX='IMX.V1R3M0',           <= (IF NECESSARY) CHANGE
//*
//     UT=TRK, TBL.........................<= (IF NECESSARY) CHANGE
//*
//* <QUANTITY> TBL           TB1=PRIMARY  TB2=SECONDARY  TBM=DIRECTORY
//             TB1=0001,
//             TB2=0001,
//             TBM=0001
//*
//*TB@DEL  EXEC PGM=IEFBR14
//*TBL       DD DSNAME=&ANAPFIX..&SYSID..TBL,
//*            UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//*            DISP=(OLD,DELETE)
//TB@ALC  EXEC PGM=IEFBR14
//TBL       DD DSNAME=&ANAPFIX..&SYSID..TBL,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PO,LRECL=04096,BLKSIZE=20480,RECFM=FB),
//             SPACE=(TRK,(0001,0000,0001)),
//             DISP=(NEW,KEEP)
//TB@UCT  EXEC PGM=IEFBR14
//TBL       DD DSNAME=&ANAPFIX..&SYSID..TBL,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,UNCATLG)
//TB@DEL  EXEC PGM=IEFBR14
//TBL       DD DSNAME=&ANAPFIX..&SYSID..TBL,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,DELETE)
//*
```

Figure 17 (Part 1 of 2). Sample JCL for the Table Registration Process

```
//TB@ECG  EXEC PGM=IMXAO01
//SYSIN     DD DUMMY
//TBL       DD DSNAME=&&TB@W,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00108,BLKSIZE=01080,RECFM=FB),
//             SPACE=(CYL,(0010,0010,0000),RLSE),
//             DISP=(NEW,PASS)
//SYSPRINT  DD SYSOUT=*
//*
//TB@ECE  EXEC PGM=IKJEFT01,
//             COND=((0,NE,TB@ECG))
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//ELBBLLIB  DD DSNAME=&IMXPFIX..SIMXBLK,
//             DISP=(SHR,KEEP)
//SYSENV    DD DSNAME=&IMXPFIX..SIMXENV(SYS@),
//             DISP=(SHR,KEEP)
//SYSPRM    DD DSNAME=&IMXPFIX..SIMXPRM,
//             DISP=(SHR,KEEP)
//INCARD    DD DSNAME=&IMXPFIX..SIMXCARD(IMXAO#02),
//             DISP=(SHR,KEEP)
//TBLIN     DD DSNAME=&&TB@W,
//             DISP=(OLD,DELETE)
//TBLOUT    DD DSNAME=&ANAPFIX..&SYSID..TBL,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PO,LRECL=04096,BLKSIZE=20480,RECFM=FB),
//             SPACE=(&UT,(&TB1,&TB2,&TBM)),
//             DISP=(NEW,CATLG)
//*
//        PEND
//*
//IMXTBL  EXEC IMXTBL
//*
//TB@ECE.SYSTSIN DD *
  IMXRR01
/*
//
```

Figure 17 (Part 2 of 2). Sample JCL for the Table Registration Process

3.4.2.4 IMX7CKM: Checking Unregistered Members

Job IMX7CKM searches JCL members and programs to identify executed or called programs that have not been registered as belonging to the subsystem in question. This job uses the program and JCL intermediate data as input and writes the results to a message file. Specify which programs should be excluded from the search on the SYSIN card.

Because of the limited scope of our sample application, the execution of this job was not required. The processing of a larger system, however, would normally necessitate that this job be executed.

3.4.2.5 IMX7CKD: Checking Unspecified DL/I Calls

Job IMX7CKD searches programs for DL/I call statements and segment input areas for which the associated segments cannot be identified.

Because our sample application included no DL/I calls, the execution of this job was not necessary.

3.5 JCL Processing

This section reviews in detail the activities we performed in order to establish the MA2000 analysis and retrieval environments for the JCL elements of our sample application. Each JCL-related batch job outlined in 3.3.1, "Execution JCL" on page 36, is examined more completely, to identify what input parameters might require adaptation in your environment, as well as to indicate where modifications to the supplied JCL must be performed prior to execution. In addition, a sample of the execution JCL we used to process the JCL members of our sample application is provided to illustrate the customization process.

Note

Before submitting a job, you may have to eliminate the initial data set deletion steps performed in the JCL. See 3.3.1, "Execution JCL" on page 36 for a discussion of how to eliminate potential JCL errors.

3.5.1 Creating the JCL Analysis Environment

JCL analysis is performed for each subsystem within the system being processed. The activities that make up the JCL analysis function are prerequisites for the subsequent creation of the JCL-related dictionaries in the retrieval environment, the establishment of which is, in turn, a prerequisite for the invocation of the on-line JCL-related retrieval result functions.

The JCL we used for the creation of the analysis environment for our sample application JCL members was taken from the SIMXLIB1 data set, because our sample consisted of only one subsystem. If the system you are processing comprises more than one subsystem, use the members located in SIMXLIB2 or SIMXLIB3, as appropriate. We executed the following three jobs to establish the JCL analysis environment for our system:

1. IMX1ALC
2. IMX2CDJ
3. IMX3ANJ

3.5.1.1 IMX1ALC: Intermediate Data Set Allocation

Job IMX1ALC allocates and catalogs the intermediate data sets required by MA2000. For a list of MA2000 data sets, see 3.3.3.1, "Analysis and Retrieval Environment Data Sets" on page 38. Since job IMX1ALC must only be executed once for each system being analyzed, if you executed it before in conjunction with program source analysis, you need not submit this job again. If, however, you decide to begin with the creation of the JCL analysis environment, the execution of this job is a prerequisite for all subsequent steps. For a detailed discussion of job IMX1ALC, see 3.4.1.1, "IMX1ALC: Intermediate Data Set Allocation" on page 42.

3.5.1.2 IMX2CDJ: JCL Analysis Input Card Creation

This job uses a partitioned data set containing JCL source as input and generates analysis input cards for JCL analysis job IMX3ANJ.

Input Parameters: Table 19 presents the parameters that required modification in order for us to create the JCL input analysis cards for our sample scenario. All other parameters retained their initial values.

Table 19. IMX2CDJ Input Parameters Requiring Modification for the Sample Application

Parameter	Description	Initial Value	Modified Value
RUNLIB1	IBM C/370 (SEDCLINK) JOBLIB library name	#RUNLIB1	We deleted this JOBLIB DD statement because our system is running under Language Environment.
RUNLIB2	IBM SIBMLINK JOBLIB library name	#RUNLIB2	We deleted this JOBLIB DD statement because our system is running under Language Environment.
UNIT	Unit name of DASD	#DISK	SYSDA
VOLSER	Volume on which data sets are to be allocated	#VOLSER	None
ANAPFIX	Primary analysis data set name qualifier	#ANAHLQ	ITSORS3
SYSID	System ID	#SYSID	SYSITSO
SUBID	Subsystem ID	#SUBID1	SUBSITSO
IMXPFIX	High-level qualifiers of the MA2000 product data sets	#IMXHLQ	IMX.V1R3M0
ELBPFIX	High-level qualifiers of the Prolog product data sets	#ELBHLQ	IMX.V1R3M0
JCLSRC	JCL source library	#JCLSRC1	ITSORS2.SAMPAPPL.JCL
IJ*	Primary and secondary allocation quantities for JCL analysis card	> 1	= 1

Note

The SYSTSIN input card allows you to change the number of items for which analysis cards are created. If you wish to create analysis cards for more than 5,000 items, change the processing count of 5,000 specified after the job name IMXRR01 in the IJ@CRE.SYSTSIN procedure override statement.

Adaptation: We found it necessary to adapt the JCL supplied for job IMX2CDJ by eliminating all the initial deletions on the data sets allocated in the job. Before the initial execution of IMX2CDJ, we removed the following step at the outset of the job in order to avoid the JCL error that otherwise would have resulted:

- //IJ@DEL EXEC PGM=IEFBR14

Execution JCL: Figure 18 shows the JCL we executed in order to create the JCL analysis input cards for our sample application.

```
//ITSORS3A JOB ,
//             MSGLEVEL=(1,1),MSGCLASS=H,TIME=(,59),
//             NOTIFY=&SYSUID,USER=&SYSUID
//JOBLIB    DD DSNAME=IMX.V1R3M0.SELBMOD0,    <= (IF NECESSARY) CHANGE
//             DISP=(SHR,KEEP)
//*
//*         DD DSNAME=#RUNLIB1,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*         DD DSNAME=#RUNLIB2,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*
//IMXCDJ  PROC UNIT=SYSDA,                     <= (IF NECESSARY) CHANGE
//*            VOLSER=#VOLSER,                 <================ CHANGE
//             ANAPFIX=ITSORS3,                <= (IF NECESSARY) CHANGE
//             SYSID=SYSITSO,
//             SUBID=SUBSITSO,
//*
//             IMXPFIX='IMX.V1R3M0',           <= (IF NECESSARY) CHANGE
//             ELBPFIX='IMX.V1R3M0',           <= (IF NECESSARY) CHANGE
//*
//             JCLSRC='ITSORS2.SAMPAPPL.JCL',   <= CHANGE
//*
//     UI=TRK, INJCL01.............................................
//*
//* <QUANTITY> INJCL01       IJ1=PRIMARY  IJ2=SECONDARY  <- SEQUENTIAL
//             IJ1=0001,
//             IJ2=0001
//*
//*IJ@DEL  EXEC PGM=IEFBR14
//*INJCL01   DD DSNAME=&ANAPFIX..&SYSID..&SUBID..INJCL01,
//*             UNIT=&UNIT,
//*             VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
//IJ@ALC  EXEC PGM=IEFBR14
//INJCL01   DD DSNAME=&ANAPFIX..&SYSID..&SUBID..INJCL01,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB),
//             SPACE=(TRK,(0001,0001,0000)),
//             DISP=(NEW,KEEP)
//IJ@UCT  EXEC PGM=IEFBR14
//INJCL01   DD DSNAME=&ANAPFIX..&SYSID..&SUBID..INJCL01,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,UNCATLG)
//IJ@DEL  EXEC PGM=IEFBR14
//INJCL01   DD DSNAME=&ANAPFIX..&SYSID..&SUBID..INJCL01,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,DELETE)
//*
```

Figure 18 (Part 1 of 2). Sample JCL for JCL Analysis Input Card Creation

```
//IJ@CRE  EXEC PGM=IKJEFT01
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//ELBBLLIB  DD DSNAME=&IMXPFIX..SIMXBLK,
//             DISP=(SHR,KEEP)
//SYSENV    DD DSNAME=&IMXPFIX..SIMXENV(SYS0),
//             DISP=(SHR,KEEP)
//SYSPRM    DD DSNAME=&IMXPFIX..SIMXPRM,
//             DISP=(SHR,KEEP)
//INCARD    DD DSNAME=&IMXPFIX..SIMXCARD(IMXAF#2J),
//             DISP=(SHR,KEEP)
//INPDS     DD DSNAME=&JCLSRC,
//             DISP=(SHR,KEEP)
//INCRD01   DD DSNAME=&ANAPFIX..&SYSID..&SUBID..INJCL01,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB),
//             SPACE=(&UI,(&IJ1,&IJ2,0000)),
//             DISP=(NEW,CATLG)
//INCRD02   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//INCRD03   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//INCRD04   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//INCRD05   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//INCRD06   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//INCRD07   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//INCRD08   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//INCRD09   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//INCRD10   DD DUMMY,
//             DCB=(DSORG=PS,LRECL=00259,BLKSIZE=32760,RECFM=VB)
//*
//        PEND
//*
//IMXCDJ1 EXEC IMXCDJ
//*
//*
//*
//IJ@CRE.SYSTSIN DD *
  IMXRR01      5000
/*
//
```

Figure 18 (Part 2 of 2). Sample JCL for JCL Analysis Input Card Creation

3.5.1.3 IMX3ANJ: JCL Analysis

This function analyzes JCL source members for each subsystem and creates intermediate data in the analysis buffer for subsequent processing by the storing process performed in job IMX4SBJ. The standard utility analysis function is automatically invoked during JCL analysis, in order to analyze utility control cards and track the data flow from a particular column of an input record to a particular column of an output record. The standard utility analysis function automatically recognizes the utility steps of the following utilities during JCL analysis:

- SORT
- ICEMAN
- DFSORT
- IEBCOPY
- IEBGENER
- IDCAMS

Note

The following restrictions apply to JCL analysis:

- The job name must be the same as the source member name.
- The standard utility analysis function assumes that utility control cards are provided in an in-stream data set. External files containing control card statements are not supported by the MA2000 JCL analysis function.
- With the exception of the standard utility control cards mentioned above, MA2000 JCL does not support the use of in-stream data sets.

Input Parameters: Table 20 presents the parameters that required modification in order for us to successfully execute the JCL analysis function against the source members in our sample application. All other parameters retained their original values.

Table 20 (Page 1 of 2). IMX3ANJ Input Parameters Requiring Modification for the Sample Application

Parameter	Description	Initial Value	Modified Value
RUNLIB1	IBM C/370 (SEDCLINK) JOBLIB library name	#RUNLIB1	We deleted this JOBLIB DD statement because our system is running under Language Environment.
RUNLIB2	IBM SIBMLINK JOBLIB library name	#RUNLIB2	We deleted this JOBLIB DD statement because our system is running under Language Environment.
UNIT	Unit name of DASD	#DISK	SYSDA
VOLSER	Volume on which data sets are to be allocated	#VOLSER	None
ANAPFIX	Primary analysis data set name qualifier	#ANAHLQ	ITSORS3
SYSID	System ID	#SYSID	SYSITSO
SUBID	Subsystem ID	#SUBID1	SUBSITSO
IMXPFIX	High-level qualifiers of the MA2000 product data sets	#IMXHLQ	IMX.V1R3M0
ELBPFIX	High-level qualifiers of the Prolog product data sets	#ELBHLQ	IMX.V1R3M0
JCLSRC	JCL source library	#JCLSRC1	ITSORS2.SAMPAPPL.COBOL

Table 20 (Page 2 of 2). IMX3ANJ Input Parameters Requiring Modification for the Sample Application

Parameter	Description	Initial Value	Modified Value
JCLPRC	Cataloged procedure library	#JCLPRC1	ITSORS2.SAMPAPPL. JCLPRC
SYSPRM	Nonstandard system parameter library	#SYSPRM	ITSORS3.MA2000. SYSPRM
UC	Unit of space allocation for JI.B/JX.B/JMSG	CYL	TRK
JI1,JX1,JM1	Primary allocation quantity for JCL analysis intermediate data buffer, JCL analysis transaction, and JCL analysis message data sets	> 1	= 1
JI2,JX2, JM2	Secondary allocation quantity for JCL analysis intermediate data buffer, JCL analysis transaction, and JCL analysis message data sets	> 1	= 1
JIM	Allocation quantity for directory of JCL intermediate data buffer data set	200	1

Adaptation: We found it necessary to adapt the JCL supplied for job IMX3ANJ by eliminating all the preliminary deletions performed on data sets subsequently allocated during the job. Before the initial execution of IMX3ANJ, we removed the following introductory steps in order to avoid the JCL errors that otherwise would have resulted:

- //JI@DEL EXEC PGM=IEFBR14
- //JX@DEL EXEC PGM=IEFBR14
- //JM@DEL EXEC PGM=IEFBR14

Execution JCL: The JCL shown in Figure 19 on page 75 is a sample of the JCL that we executed in order to analyze the JCL source belonging to our sample application.

```
//ITSORS3B JOB ,
//             MSGLEVEL=(1,1),MSGCLASS=H,TIME=(,59),
//             NOTIFY=&SYSUID,USER=&SYSUID
//JOBLIB    DD DSNAME=IMX.V1R3M0.SELBMODO,    <= (IF NECESSARY) CHANGE
//             DISP=(SHR,KEEP)
//*
//*         DD DSNAME=#RUNLIB1,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*         DD DSNAME=#RUNLIB2,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*
//          DD DSNAME=IMX.V1R3M0.SIMXLOAD,    <= (IF NECESSARY) CHANGE
//             DISP=(SHR,KEEP)
//*
//IMXANJ  PROC UNIT=SYSDA,                     <= (IF NECESSARY) CHANGE
//*            VOLSER=#VOLSER,                 <================ CHANGE
//             ANAPFIX=ITSORS3,                <= (IF NECESSARY) CHANGE
//             SYSID=SYSITSO,
//             SUBID=SUBSITSO,
//*
//             IMXPFIX='IMX.V1R3M0',           <= (IF NECESSARY) CHANGE
//             ELBPFIX='IMX.V1R3M0',           <= (IF NECESSARY) CHANGE
//*
//             JCLSRC='ITSORS2.SAMPAPPL.JCL',   <= CHANGE
//*
//             JCLPRC='ITSORS2.SAMPAPPL.JCLPRC',     <= CHANGE
//*
//             SYSPRM='ITSORS3.MA2000.SYSPRM', <=(IF NECESSARY) CHANGE
//*
//     UC=TRK, JI.B/JX.B/JMSG.................<= (IF NECESSARY) CHANGE
//*
//* <QUANTITY> JI.B          JI1=PRIMARY  JI2=SECONDARY  JIM=DIRECTORY
//             JI1=0001,
//             JI2=0001,
//             JIM=0001,
//*
//* <QUANTITY> JX.B          JX1=PRIMARY  JX2=SECONDARY  <- SEQUENTIAL
//             JX1=0001,
//             JX2=0001,
//*
//* <QUANTITY> JMSG          JM1=PRIMARY  JM2=SECONDARY  <- SEQUENTIAL
//             JM1=0001,
//             JM2=0001
//*
//*JI@DEL   EXEC PGM=IEFBR14
//*JI@B       DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JI.B,
//*              UNIT=&UNIT,
//**             VOLUME=SER=&VOLSER,
//*              DISP=(OLD,DELETE)
//JI@ALC  EXEC PGM=IEFBR14
//JI@B      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JI.B,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PO,LRECL=04096,BLKSIZE=20480,RECFM=FB),
//             SPACE=(TRK,(0001,0000,0001)),
//             DISP=(NEW,KEEP)
//JI@UCT  EXEC PGM=IEFBR14
//JI@B      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JI.B,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,UNCATLG)
```

Figure 19 (Part 1 of 3). Sample JCL for JCL Analysis

```
//JI@DEL  EXEC PGM=IEFBR14
//JI@B      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JI.B,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,DELETE)
//*
//*JX@DEL   EXEC PGM=IEFBR14
//*JX@B       DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JX.B,
//*              UNIT=&UNIT,
//***            VOLUME=SER=&VOLSER,
//*              DISP=(OLD,DELETE)
//JX@ALC  EXEC PGM=IEFBR14
//JX@B      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JX.B,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PS,LRECL=00512,BLKSIZE=23476,RECFM=VB),
//             SPACE=(TRK,(0001,0000,0000)),
//             DISP=(NEW,KEEP)
//JX@UCT  EXEC PGM=IEFBR14
//JX@B      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JX.B,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,UNCATLG)
//JX@DEL  EXEC PGM=IEFBR14
//JX@B      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JX.B,
//             UNIT=&UNIT,
//**            VOLUME=SER=&VOLSER,
//             DISP=(OLD,DELETE)
//*
//*JM@DEL   EXEC PGM=IEFBR14
//*JMSG       DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JMSG,
//*              UNIT=&UNIT,
//**             VOLUME=SER=&VOLSER,
//*              DISP=(OLD,DELETE)
//JM@ALC  EXEC PGM=IEFBR14
//JMSG      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JMSG,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PS,LRECL=00512,BLKSIZE=23476,RECFM=VB),
//             SPACE=(TRK,(0001,0000,0000)),
//             DISP=(NEW,KEEP)
//JM@UCT  EXEC PGM=IEFBR14
//JMSG      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JMSG,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,UNCATLG)
//JM@DEL  EXEC PGM=IEFBR14
//JMSG      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JMSG,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,DELETE)
//*
```

Figure 19 (Part 2 of 3). Sample JCL for JCL Analysis

```
//JCLANA  EXEC PGM=IKJEFT01
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//ELBBLLIB  DD DSNAME=&IMXPFIX..SIMXBLK,
//             DISP=(SHR,KEEP)
//SYSENV    DD DSNAME=&IMXPFIX..SIMXENV(SYS0),
//             DISP=(SHR,KEEP)
//SYSPRM    DD DSNAME=&SYSPRM,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&IMXPFIX..SIMXPRM,
//             DISP=(SHR,KEEP)
//INCARD    DD DSNAME=&ANAPFIX..&SYSID..&SUBID..INJCL01,
//             DISP=(SHR,KEEP)
//JCL       DD DSNAME=&JCLSRC,
//             DISP=(SHR,KEEP)
//PROC      DD DSNAME=&JCLPRC,
//             DISP=(SHR,KEEP)
//JI        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JI.B,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PO,LRECL=04096,BLKSIZE=20480,RECFM=FB),
//             SPACE=(&UC,(&JI1,&JI2,&JIM)),
//             DISP=(NEW,CATLG)
//JIM       DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JI,
//             DISP=(SHR,KEEP)
//JX        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JX.B,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PS,LRECL=00512,BLKSIZE=23476,RECFM=VB),
//             SPACE=(&UC,(&JX1,&JX2,0000)),
//             DISP=(NEW,CATLG)
//MSG       DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JMSG,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PS,LRECL=00512,BLKSIZE=23476,RECFM=VB),
//             SPACE=(&UC,(&JM1,&JM2,0000)),
//             DISP=(NEW,CATLG)
//*
//        PEND
//*
//IMXANJ1 EXEC IMXANJ
//*
//*
//*
//JCLANA.SYSTSIN DD *
  IMXRR01
/*
//JCLANA.PROC    DD
//
//*         DD DSNAME=JCLPRC1A,              <= (IF NECESSARY)
//*            DISP=(SHR,KEEP)                   CONCATENATE DATA SET
//*         DD DSNAME=JCLPRC1B,              <= (IF NECESSARY)
//*            DISP=(SHR,KEEP)                   CONCATENATE DATA SET
//*         DD DSNAME=JCLPRC1C,              <= (IF NECESSARY)
//*            DISP=(SHR,KEEP)                   CONCATENATE DATA SET
//*
//
```

Figure 19 (Part 3 of 3). Sample JCL for JCL Analysis

3.5.2 Creating the JCL Retrieval Environment

A JCL-related retrieval environment is initially created for each individual subsystem assigned to the system being processed. In a subsequent step, the subsystem intermediate data resulting from this preparatory process is used to create JCL-related intermediate data at the system level. Intermediate data management functions include the generation of a copy of the original JCL source library and JCL-related dictionaries. The activities that make up the retrieval environment creation process are prerequisites for the invocation of the JCL-related retrieval functions available in the MA2000 on-line environment.

The JCL we used for the establishment of the retrieval environment for our application was taken from the SIMXLIB1 library, since only one retrieval view (= subsystem ID) of the sample was required. If, however, the system you are processing comprises more than one subsystem, use the members supplied in SIMXLIB2 or SIMXLIB3 as appropriate.

We performed the following three jobs to create the JCL-related retrieval environment for our sample application:

1. IMX4SBJ
2. IMX5DCJ
3. IMX7CKM

Job IMX7CKM is the same for both JCL and program processing. See 3.4.2.4, “IMX7CKM: Checking Unregistered Members” on page 68 for a further explanation of the Unregistered Members Checking job.

3.5.2.1 IMX4SBJ: Storing Subsystem JCL Intermediate Data

Job IMX4SBJ creates intermediate JCL-related data for a subsystem. During this process, data is copied from the buffer data sets created during the analysis process to subsystem intermediate data sets where it is stored in preparation for the system intermediate data creation process which follows. Any data that caused an error during the analysis phase is not stored in the subsystem intermediate data sets.

As this intermediate data is created at the subsystem level, the JCL to execute this process must be obtained from the sample JCL library appropriate to the number of subsystems in your concatenation sequence. You must specify the priority of the subsystems within a system by means of the input parameters provided. For an explanation of these parameters, see 3.1.3.2, “Defining Systems and Subsystems within the MA2000 Environment” on page 21.

Job IMX4SBJ also allows you to delete JCL source members which have erroneously registered in a subsystem. For details about eliminating members that have been mistakenly registered, see Chapter 4, ″Preparing to Retrieve Analysis Data″ in the manual *Maintenance 2000 Guide and Reference Version 1, Release 3*.

Input Parameters: Table 21 on page 79 shows the parameters we had to modify in order for us to successfully execute the subsystem intermediate data storage process for the JCL source members in our sample application. All other parameters retained their original values.

Table 21. IMX4SBJ Input Parameters Requiring Modification for the Sample Application

Parameter	Description	Initial Value	Modified Value
RUNLIB1	IBM C/370 (SEDCLINK) JOBLIB library name	#RUNLIB1	We deleted this JOBLIB DD statement because our system is running under Language Environment.
RUNLIB2	IBM SIBMLINK JOBLIB library name	#RUNLIB2	We deleted this JOBLIB DD statement because our system is running under Language Environment.
UNIT	Unit name of DASD	#DISK	SYSDA
VOLSER	Volume on which data sets are to be allocated	#VOLSER	None
ANAPFIX	Primary data set name qualifier	#ANAHLQ	ITSORS3
SYSID	System ID	#SYSID	SYSITSO
SUBID	Subsystem ID	#SUBID1	SUBSITSO
IMXPFIX	High-level qualifiers of the MA2000 product data sets	#IMXHLQ	IMX.V1R3M0
ELBPFIX	High-level qualifiers of the Prolog product data sets	#ELBHLQ	IMX.V1R3M0
JCLSRC	JCL source library	#JCLSRC1	ITSORS2.SAMPAPPL.JCL
UW	Unit of space allocation for buffer	CYL	TRK
JI1,JS1	Primary allocation quantity for JCL analysis intermediate data buffer and the JCL storage data buffer	30	15
JI2,JS2,	Secondary allocation quantity for JCL analysis intermediate data buffer and the JCL storage data buffer	30	1
JIM,JSM	Directory allocation for JCL analysis intermediate data buffer and JCL storage data buffer data sets	100	1
UD	Unit of space allocation for JCL-dictionary data set	CYL	TRK

Adaptation: We found it necessary to adapt the JCL supplied for job IMX4SBJ by eliminating all the preliminary deletions performed on data sets subsequently allocated during the job. Before the initial execution of IMX4SBJ, we removed the following preliminary step in order to avoid the JCL error that otherwise would have resulted:

- //DJ@DEL EXEC PGM=IEFBR14

Execution JCL: Figure 20 shows the JCL we executed in order to store the JCL-related subsystem intermediate data for our sample application JCL source.

```
//ITSORS3C JOB ,
//            MSGLEVEL=(1,1),MSGCLASS=H,TIME=(,59),
//            NOTIFY=&SYSUID,USER=&SYSUID
//JOBLIB    DD DSNAME=IMX.V1R3M0.SELBMODO,    <= (IF NECESSARY) CHANGE
//            DISP=(SHR,KEEP)
//*
//*         DD DSNAME=#RUNLIB1,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*         DD DSNAME=#RUNLIB2,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*
//          DD DSNAME=IMX.V1R3M0.SIMXLOAD,    <= (IF NECESSARY) CHANGE
//            DISP=(SHR,KEEP)
//*
//IMXSBJ  PROC UNIT=SYSDA,                     <= (IF NECESSARY) CHANGE
//*            VOLSER=#VOLSER,                 <================ CHANGE
//             ANAPFIX=ITSORS3,                <= (IF NECESSARY) CHANGE
//             SYSID=SYSITSO,
//             SUBID=SUBSITSO,
//*
//             IMXPFIX='IMX.V1R3M0',           <= (IF NECESSARY) CHANGE
//             ELBPFIX='IMX.V1R3M0',           <= (IF NECESSARY) CHANGE
//*
//             JCLSRC='ITSORS2.SAMPAPPL.JCL',  <= CHANGE
//*
//             JIVOLS=01,
//*
//     UD=TRK, DICJ.........................<= (IF NECESSARY) CHANGE
//*
//* <QUANTITY> DICJ          DJ1=PRIMARY  DJ2=SECONDARY  <- SEQUENTIAL
//             DJ1=0001,
//             DJ2=0001,
//*
//     UW=TRK, SpaceForJCLSRCWorkingDataSet...<= (IF NECESSARY) CHANGE
//*            SpaceFor<<JI>>WorkingDataSet.....................
//*
//* <QUANTITY> ********      JS1=PRIMARY  JS2=SECONDARY  JSM=DIRECTORY
//             JS1=0015,                      <= (IF NECESSARY) CHANGE
//             JS2=0001,                      <= (IF NECESSARY) CHANGE
//             JSM=0001,                      <= (IF NECESSARY) CHANGE
//*
//* <QUANTITY> ********      JI1=PRIMARY  JI2=SECONDARY  JIM=DIRECTORY
//             JI1=0015,                      <= (IF NECESSARY) CHANGE
//             JI2=0001,                      <= (IF NECESSARY) CHANGE
//             JIM=0001                       <= (IF NECESSARY) CHANGE
//*
```

Figure 20 (Part 1 of 7). Sample JCL for Subsystem Intermediate JCL Data Storage Process

```
//*DJ@DEL  EXEC PGM=IEFBR14
//*DICJ      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DICJ,
//*             UNIT=&UNIT,
//***           VOLUME=SER=&VOLSER,
//*             DISP=(OLD,DELETE)
//DJ@ALC  EXEC PGM=IEFBR14
//DICJ       DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DICJ,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PS,LRECL=00080,BLKSIZE=03200,RECFM=FB),
//             SPACE=(TRK,(0001,0001,0000)),
//             DISP=(NEW,KEEP)
//DJ@UCT  EXEC PGM=IEFBR14
//DICJ       DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DICJ,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,UNCATLG)
//DJ@DEL  EXEC PGM=IEFBR14
//DICJ       DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DICJ,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DISP=(OLD,DELETE)
//*
//JX@SRT  EXEC PGM=ICEMAN
//SYSIN     DD DUMMY
//SORTIN    DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JX.B,
//             DISP=(SHR,KEEP)
//SORTOUT   DD DSNAME=&&JX1B,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00512,BLKSIZE=23476,RECFM=VB),
//             SPACE=(CYL,(0010,0010,0000),RLSE),
//             DISP=(NEW,PASS)
//SORTWK01  DD UNIT=&UNIT,
//             SPACE=(CYL,(0010,0010,0000))
//SORTWK02  DD UNIT=&UNIT,
//             SPACE=(CYL,(0010,0010,0000))
//SORTWK03  DD UNIT=&UNIT,
//             SPACE=(CYL,(0010,0010,0000))
//SYSOUT    DD SYSOUT=*
//*
//JX@RAS  EXEC PGM=IKJEFT01,
//             COND=((0,NE,JX@SRT))
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//ELBBLLIB  DD DSNAME=&IMXPFIX..SIMXBLK,
//             DISP=(SHR,KEEP)
//SYSENV    DD DSNAME=&IMXPFIX..SIMXENV(SYS@),
//             DISP=(SHR,KEEP)
//SYSPRM    DD DSNAME=&IMXPFIX..SIMXPRM,
//             DISP=(SHR,KEEP)
```

Figure 20 (Part 2 of 7). Sample JCL for Subsystem Intermediate JCL Data Storage Process

```
//INCARD    DD DSNAME=&IMXPFIX..SIMXCARD(IMXAN#01),
//             DISP=(SHR,KEEP)
//DXBT1     DD DSNAME=&&JX1B,
//             DISP=(OLD,DELETE)
//DXBT2     DD DSNAME=&&JX2B,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00512,BLKSIZE=23476,RECFM=VB),
//             SPACE=(CYL,(0010,0010,0000),RLSE),
//             DISP=(NEW,PASS)
//DL        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JCLDL,
//             DISP=(SHR,KEEP)
//COPYI     DD DSNAME=&&CJ@I,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00080,BLKSIZE=03200,RECFM=FB),
//             SPACE=(CYL,(0010,0010,0000),RLSE),
//             DISP=(NEW,PASS)
//*
//JX@UPD  EXEC PGM=IKJEFT01,
//             COND=((0,NE,JX@SRT),(4,LT,JX@RAS))
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//ELBBLLIB  DD DSNAME=&IMXPFIX..SIMXBLK,
//             DISP=(SHR,KEEP)
//SYSENV    DD DSNAME=&IMXPFIX..SIMXENV(SYS@),
//             DISP=(SHR,KEEP)
//SYSPRM    DD DSNAME=&IMXPFIX..SIMXPRM,
//             DISP=(SHR,KEEP)
//INCARD    DD DSNAME=&IMXPFIX..SIMXCARD(IMXAN#02),
//             DISP=(SHR,KEEP)
//DXBT2     DD DSNAME=&&JX2B,
//             DISP=(OLD,DELETE)
//DX        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JX,
//             DISP=(SHR,KEEP)
//DL        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JCLDL,
//             DISP=(SHR,KEEP)
//DXT       DD DSNAME=&&JX3B,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00512,BLKSIZE=23476,RECFM=VB),
//             SPACE=(CYL,(0010,0010,0000),RLSE),
//             DISP=(NEW,PASS)
//DIC       DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DICJ,
//             UNIT=&UNIT,
//*            VOLUME=SER=&VOLSER,
//             DCB=(DSORG=PS,LRECL=00080,BLKSIZE=03200,RECFM=FB),
//             SPACE=(&UD,(&DJ1,&DJ2,0000)),
//             DISP=(NEW,CATLG)
//*
```

Figure 20 (Part 3 of 7). Sample JCL for Subsystem Intermediate JCL Data Storage Process

```
//HJ@UPD  EXEC PGM=IMXBNO1,
//             PARM='&JIVOLS.',
//             REGION=1024K,
//             COND=((0,NE,JX@SRT),(4,LT,JX@RAS))
//HASHIDD   DD DSNAME=&ANAPFIX..&SYSID..&SUBID..HASHJI,
//             DISP=(SHR,KEEP)
//COPYIDD   DD DSNAME=&&CJ@I,
//             DISP=(OLD,DELETE)
//HASHODD   DD DSNAME=&&HJ@B,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00080,BLKSIZE=03200,RECFM=FB),
//             SPACE=(CYL,(0010,0010,0000),RLSE),
//             DISP=(NEW,PASS)
//COPYODD   DD DSNAME=&&CJ@O,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00080,BLKSIZE=03200,RECFM=FB),
//             SPACE=(CYL,(0010,0010,0000),RLSE),
//             DISP=(NEW,PASS)
//DELIDD    DD DUMMY
//DELODD    DD DUMMY
//MMSGIDD   DD DUMMY
//SYSPRINT  DD SYSOUT=*
//*
//JITCPY  EXEC PGM=IEBCOPY,
//             COND=((0,NE,JX@SRT),(4,LT,JX@RAS),(0,NE,HJ@UPD))
//SYSIN     DD DSNAME=&&CJ@O,
//             DISP=(OLD,PASS)
//XIB       DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JI.B,
//             DISP=(SHR,KEEP)
//XIO1      DD DSNAME=&&JI@B,
//             UNIT=&UNIT,
//             SPACE=(&UW,(&JI1,&JI2,&JIM),RLSE),
//             DISP=(NEW,PASS)
//SYSPRINT  DD SYSOUT=*
//*
//JI@RAS  EXEC PGM=IKJEFT01,
//             COND=((0,NE,JX@SRT),(4,LT,JX@RAS),(0,NE,HJ@UPD),
//            (0,NE,JITCPY))
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//XIO1      DD DSNAME=&&JI@B,
//             DISP=(OLD,PASS)
//MCARD     DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JCLDL,
//             DISP=(SHR,KEEP)
//*
//JI@CPY  EXEC PGM=IEBCOPY,
//             COND=((0,NE,JX@SRT),(4,LT,JX@RAS),(0,NE,HJ@UPD),
//            (0,NE,JITCPY),(0,NE,JI@RAS))
//SYSIN     DD DUMMY
//XIO1      DD DSNAME=&&JI@B,
//             DISP=(OLD,DELETE)
//XI        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JI,
//             DISP=(SHR,KEEP)
//SYSPRINT  DD SYSOUT=*
//*
```

Figure 20 (Part 4 of 7). Sample JCL for Subsystem Intermediate JCL Data Storage Process

```
//JI@CPS  EXEC PGM=IEBCOPY,
//             COND=((0,NE,JX@SRT),(4,LT,JX@RAS),(0,NE,HJ@UPD),
//           (0,NE,JITCPY),(0,NE,JI@RAS),
//           (0,NE,JI@CPY))
//SYSIN     DD DUMMY
//XI        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JI,
//             DISP=(SHR,KEEP)
//SYSPRINT  DD SYSOUT=*
//*
//JCTCPY  EXEC PGM=IEBCOPY,
//             COND=((0,NE,JX@SRT),(4,LT,JX@RAS),(0,NE,HJ@UPD))
//SYSIN     DD DSNAME=&&CJ@0,
//             DISP=(OLD,DELETE)
//XIB       DD DSNAME=&JCLSRC,
//             DISP=(SHR,KEEP)
//XIO1      DD DSNAME=&&JC@B,
//             UNIT=&UNIT,
//             SPACE=(&UW,(&JS1,&JS2,&JSM),RLSE),
//             DISP=(NEW,PASS)
//SYSPRINT  DD SYSOUT=*
//*
//JC@RAS  EXEC PGM=IKJEFT01,
//             COND=((0,NE,JX@SRT),(4,LT,JX@RAS),(0,NE,HJ@UPD),
//           (0,NE,JCTCPY))
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//XIO1      DD DSNAME=&&JC@B,
//             DISP=(OLD,PASS)
//MCARD     DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JCLDL,
//             DISP=(SHR,KEEP)
//*
//JC@CPY  EXEC PGM=IEBCOPY,
//             COND=((0,NE,JX@SRT),(4,LT,JX@RAS),(0,NE,HJ@UPD),
//           (0,NE,JCTCPY),(0,NE,JC@RAS))
//SYSIN     DD DUMMY
//XIO1      DD DSNAME=&&JC@B,
//             DISP=(OLD,DELETE)
//XI        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JCL,
//             DISP=(SHR,KEEP)
//SYSPRINT  DD SYSOUT=*
//*
//JC@CPS  EXEC PGM=IEBCOPY,
//             COND=((0,NE,JX@SRT),(4,LT,JX@RAS),(0,NE,HJ@UPD),
//           (0,NE,JCTCPY),(0,NE,JC@RAS),
//           (0,NE,JC@CPY))
//SYSIN     DD DUMMY
//XI        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JCL,
//             DISP=(SHR,KEEP)
//SYSPRINT  DD SYSOUT=*
//*
//HJ@CPY  EXEC PGM=IEBGENER,
//             COND=((0,NE,JX@SRT),(4,LT,JX@RAS),(0,NE,HJ@UPD))
//SYSIN     DD DUMMY
//SYSUT1    DD DSNAME=&&HJ@B,
//             DISP=(OLD,DELETE)
//SYSUT2    DD DSNAME=&ANAPFIX..&SYSID..&SUBID..HASHJI,
//             DISP=(OLD,KEEP)
//SYSPRINT  DD SYSOUT=*
//*
```

Figure 20 (Part 5 of 7). Sample JCL for Subsystem Intermediate JCL Data Storage Process

```
//JX3SRT  EXEC PGM=ICEMAN,
//             COND=((0,NE,JX@SRT),(4,LT,JX@RAS),(0,NE,JX@UPD))
//SYSIN     DD DUMMY
//SORTIN    DD DSNAME=&&JX3B,
//             DISP=(OLD,DELETE)
//SORTOUT   DD DSNAME=&&JX@B,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00512,BLKSIZE=23476,RECFM=VB),
//             SPACE=(CYL,(0010,0010,0000),RLSE),
//             DISP=(NEW,PASS)
//SORTWK01  DD UNIT=&UNIT,
//             SPACE=(CYL,(0010,0010,0000))
//SORTWK02  DD UNIT=&UNIT,
//             SPACE=(CYL,(0010,0010,0000))
//SORTWK03  DD UNIT=&UNIT,
//             SPACE=(CYL,(0010,0010,0000))
//SYSOUT    DD SYSOUT=*
//*
//JX@CPY  EXEC PGM=IEBGENER,
//             COND=((0,NE,JX@SRT),(4,LT,JX@RAS),(0,NE,JX@UPD),
//            (0,NE,JX3SRT))
//SYSIN     DD DUMMY
//SYSUT1    DD DSNAME=&&JX@B,
//             DISP=(OLD,DELETE)
//SYSUT2    DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JX,
//             DISP=(SHR,KEEP)
//SYSPRINT  DD SYSOUT=*
//*
//        PEND
//*
//IMXSBJ1 EXEC IMXSBJ
//*
//JX@SRT.SYSIN   DD *
  SORT         FIELDS=(11,8,CH,A),EQUALS
/*
//JX@RAS.SYSTSIN DD *
  IMXRR01
/*
//JX@UPD.SYSTSIN DD *
  IMXRR01
/*
//*
//JI@RAS.SYSTSIN DD *
  IMXAN01
/*
//*
//JI@CPY.SYSIN   DD *
  COPY         INDD=((XI01,R)),OUTDD=XI
/*
//JI@CPS.SYSIN   DD *
  COPY         INDD=XI,OUTDD=XI
/*
//*
```

Figure 20 (Part 6 of 7). Sample JCL for Subsystem Intermediate JCL Data Storage Process

```
//JC@RAS.SYSTSIN DD *
  IMXANO1
/*
//*
//JC@CPY.SYSIN   DD *
  COPY         INDD=((XIO1,R)),OUTDD=XI
/*
//JC@CPS.SYSIN   DD *
  COPY         INDD=XI,OUTDD=XI
/*
//JX3SRT.SYSIN   DD *
  SORT         FIELDS=(11,8,CH,A),EQUALS
/*
//
```

Figure 20 (Part 7 of 7). Sample JCL for Subsystem Intermediate JCL Data Storage Process

3.5.2.2 IMX5DCJ: JCL-Related Dictionary Creation

Job IMX5DCJ creates the JCL-related dictionaries, which constitute a portion of the system intermediate data. As the JCL-related dictionary creation process takes place at the system level rather than at the individual subsystem level, job IMX5DCJ must only be executed once per system. After obtaining the JCL for this job from the sample JCL library appropriate to the number of subsystems in the subsystem concatenation sequence, you must specify the priority of the subsystems being processed by means of the input parameters provided. Dictionaries are then created in accordance with the retrieval views which you have defined by means of the subsystem ID input parameters. For an explanation of how to define the priority of subsystems in the subsystem concatenation sequence, see 3.1.3.2, “Defining Systems and Subsystems within the MA2000 Environment” on page 21.

The following types of dictionaries are created or updated by this process:

- JCL-related dictionary
- Program-related dictionary
- Input-output file dictionary
- Input-output data set name dictionary

Input Parameters: Table 22 on page 87 illustrates the parameters which required modification in order for us to successfully create the JCL-related dictionaries for our sample application JCL source. All other parameters remained unchanged.

Table 22. IMX5DCJ Input Parameters Requiring Modification for the Sample Application

Parameter	Description	Initial Value	Modified Value
RUNLIB1	IBM C/370 (SEDCLINK) JOBLIB library name	#RUNLIB1	We deleted this JOBLIB DD statement because our system is running under Language Environment.
RUNLIB2	IBM SIBMLINK JOBLIB library name	#RUNLIB2	We deleted this JOBLIB DD statement because our system is running under Language Environment.
UNIT	Unit name of DASD	#DISK	SYSDA

Table 22. IMX5DCJ Input Parameters Requiring Modification for the Sample Application

Parameter	Description	Initial Value	Modified Value
ANAPFIX	Primary data set name qualifier	#ANAHLQ	ITSORS3
SYSID	System ID	#SYSID	SYSITSO
SUBID	Subsystem ID	#SUBID1	SUBSITSO
IMXPFIX	High level qualifiers of the MA2000 product data sets	#IMXHLQ	IMX.V1R3M0
ELBPFIX	High level qualifiers of the Prolog product data sets	#ELBHLQ	IMX.V1R3M0

Adaptation: We did not find it necessary to adapt the JCL supplied for job IMX5DCJ, as no initial delete operations were performed on data sets at the outset of the job.

Execution JCL: The JCL shown in Figure 21 on page 88 is a sample of the JCL we executed in order to create the JCL-related dictionaries for our sample application.

```
//ITSORS3D JOB ,
//            MSGLEVEL=(1,1),MSGCLASS=H,TIME=(,59),
//            NOTIFY=&SYSUID,USER=&SYSUID
//JOBLIB    DD DSNAME=IMX.V1R3M0.SELBMODO,    <= (IF NECESSARY) CHANGE
//            DISP=(SHR,KEEP)
//*
//*         DD DSNAME=#RUNLIB1,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*         DD DSNAME=#RUNLIB2,               <= (IF NECESSARY) CHANGE
//*            DISP=(SHR,KEEP)
//*
//          DD DSNAME=IMX.V1R3M0.SIMXLOAD,    <= (IF NECESSARY) CHANGE
//            DISP=(SHR,KEEP)
//*
//IMXDCJ  PROC UNIT=SYSDA,                    <= (IF NECESSARY) CHANGE
//*
//            ANAPFIX=ITSORS3,                <= (IF NECESSARY) CHANGE
//            SYSID=SYSITSO,
//            SUBID=SUBSITSO,
//*
//*
//            IMXPFIX='IMX.V1R3M0',           <= (IF NECESSARY) CHANGE
//            ELBPFIX='IMX.V1R3M0'            <= (IF NECESSARY) CHANGE
//*
```

Figure 21 (Part 1 of 4). Sample JCL for JCL-Related Dictionary Creation

```
//DJ@WRT  EXEC PGM=IKJEFT01
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//ELBBLLIB  DD DSNAME=&IMXPFIX..SIMXBLK,
//             DISP=(SHR,KEEP)
//SYSENV    DD DSNAME=&IMXPFIX..SIMXENV(SYS0),
//             DISP=(SHR,KEEP)
//SYSPRM    DD DSNAME=&IMXPFIX..SIMXPRM,
//             DISP=(SHR,KEEP)
//INCARD    DD DSNAME=&IMXPFIX..SIMXCARD(IMXAM#1J),
//             DISP=(SHR,KEEP)
//XB        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JX,
//             DISP=(SHR,KEEP)
//DL        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JCLDL,
//             DISP=(SHR,KEEP)
//RD        DD DSNAME=&ANAPFIX..&SYSID..&SUBID..XJ,
//             DISP=(SHR,KEEP)
//RD1       DD DSNAME=&ANAPFIX..&SYSID..&SUBID..XJ,
//             DISP=(SHR,KEEP)
//RD2       DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DUMMYPO,
//             DISP=(SHR,KEEP)
//MWKO2     DD DSNAME=&&DJ2W,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00102,BLKSIZE=10200,RECFM=FB),
//             SPACE=(CYL,(0040,0040,0000),RLSE),
//             DISP=(NEW,PASS)
//*
//DJ@SRT  EXEC PGM=ICEMAN,
//             COND=((0,NE,DJ@WRT))
//SYSIN     DD DUMMY
//SORTIN    DD DSNAME=&&DJ2W,
//             DISP=(OLD,DELETE)
//SORTOUT   DD DSNAME=&&DJ@W,
//             UNIT=&UNIT,
//             DCB=(DSORG=PS,LRECL=00102,BLKSIZE=10200,RECFM=FB),
//             SPACE=(CYL,(0040,0040,0000),RLSE),
//             DISP=(NEW,PASS)
//SORTWKO1  DD UNIT=&UNIT,
//             SPACE=(CYL,(0060,0060,0000))
//SORTWKO2  DD UNIT=&UNIT,
//             SPACE=(CYL,(0060,0060,0000))
//SORTWKO3  DD UNIT=&UNIT,
//             SPACE=(CYL,(0060,0060,0000))
//SYSOUT    DD SYSOUT=*
//*
```

Figure 21 (Part 2 of 4). Sample JCL for JCL-Related Dictionary Creation

```
//XP@UPD  EXEC PGM=IKJEFT01,
//             COND=((0,NE,DJ@WRT),
//           (0,NE,DJ@SRT))
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//ELBBLLIB  DD DSNAME=&IMXPFIX..SIMXBLK,
//             DISP=(SHR,KEEP)
//SYSENV    DD DSNAME=&IMXPFIX..SIMXENV(SYS@),
//             DISP=(SHR,KEEP)
//SYSPRM    DD DSNAME=&IMXPFIX..SIMXPRM,
//             DISP=(SHR,KEEP)
//INCARD    DD DSNAME=&IMXPFIX..SIMXCARD(IMXAM#JX),
//             DISP=(SHR,KEEP)
//DICTIN    DD DSNAME=&&DJ@W,
//             DISP=(OLD,PASS)
//MDICTRD1  DD DSNAME=&ANAPFIX..&SYSID..&SUBID..XP,
//             DISP=(SHR,KEEP)
//MDICTRD2  DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DUMMYPO,
//             DISP=(SHR,KEEP)
//MDICTWT   DD DSNAME=&ANAPFIX..&SYSID..&SUBID..XP,
//             DISP=(SHR,KEEP)
//*
//FI@UPD  EXEC PGM=IKJEFT01,
//             COND=((0,NE,DJ@WRT),
//           (0,NE,DJ@SRT))
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//ELBBLLIB  DD DSNAME=&IMXPFIX..SIMXBLK,
//             DISP=(SHR,KEEP)
//SYSENV    DD DSNAME=&IMXPFIX..SIMXENV(SYS@),
//             DISP=(SHR,KEEP)
//SYSPRM    DD DSNAME=&IMXPFIX..SIMXPRM,
//             DISP=(SHR,KEEP)
//INCARD    DD DSNAME=&IMXPFIX..SIMXCARD(IMXAM#FJ),
//             DISP=(SHR,KEEP)
//DICTIN    DD DSNAME=&&DJ@W,
//             DISP=(OLD,PASS)
//MDICTRD1  DD DSNAME=&ANAPFIX..&SYSID..&SUBID..FI,
//             DISP=(SHR,KEEP)
//MDICTRD2  DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DUMMYPO,
//             DISP=(SHR,KEEP)
//MDICTWT   DD DSNAME=&ANAPFIX..&SYSID..&SUBID..FI,
//             DISP=(SHR,KEEP)
//*
```

Figure 21 (Part 3 of 4). Sample JCL for JCL-Related Dictionary Creation

```
//DS@UPD  EXEC PGM=IKJEFT01,
//             COND=((0,NE,DJ@WRT),
//           (0,NE,DJ@SRT))
//SYSTSIN   DD DUMMY
//SYSTSPRT  DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSEXEC   DD DSNAME=&IMXPFIX..SIMXEXEC,
//             DISP=(SHR,KEEP)
//          DD DSNAME=&ELBPFIX..SELBEXEC,
//             DISP=(SHR,KEEP)
//ELBWSLIB  DD DSNAME=&ELBPFIX..SELBWS,
//             DISP=(SHR,KEEP)
//ELBBLLIB  DD DSNAME=&IMXPFIX..SIMXBLK,
//             DISP=(SHR,KEEP)
//SYSENV    DD DSNAME=&IMXPFIX..SIMXENV(SYS@),
//             DISP=(SHR,KEEP)
//SYSPRM    DD DSNAME=&IMXPFIX..SIMXPRM,
//             DISP=(SHR,KEEP)
//INCARD    DD DSNAME=&IMXPFIX..SIMXCARD(IMXAM#DS),
//             DISP=(SHR,KEEP)
//DICTIN    DD DSNAME=&&DJ@W,
//             DISP=(OLD,DELETE)
//MDICTRD1  DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DS,
//             DISP=(SHR,KEEP)
//MDICTRD2  DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DUMMYPO,
//             DISP=(SHR,KEEP)
//MDICTWT   DD DSNAME=&ANAPFIX..&SYSID..&SUBID..DS,
//             DISP=(SHR,KEEP)
//*
//JI@DEL  EXEC PGM=IEFBR14,
//             COND=((0,NE))
//JI@B      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JI.B,
//             DISP=(SHR,DELETE)
//*
//JX@DEL  EXEC PGM=IEFBR14,
//             COND=((0,NE))
//JX@B      DD DSNAME=&ANAPFIX..&SYSID..&SUBID..JX.B,
//             DISP=(SHR,DELETE)
//*
//        PEND
//*
//IMXDCJ  EXEC IMXDCJ
//*
//DJ@WRT.SYSTSIN DD *
  IMXRR01
/*
//*
//DJ@SRT.SYSIN   DD *
  SORT         FIELDS=(3,4,CH,A,18,16,CH,A)
/*
//*
//XP@UPD.SYSTSIN DD *
  IMXRR01
/*
//FI@UPD.SYSTSIN DD *
  IMXRR01
/*
//DS@UPD.SYSTSIN DD *
  IMXRR01
/*
//
```

Figure 21 (Part 4 of 4). Sample JCL for JCL-Related Dictionary Creation

3.6 Establishing the MA2000 On-line Environment

This section details the tasks necessary to integrate MA2000 into your TSO/E environment so that its batch analysis and retrieval processes can be invoked and their results displayed on-line. This process includes the following activities:

- The addition of MA2000 definitions to a TSO command procedure (CLIST), which is used to invoke the on-line ISPF portion of MA2000
- The optional adjustment of the values of other TSO variables that govern the session in which MA2000 is invoked

Note

The establishment of the retrieval environment is a prerequisite for on-line invocation of MA2000. For a description of the analysis and retrieval creation processes, see 3.3, “Overview of Analysis and Retrieval Environment Creation” on page 35.

3.6.1 On-Line Libraries for Maintenance 2000

Table 23 illustrates the libraries required for the on-line invocation of MA2000. The library names represent default values supplied during the MA2000 installation process and will undoubtedly require modification in accordance with your own system environment. Please consult your system programmer if necessary to obtain the values specific to your installation.

Table 23 (Page 1 of 2). MA2000 On-line Libraries

DD Name	Data Set Name	Description
STEPLIB	IMX.V1R3M0.SIMXLOAD	MA2000 load library
STEPLIB	IMX.V1R3M0.SELBMOD0	Prolog load library
STEPLIB	CEEV1R50.SCEERUN or EDC.V2R1M0.SEDCLINK and PLI.V2R3M0.PLILINK and PLI.V2R3M0.SIBMLINK	Language environment or C/370 run-time library and PL/I unique run-time library and PL/I common run-time library
ISPMLIB	IMX.V1R3M0.SIMXMENU	MA2000 ISPF message library
ISPPLIB	IMX.V1R3M0.SIMXPENU	MA2000 ISPF panel library
ISPSLIB	IMX.V1R3M0.SIMXSLIB	MA2000 ISPF skeleton library
SYSEXEC	IMX.V1R3M0.SIMXEXEC IMX.V1R3M0.SELBEXEC IMX.V1R3M0.SIMXENV	MA2000 command library Prolog command library MA2000 environment parameter library
ELBWSLIB	IMX.V1R3M0.SELBWS	Prolog workspace library
ELBBLLIB	IMX.V1R3M0.SIMXBLK	MA2000 blockspace library

3.6.2 TSO Command Procedure for MA2000 Invocation

The TSO commands used to allocate and concatenate the libraries listed in Table 23 must be placed in a TSO command procedure file. Whether or not you decide to include these commands in your default TSO logon procedure as we did depends on the conventions prevalent in your own installation.

If you wish to incorporate the commands necessary to establish the MA2000 on-line environment in your default TSO logon procedure, the basic procedure is as follows:

1. Add the commands to the TSO command procedure executed when you log on to TSO. In our case, a procedure named *DEFAULT* located in data set #USERID.CLIST.CLIST is executed at logon time. Figure 22 on page 92 illustrates this procedure before the addition of the MA2000 definitions.

```
CONTROL NOMSG NOFLUSH NOLIST
SE '***** STARTING DEFAULT CLIST FOR' USER(*)
CONCATD F(SYSEXEC) DA('ITSORS3.CLIST.CLIST') SHR
```

Figure 22. Sample TSO Logon CLIST Before Inclusion of MA2000 Libraries

Figure 23 shows an example of the logon CLIST used in our sample environment after the TSO commands to allocate and concatenate were added.

```
CONTROL NOMSG NOFLUSH NOLIST
SE '***** STARTING DEFAULT CLIST FOR' USER(*)
ALLOC F(LOADLIB) DA('IMX.V1R3M0.SELBMOD0' -
                    'CEEV1R50.SCEERUN'    -
                    'IMX.V1R3M0.SIMXLOAD') SHR REUSE
STEPLIB SET(LOADLIB)
CONCATD F(ELBWSLIB) DA('IMX.V1R3M0.SELBWS') SHR
CONCATD F(ELBBLLIB) DA('IMX.V1R3M0.SIMXBLK') SHR
CONCATD F(SYSEXEC) DA('IMX.V1R3M0.SIMXEXEC') SHR
CONCATD F(SYSEXEC) DA('IMX.V1R3M0.SELBEXEC') SHR
CONCATD F(SYSEXEC) DA('IMX.V1R3M0.SIMXENV') SHR
CONCATD F(SYSEXEC) DA('ITSORS3.CLIST.CLIST') SHR
CONCATD F(ISPPLIB) DA('IMX.V1R3M0.SIMXPENU') SHR
CONCATD F(ISPMLIB) DA('IMX.V1R3M0.SIMXMENU') SHR
CONCATD F(ISPSLIB) DA('IMX.V1R3M0.SIMXSLIB') SHR
```

Figure 23. Sample TSO Logon CLIST After Inclusion of MA2000 Libraries

2. Specify the name of this procedure as the procedure to be used at logon time in accordance with the standards defined at your installation.
3. Invoke MA2000 from within your ISPF environment by entering the TSO command “MA2000” on any command line, as Figure 24 on page 93 illustrates.

```
  Menu  Utilities  Compilers  Options  Status  Help
--------------------------------------------------------------------------------
                           ISPF Primary Option Menu
Option ===> tso ma2000

0  Settings      Terminal and user parameters           User ID . : ITSORS3
1  View          Display source data or listings        Time. . . : 16:44
2  Edit          Create or change source data           Terminal. : 3278
3  Utilities     Perform utility functions              Screen. . : 1
4  Foreground    Interactive language processing        Language. : ENGLISH
5  Batch         Submit job for language processing     Appl ID . : ISR
6  Command       Enter TSO or Workstation commands      TSO logon : TSOUSER
7  Dialog Test   Perform dialog testing                 TSO prefix: ITSORS3
10 SCLM          SW Configuration Library Manager       System ID : STLMVS1
11 Workplace     ISPF Object/Action Workplace           MVS acct. : ITSORS3,
                                                        Release . : ISPF 4.5
DB Database      Interactive Database Functions
G  STL/AdStaR    Santa Teresa/San Jose Local Selections
D  Dept          Department Panel                       Name ===> DPTPRI
U  User          User Panel                             Name ===> USRPRI
C  CLEAR         CLEAR Facility Selections
N  NEWS          MVS/STL News and Information

Enter X to Terminate using log/list defaults
```

Figure 24. Invoking MA2000 from within ISPF

Note

Use this command to invoke MA2000 from within your ISPF environment, regardless of whether you perform the allocations and concatenations required for MA2000 initially in your TSO logon CLIST or exeute them using an independent procedure.

3.6.3 TSO Session Variables

We found it more comfortable to increase the region size allocated to our TSO session to 64 MB. In our installation, we could accomplish this by supplying a new value directly on our TSO logon panel, as Figure 25 on page 94 shows.

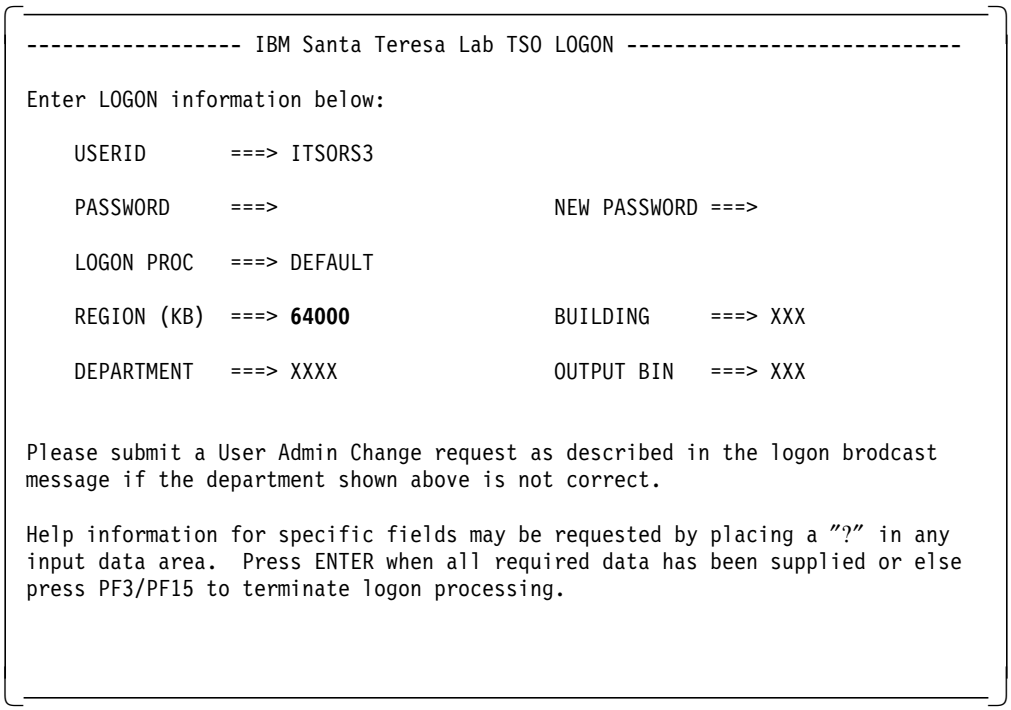


Figure 25. Changing TSO Session Variables Directly

Chapter 4. Maintenance 2000 On-line Interactive Tasks

This chapter describes the tasks we performed in order to find all the program items in our entire application (MA2000 system) that might contain a date. We created this information as a list (report) and in the format required by CCCA as input for its DATE FORMAT conversion option. The report can be used as information, based on which a manual fix can be applied to the programs. The Date Identification File is used by CCCA to apply an automated fix to the programs based on the solution provided by the Millennium Language Extensions.

The tasks include:

- Create a starting-point file using the search option or the specific starting-point file creation option.
- Run the Year 2000 analysis.
- Process the analysis results.

4.1 Overview

In 2.2.2, “MA2000 Retrieval Functions” on page 6 and 2.2.3, “MA2000 Retrieval Result Management Functions” on page 11, we described the basic on-line functions available from the MA2000 main menu (see Figure 26 on page 96). You have to select some of these options and execute the corresponding functions in order to produce information about date usage in your applications. You can use different options to create the same results. For example, to create the starting-point file used as input for the multiple impact analysis or the Year 2000 analysis, you can:

- Use the MA2000 option **A (Starting point file creation)**.
- Use the MA2000 option **2 (Search)**.
- Write it manually, using any editor.

The way you might want to use to achieve the results you are looking for depends on how big your MA2000 system is and how well you know your applications and programs.

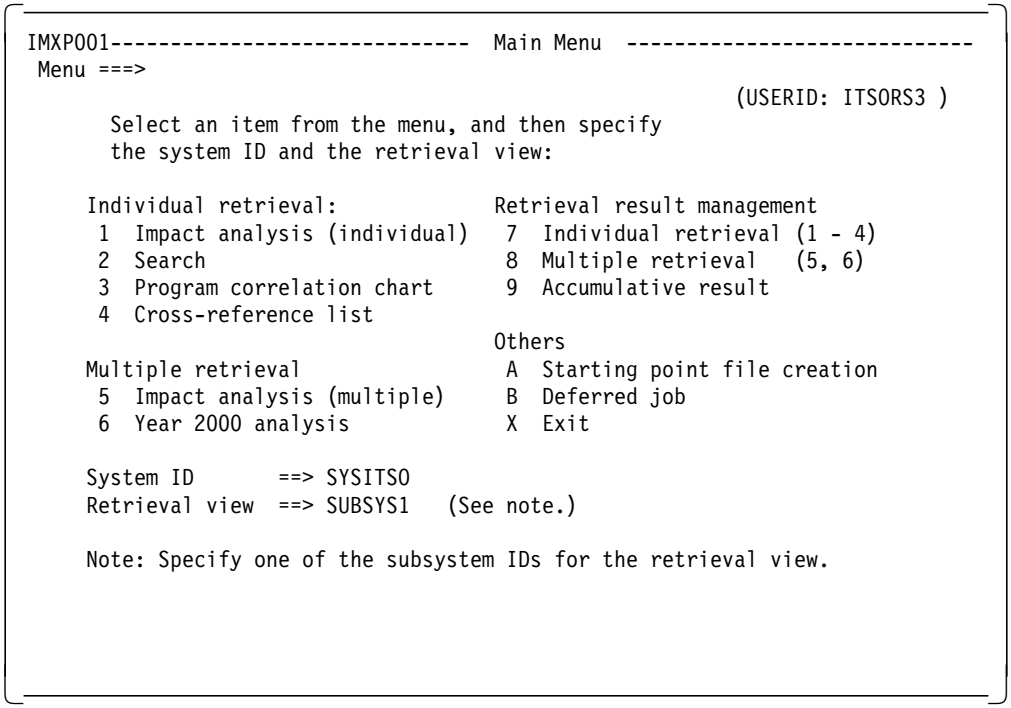


Figure 26. The MA2000 Main Menu - On-line Options

Figure 27 on page 97 shows the functions involved in the date usage analysis and their options. The basic functions that create the date usage information are the impact analysis functions:

- **Multiple Impact Analysis (Option 5)** — A starting-point file is needed for the multiple impact analysis. This starting-point file can contain one or more entries. An entry can be a
  - Program variable
  - Position in a record of a data set
  - Position in an IMS segment
  - Column in a DB2 table
  - Position in a record of a CICS file

  You can use Option 2 or Option A to generate the starting-point file.

  You can execute a cause analysis, influence analysis, or equivalency analysis.
- **Year 2000 Analysis (Option 6)** — The Year 2000 analysis function performs a multiple impact equivalency analysis. In addition to the impact analysis of the starting-points in the starting-point file, the Year 2000 analysis performs an impact analysis of program variables that would have been retrieved by the date function search function.
- **Individual Impact Analysis (Option 1)** — One starting-point is needed for the individual impact analysis. A starting-point can be a
  - Program variable
  - Position in a record of a data set
  - Position in an IMS segment
  - Column in a DB2 table
  - Position in a record of a CICS file

  You have to specify the starting-point directly when you execute Option 1.

You can execute a cause analysis, influence analysis, equivalency analysis, or Year 2000 analysis. The individual Year 2000 analysis is a multiple equivalency analysis. In addition to the impact analysis of the starting-point, the individual Year 2000 analysis performs an impact analysis of program variables that would have been retrieved by the date function search function.

The **Search (Option 2)** and **Starting Point File Creation (Option A)** options assist in developing the starting-point files required by the multiple impact and Year 2000 analysis functions.

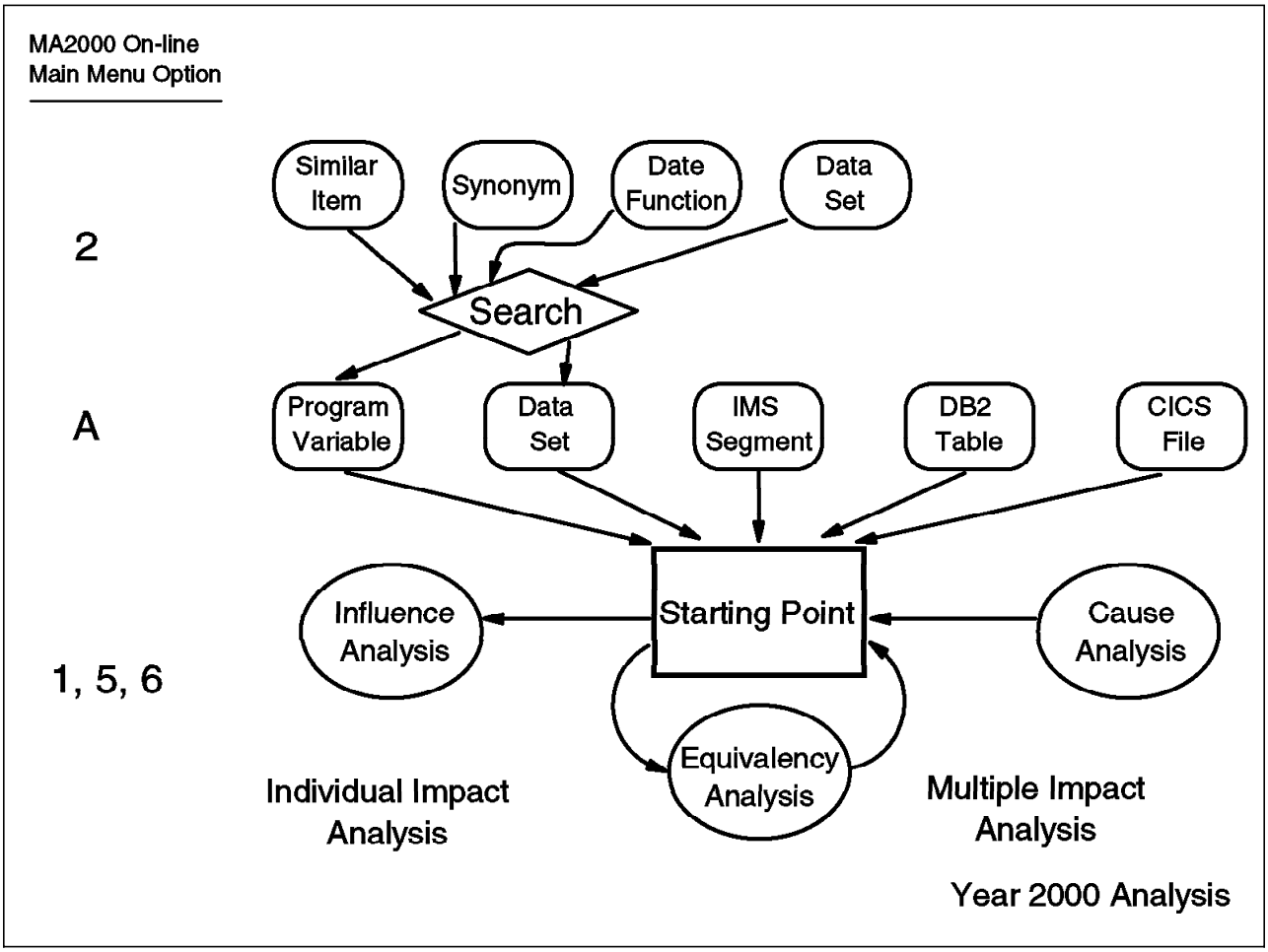


Figure 27. Overview of the Year 2000 Analysis On-line Functions

Table 24 on page 98 explains the differences between Option A (Starting point file creation) and Option 2 (Search) when used to create a starting-point file. Please note that you don't have to create a starting-point file when you use the search function. The result of the search can also be provided as an MA2000 summary and detailed report, which can be viewed through the individual retrieval option (Option 7 on the MA2000 Main Menu).

Table 24. MA2000 Starting-Point Creation Options				
Main menu option	***Starting-point item***	***Retrieval amount or search condition***	***Batch or foreground execution***	***Overwrite or append to starting-point file***
A	**Program variable**	One or more selected from a list	Foreground	Append (or create new)
	Data set	All of the system	Batch	Overwrite (or create new)
	IMS segment	All of the system	Foreground	Overwrite (or create new)
	DB2 table	All of the system	Batch	Overwrite (or create new)
	CICS file	All of the system	Foreground	Overwrite (or create new)
2	**Program variable**	• **Similar item**: pattern can be specified • **Synonym**: full variable name must be specified • **Date function**: all of the domain	Batch	Overwrite or append (or create new)
	Data set	Pattern can be specified	Batch	Overwrite or append (or create new)

There are two features which are unique to the search function (Option 2):

- You can specify if you want to have a potential date format added to the starting-point. After the creation of the starting-point file, you can review the generated date formats and correct them if necessary. When you run the Year 2000 analysis with the date identification file option, the date formats specified in the starting-point file are used in the date identification file.
- In addition to the conditions mentioned in Table 24, you can apply a retrieval filter by specifying a domain. The domain consists of a list of programs and jobs (JCL members) that you want to include or exclude in the search if you don′t want to perform the search against the entire system.

In the following sections, we explain in detail how to create the starting-point file and how to run the Year 2000 analysis.

4.2 Input and Output Data Set Allocations

The various on-line tasks require several input and output data sets. Some of these data sets need to be allocated beforehand, others are allocated by MA2000 when you run the corresponding function. Table 25 shows an overview of the input and output data sets by function. Please refer to Chapter 3, “Maintenance 2000 Preparation Tasks” on page 19 for detailed information about how to create the dictionary (retrieval environment) for your system.

Table 25 (Page 1 of 2). MA2000 Input and Output Data Sets for On-line Tasks			
Main Menu Option	**File**	**Input or Output**	**Mandatory or Optional**
All options	Dictionary	Input	Mandatory
A	Starting-Point File	Output	Mandatory
1	Domain File	Input	Optional
	Summary Report Detailed Report	Output	Mandatory

Table 25 (Page 2 of 2). MA2000 Input and Output Data Sets for On-line Tasks

Main Menu Option	File	Input or Output	Mandatory or Optional
2	Domain File	Input	Optional
	Condition File	Input	Using a file is optional; up to ten conditions can be entered on the search panel
	Starting-Point File	Output	Optional
	Summary Report Detailed Report	Output	Mandatory
5	Starting-Point File	Input	Mandatory
	Domain File	Input	Optional
	Summary Report Detailed Report	Output	Mandatory
6	Starting-Point File	Input	Mandatory
	Domain File	Input	Optional
	Date Identification File	Output	Optional
	Summary Report Detailed Report	Output	Mandatory

The files are as follows:

- **Report files** — The report files are always created by MA2000 as sequential files with the name `userid.SUMRY.#date.Ttime` for the summary reports and `userid.DETAL2.#date.Ttime` for the detailed reports. The record format is fixed block, and the logical record length is 133. You access these files through the retrieval result management on-line functions.
- **Starting-Point File** — You must allocate the starting-point file manually with any name. It can be a sequential file or a partitioned data set. Some of the starting-point creation options require it to be fixed block with a record length of 80. For other options, the record length and format don't matter.

  Although a record in a starting-point file can span more than one line, it is easier to sort a starting-point file if every record is on only one line. You may have to sort the starting-point files when you consolidate several files (see 4.3.5, "Consolidating the Starting-Point Files" on page 119).

  Especially if you use the search functions to generate a starting-point file, the records can be longer than 80 bytes. We therefore recommend you allocate two starting-point files: one fixed block with a record length of 80 and one with a longer record length, for example 256.

  As shown in Table 24 on page 98, some of the options cannot append additional starting-points to an existing starting-point file. They can only replace the existing file. This means that you might have to allocate a lot of starting-point files. Therefore, it makes sense to allocate just two starting-point files as a partitioned data sets and create a new member every time you execute a starting-point file creation function.
- **Domain File** — The domain file contains a list of programs, jobs, or both, that you want to include in the search or analysis. If you don't specify a domain file, all programs and jobs of your system (the entire dictionary) are being searched or analyzed.

  If you want to use a domain file, you must allocate it manually with any name. It can be a sequential or partitioned data set. The record format and

record length don't matter. The record length must be large enough to hold the entries, therefore at least 12 bytes.

- **Condition File** — You have to use a condition file if you have more than ten search conditions. If you have fewer than ten conditions, you can use the on-line panel to specify these conditions. But this panel is the same for all the search options. This means that you have to type the conditions again every time you use another search option, because the conditions are different for every search option. It can therefore be useful to use condition files even if you have fewer than ten conditions.

  If you use a condition file, you must allocate it manually with any name. It can be a sequential or partitioned data set. The record format and record length don't matter. The record length must be large enough to hold the entries, which depends on your specific conditions.

- **Date Identification File** — The date identification file used as input for the CCCA DATE FORMAT conversion option is created automatically by the Year 2000 analysis function if you specify the corresponding option. The record format is fixed block; the logical record length is 80. You have to specify a new name for the date identification file every time you execute this function. MA2000 does not replace or update an existing file.

4.3 Creating the Starting-Point File

Besides the dictionary of your system which you created by performing the MA2000 batch tasks, the starting-point file is the main input for the Year 2000 analysis. If you perform an individual impact analysis, you have to provide one single starting-point. But for a multiple impact analysis, like the Year 2000 analysis, you have to provide a file containing a list of one or more starting-points.

A starting-point is either a basic program variable, a substring of a variable, or a specific position within an input/output element like a file record, DB2 table row, or IMS segment. For example, if you have a variable containing a date with the format `MM/DD/YY` you might only be interested in the impact of the `YY` part of this variable. You would then define the starting-point as the offset 7, length 2 of this variable.

4.3.1 Starting-Point File Format

The starting-point file is a flat file with a specific record layout:

1. Type of item
2. Name of element the item belongs to
3. Name of item (depends on type of item; some items have no name, they are defined by the position within the element)
4. Offset (starting position) within the item or element if there is no item name
5. Length in bytes
6. Date format (optional)

The definition of a new item has to start on a new line with the item type in Position 1. The parameters that belong to one item definition are separated by a comma, and there are no blanks before and after the comma. If an item

definition is longer than the record length of the starting-point file, you can split a record after a comma.

Figure 28 shows an example of a starting-point file.

```
   ⋮
I,TARMU3B,WORK-JOINED-MMDD OF WORK-JOINED-MMDDYY OF WORK-VARS,1,6,YYXXXX
I,TARMU3B,WORK-TERMINATED-YYDDD OF WORK-VARS,1,5,YYXXX
I,TARMU3B,TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF TARDATE-PARAMETERS,1,5,
YYXXX
I,TARMU3B,TARDATE-DATE-YYMMDD OF TARDATE-DATE-RED2 OF TARDATE-PARAMETERS,1,6,
YYXXXX
I,TARMU5,WORK-DATE-YYMMDD,1,6,YYXXXX
   ⋮
```

Figure 28. Starting-Point File with Records Spanning More Than One Line

Table 26 explains the starting-point types that are available in MA2000 and the layout of their definition in a starting-point file.

Table 26 (Page 1 of 2). Starting-Point File Layout Description

Description	Item Type	Element Name	Item Name	Offset	Length
Comment line	*				
Program variable (basic item)	I	Program name	Variable name, fully qualified if part of a structure	Offset within variable	Length
Program variable (basic item) which is **not** to be considered as a starting-point by the impact analysis function.	NI	Program name	Variable name, fully qualified if part of a structure		
Position within a data set record	D	Data set name	--	Offset within record	Length
Data set which is **not** to be considered as a starting-point by the impact analysis function.	ND	Data set name			
Position within a record of any data set that contains the element name in its data set name. This starting-point is not valid for a cause or influence analysis.	F	Substring of a data set name	--	Offset within record	Length
Position within an IMS Segment	S	Segment name	--	Offset within segment	Length
IMS segment which is **not** to be considered as a starting-point by the impact analysis function.	NS	Segment name			
DB2 column	T	DB2 table or view name	Column name	Offset within column	Length
DB2 column which is **not** to be considered as a starting-point by the impact analysis function.	NT	DB2 table or view name	Column name		
Position within a CICS file record	C	CICS file name	--	Offset within record	Length
CICS file which is **not** to be considered as a starting-point by the impact analysis function.	NC	CICS file name			

Table 26 (Page 2 of 2). Starting-Point File Layout Description					
Description	**Item Type**	**Element Name**	**Item Name**	**Offset**	**Length**
Program variable defined in an %INCLUDE or copybook	L	%INCLUDE name or copybook name	Variable name, fully qualified if part of a structure. If you use the COPY REPLACING statement in a COBOL program, you have to specify the variable name as it occurs after the COBOL program preprocessing, not as it is defined in the copybook.	Offset within variable	Length
Program variable defined in an %INCLUDE or copybook which is **not** to be considered as a starting-point by the impact analysis function.	NL	%INCLUDE name or copybook name	Variable name		

As a last parameter, you can always add the date format if the item contains a date. See 4.3.4, “Using the Search Option” on page 106 for a description of the valid date formats in MA2000.

4.3.2 Domain File Format

A domain is a list of programs and jobs used to limit the scope of a search or impact analysis function. It must be a subset of your entire system (dictionary). You can either specify what programs and jobs you want to include in the operation or what programs and jobs you want to exclude from the operation.

Include Only the listed programs and jobs will be included in the search or analysis.

Exclude The entire system is being searched or analyzed except the listed programs and jobs.

The layout of the domain file is:

1. Member type
2. One blank
3. Member name

The member type must be one of the following:

PGM Program to be included

JOB Job to be included. All programs referenced in the specified job are included.

EXP Program to be excluded

EXJ Job to be excluded. All programs referenced in the specified job are excluded.

The member name is the name of the member in the program source or JCL library that you want to include or exclude.

If you specify in one domain file including as well as excluding member types, MA2000 will ignore the including member types.

Figure 29 on page 103 shows an example of a domain file:

```
PGM TARDTE3
PGM TARMU3
JOB TARMU5
```

Figure 29. Domain File

4.3.3 Using the Starting Point File Creation Option

Option A of the MA2000 main menu lets you create starting-point files for all possible items. Figure 30 shows the menu from where you can run the starting-point creation functions.

```
IMXP600--------------------  Create a Starting Point File  --------------------
 Menu ===>
                                                          Previous panel: PF3
     Enter a menu item and press Enter.
     (Specify the program name if you select 1.)

      1 Program (starting point selection)     Program name:  TARMU3

      2 Data sets (all data sets)

      3 Segments (all segments)

      4 DB2 tables (all DB2 tables)

      5 CICS files (all CICS files)

     * You can also create a starting point file from the Search menu.
```

Figure 30. Create a Starting Point File Menu

For Options 2 through 5, there is no further selection to make. MA2000 always searches the entire dictionary. This means that you have to edit the starting-point files after they have been created in order to include only items that you really want to use as starting-points.

Attention

The starting-points you specify when you run a Year 2000 analysis are considered to contain dates. MA2000 does perform some plausibility checks though. But if the information it gets from the context is not sufficient, the analysis function cannot correct an erroneous input from the starting-point file.

Therefore, you always have to check manually every starting-point file that has been created by an MA2000 function to make sure that the starting-points themselves really contain dates.

Options 2 and 4 run in batch, whereas Options 3 and 5 run in foreground. Therefore in addition to the starting-point file specification, for data sets and DB2 tables, you also have to specify the job card for the JCL (see Figure 31 on page 104).

You have to specify a separate starting-point file for every option because MA2000 replaces the files in every run. Options 2 and 5 require a starting-point file with a record length of 80 bytes as output.

```
IMXP610---------------  Create a Starting Point File (Data Set)  --------------
 Option ===>
                                                          Previous panel: PF3
                                                          Initial panel:  PF4
   Specify the starting point file in which you want to list
   data set names.  (PS or PO file preallocation is required.)

 Starting point file: 'ITSORS3.MA2000.SPOINT80(DATASET)'
 Volume name:                (if not cataloged)

   Note: Add the starting point item (offset and length) of each
         data set by editing the file.

     Job statement information:

  ==>  //ITSORS3D JOB ,
  ==>  //  MSGCLASS=H,MSGLEVEL=(1,1),NOTIFY=ITSORS3,
  ==>  //  TIME=(,59),REGION=64M
  ==>  //*
  ==>  //*
```

Figure 31. Create a Starting Point File (Data Set) Panel

The generated starting-point file for data sets is shown in Figure 32. For all the Options 2 through 5, you have to edit the starting-point files also in order to add the offset and length of your starting-points. You may have to repeat some lines if you have more than one starting-point in the same record.

```
D,ITSORS3.DEPT.MASTER.ONLINE,
D,ITSORS3.EMP.MASTER.ONLINE,
D,ITSORS3.TARMU3B.INPUT,
D,ITSORS3.TARMU5.REPORT,
D,ITSORS3.TARMU6.REPORT,
```

Figure 32. Starting-Point File Created for Data Sets

Figure 33 on page 105 shows the starting-point file after we edited it. The records which are written to the report files in the programs TARMU5 and TARMU6 are defined as just one long string. It is therefore not obvious if and where there are dates within these records. Actually that's exactly what we are using MA2000 for: to find the dates if they are not obvious. We therefore deleted the last two lines from the starting-point file.

The TARMU3B input data set and the employee master file contain more than one date. Therefore we added additional lines for these data sets.

```
D,ITSORS3.DEPT.MASTER.ONLINE,35,3
D,ITSORS3.EMP.MASTER.ONLINE,136,5
D,ITSORS3.EMP.MASTER.ONLINE,141,6
D,ITSORS3.EMP.MASTER.ONLINE,147,5
D,ITSORS3.EMP.MASTER.ONLINE,152,5
D,ITSORS3.EMP.MASTER.ONLINE,157,3
D,ITSORS3.TARMU3B.INPUT,136,8
D,ITSORS3.TARMU3B.INPUT,144,8
D,ITSORS3.TARMU3B.INPUT,152,8
D,ITSORS3.TARMU3B.INPUT,160,8
```

Figure 33. Starting-Point File for Data Sets after Editing

For Option 1 of the **Create a Starting Point File** menu, you have to specify the program name. The next panel then provides you with a list of all the Level 1 variables or structures defined in the specified program. If you select a structure, you will get a list with all the levels of this structure, as shown for example in Figure 34. From this list, you can select one or more items and immediately specify the offset and length within the items. This function appends all the items you select to the starting-point file you specify. Therefore you can go back one level, select another structure, and add items to the same starting-point file. Or you can even return to the **Create a Starting Point File** menu and specify a new program, select the structures and items you want, and add them to the same starting-point file.

```
IMXP602---------------  Create a Starting Point File (Progra Row 1 to 11 of 11
 Command ===>
                                                          Previous panel: PF3
                                                          Initial panel:  PF4
 Type S next to the item you want to select and press Enter.
 (You can select more than one item.)

    Program name: TARMU3
    Major structure: WORK-VARS

           Item name                              Position   Length
    _ 1      WORK-VARS                               1         44
    s  3     WORK-TODAYS-MMDDYY                      1          8
    _  3     WORK-MSG-CODE                           1          2
    s  3     WORK-EIB-DATE                           1          4
    _  3     WORK-EIB-DATE-CHAR                      1          7
    s   5    WORK-EIB-YYDDD                          1          5
    s  3     WORK-JOINED-YYDDD                       1          5
    _  3     WORK-JOINED-MMDDYY                      1          8
    _   5    WORK-JOINED-MMDD                        1          6
    s   5    WORK-JOINED-YY                          1          2
    s  3     WORK-TERMINATED-YYDDD                   1          5
****************************** Bottom of data ********************************
```

Figure 34. Create a Starting Point File (Program)

The generated starting-point file is shown in Figure 35 on page 106. This file does not need to be edited unless you want to add the date formats. For a system containing a lot of programs, this option to create the I-type starting-points is not the best way. The search option as described in the next section (4.3.4, “Using the Search Option” on page 106) offers a better way to handle the program variables of a large dictionary.

In our sample application we don't have any IMS or DB2 programs. Therefore we used the create a starting-point file option only for data sets and CICS files.

```
I,TARMU3,WORK-TODAYS-MMDDYY,1,8
I,TARMU3,WORK-EIB-DATE,1,4
I,TARMU3,WORK-EIB-YYDDD,1,5
I,TARMU3,WORK-JOINED-YYDDD,1,5
I,TARMU3,WORK-JOINED-YY,1,2
I,TARMU3,WORK-TERMINATED-YYDDD,1,5
```

Figure 35. Starting-Point File for Program Variables

4.3.4 Using the Search Option

With the search option, you can create a list of program variables (basic items) or data sets. This list can have the form of a report or a starting-point file. But unlike the starting-point file creation function (4.3.3, "Using the Starting Point File Creation Option" on page 103), the search option allows you to define search conditions and thus retrieve not just one item at a time, but many.

Figure 36 shows the **Search** menu with the five search options:

- Similar item
- Similar item (Data declarations)
- Synonym
- DATE function
- Data set

```
IMXP200------------------------------  Search  ------------------------------
 Command ===>
                                                          Previous panel: PF3
    Retrieval option: _       1. Similar item
    (You can select from      2. Similar item (Data declarations)
    1 to 5)                   3. Synonym
                              4. DATE function
                              5. Data set

    If you select  1, 2, 3,or 4 :
      Domain specification:  N (Y/N)    (See note.)
      Data set name:  ____________________________________________________
      Volume name:    ______     (if not cataloged)

    If you select option  5 :
      JCL specification:  N (Y/N)    (See note.)
      Data set name:  ____________________________________________________
      Volume name:    ______     (if not cataloged)

   Note: If you select N, all analysis members will be the target
         for retrieval.  Specify Y if you want to restrict the
         retrieval range.
```

Figure 36. MA2000 Search Menu

For all the search options, you can specify a domain to limit the number of programs and jobs to be searched. See 4.3.2, "Domain File Format" on page 102 for more information about the domain file. Because our sample

application contains only a few programs and jobs, we did not find it necessary to use domains. Instead we run the search and impact analysis functions against the entire system.

For Option 5 (Data set) on the **Search** menu you can specify a data set with JCL specifications. This JCL specification data set is nothing more than a domain file. However, if you have PGM statements in this file, they will be ignored by the data set search function. If you further partition your MA2000 system into domains including programs and jobs, you can use the same domain file for program variable search functions (similar item, synonym, DATE function) as well as for the data set search function.

After you select the retrieval option and optionally specify a corresponding domain file, the search condition panel is displayed as shown in Figure 37 for the similar item search.

```
IMXP201------------------------------  Search  ------------------------------
 Command ===>
                                                          Previous panel: PF3
  Retrieval option:  Similar item                         Initial panel:  PF4
  Create a starting point file:  Y  (Y/N)
    Output type:                 1  (1:Retrieval ID code, or
                                     2:Non retrieval ID code)
    Add date format:             Y  (Y/N)
    starting point append        N  (Y:append N:update)
    Data set name:    'ITSORS3.MA2000.SPOINT(SIMILAR)'
    Volume name:      ______ (if not cataloged)

 *All conditions are ignored for the DATE function option.

  Use a conditions file: Y (Y:Specify data set name, N:Specify conditions)
    Data set name:    'ITSORS3.MA2000.COND(SIMILAR)'
    Volume name:      ______ (if not cataloged)

  Conditions (up to 10):*If condition file option is N, specify the conditions.
  ________________________________________  ________________________________________
  ________________________________________  ________________________________________
  ________________________________________  ________________________________________
  ________________________________________  ________________________________________
  ________________________________________  ________________________________________
```

Figure 37. MA2000 Search Condition Panel

The type and format of the conditions depend on the retrieval option. We describe them in detail in the following sections. The first five input fields on the search condition panel are the same for all the retrieval options:

- **Create a starting point file** — If you don't want to create a starting-point file containing the items retrieved by your search request, the results are created in the form of a summary and a detailed report. You can view and print these reports by means of the individual retrieval result management function. If you specify **Y**, the results are written to the starting-point file and the reports are created as well. The starting-point files created by the search functions can have any record length. We used a starting-point file with a record length of 256 because we wanted each entire record including date formats to fit on one line. If you specify **N**, the following input fields except the condition-related fields are ignored.

- **Output type** — In the starting-point file, a basic item can have an item type I or NI. See Table 26 on page 101 for a description of the item types.

  If you specify 1, all the basic items written to the starting-point file will have an item type of I. If you specify 2, all the basic items written to the starting-point file will have an item type of NI. See 4.3.4.1, “Similar Item Search” on page 111 for an example of how to use nonretrieval ID codes.

- **Add date format** — If you choose to add date formats to the starting-point entries in the starting-point file, MA2000 automatically adds one or more possible formats as the last parameter.

  When you run the Year 2000 analysis to create a date identification file, MA2000 takes the date formats specified in the starting-point file as a basis to determine the date formats of the retrieved items during the analysis. If you don′t have date formats in the starting-point file, MA2000 will evaluate the possible formats during the analysis. But because sometimes there is more than one format possible, it is useful to evaluate the formats during the search process. You should validate the generated date formats manually in the starting-point file before you run the Year 2000 analysis. There must only be one date format per item in the starting-point file when you run the analysis. Therefore, if MA2000 proposes more than one item you have to find out which one is correct and delete the others. If the search function adds more than one date format, it also adds a comment line in the form

  ```
  *,      ATTENTION
  ```

  Using that, you can easily find the lines that you have to edit.

  MA2000 determines the possible date formats by the layout of a structure or by the length of the PICTURE clause in case of a basic element. In the following example the layout of the structure is 2 1 2 1 2:

  ```
  01  WORK-DATE
      03  WORK-YY   PIC 9(2).
      03  FILLER1   PIC 9(1).
      03  WORK-MM   PIC 9(2).
      03  FILLER2   PIC 9(1).
      03  WORK-DD   PIC 9(2).
  ```

  For structures the default rules are as follows:

Structure Layout	*Date Format*
2 1 2 1 2	YY/XX/XX
2 2 2	YYXXXX
2 2 3	YYYYXXX
4 2	YYYYXX
5	YYXXX
6	YYXXXX
6	YYYYXX
7	YYYYXXX
2 6	YYYYXXXX
8	YYYYXXXX
2 1 4	XX/YYYY
2 2	YYXX
2 1 2	YY/XX
4 3	YYYYXXX
2 5	YYYYXXX
4 1 3	YYYY/XXX
3 1 4	XXX/YYYY

4	YYYY
2	YY
4 2 2	YYYYXXXX
2 2 2 2	YYYYXXXX
4 1 2 1 2	YYYY/XX/XX
2 1 2 1 4	XX/XX/YYYY
1 2 2 2	XYYXXXX
3 2 2	XYYXXXX

For basic items the default rules are as follows:

PICTURE Length	*Date Format*
2	YY
4	YYYY
4	YYXX
5	YYXXX
6	YYXXXX
6	YYYYXX
7	YYYYXXX
8	YYYYXXXX

If you want to change these defaults, you can do it by adding parameters to the IMXBH system parameter member. See 3.2.2.4, “Parameter Members for the Retrieval Environment” on page 29 for a discussion about this member. The parameters you have to add have the following format:

sys_date_pattern([layout or length**],**”date format”)

For example, if you want to change the default for 2 2 2 from YYXXXX to XXXXYY you have to add the following line in IMXBH:

sys_date_pattern([2,2,2],”XXXXYY”)

- **starting point append** — If you specify Y the results of your search are being added at the end of the starting-point file you specify. If you specify N, the starting-point file you specify is being overwritten with the new search results.

  We chose not to append starting-points generated by a new search directly to an existing starting-point file. Instead, we created a new member with every execution of a search function and merged all these members at the end of the process to a master starting-point file used for the Year 2000 analysis (see 4.3.5, “Consolidating the Starting-Point Files” on page 119). We found it easier to edit the starting-point file after every search run. Having separate files helped to identify the generated starting-points of the current search run. Also, if you have to repeat a specific search for any reason you would have to manually delete from a common starting-point file all the records generated by the first execution.

- **Data set name** — This is the name of your starting-point file, including the member name if it is a partitioned data set. If you fully qualify the data set name, you have to put it in single quotes. Be sure not to forget the ending quote. If you don’t put the data set name in quotes, your user ID will be added as the first level qualifier.

After you specified the starting-point file options and the search conditions the job execution panel is displayed as shown in Figure 38 on page 110.

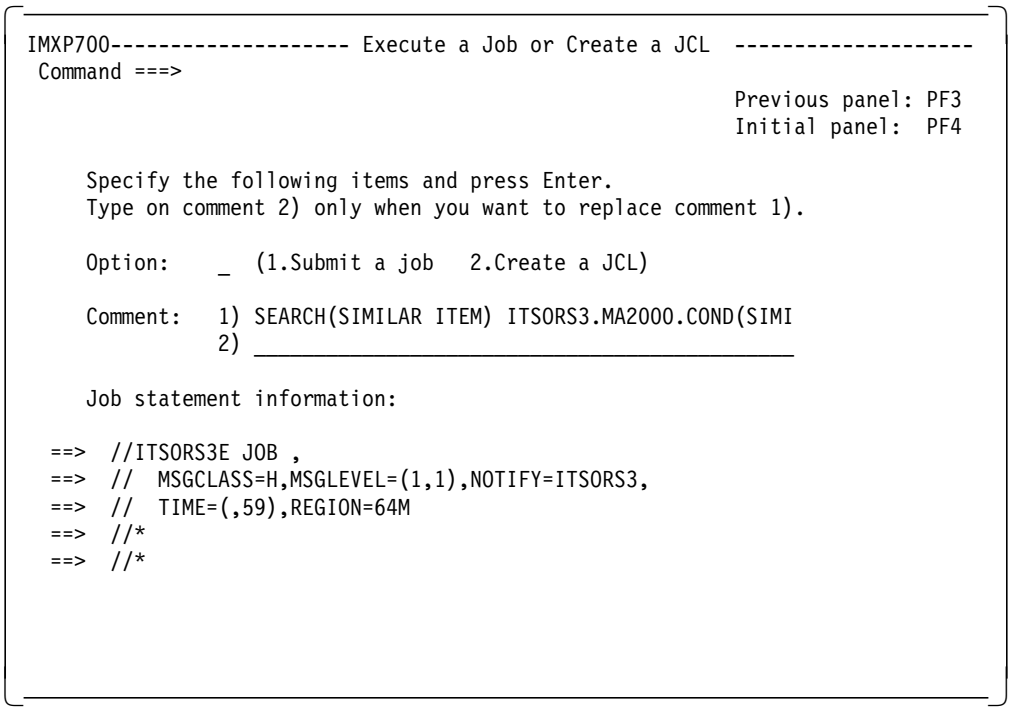


Figure 38. MA2000 Job Execution Panel

Don't forget to specify a correct job card for your environment.

MA2000 generates the **comment line 1** using the function you are performing and the condition for a search function or the starting-point for an analysis option. If you leave the **comment line 2** blank, the comment line 1 is used wherever a list of your jobs occurs, such as the list of deferred jobs (Figure 39 on page 111) or the retrieval result management panels (Figure 48 on page 123).

If you run many similar functions, you may want to specify a comment on Line 2 that is more meaningful to you than the generated comment. If **comment line 2** is not blank, it is used in the lists instead of **comment line 1**.

If you select **Option 1**, the JCL will be generated and immediately submitted. If you select **Option 2**, the JCL will be created in a temporary data set. You can then choose **Option B** (Deferred job) from the MA2000 Main Menu to get a list of the generated jobs, as shown in Figure 39 on page 111. From this list you can execute the following actions against the JCL files:

Action code	***Action***
E	Edit the temporary JCL file. We used this action to eliminate the VOL=SER= statements (by means of an ISPF edit macro) when the volume serial that we specified as a system parameter was temporarily full.
D	Delete the temporary JCL file.
S	Submit the JCL. After submitting the JCL, the temporary file is deleted automatically.

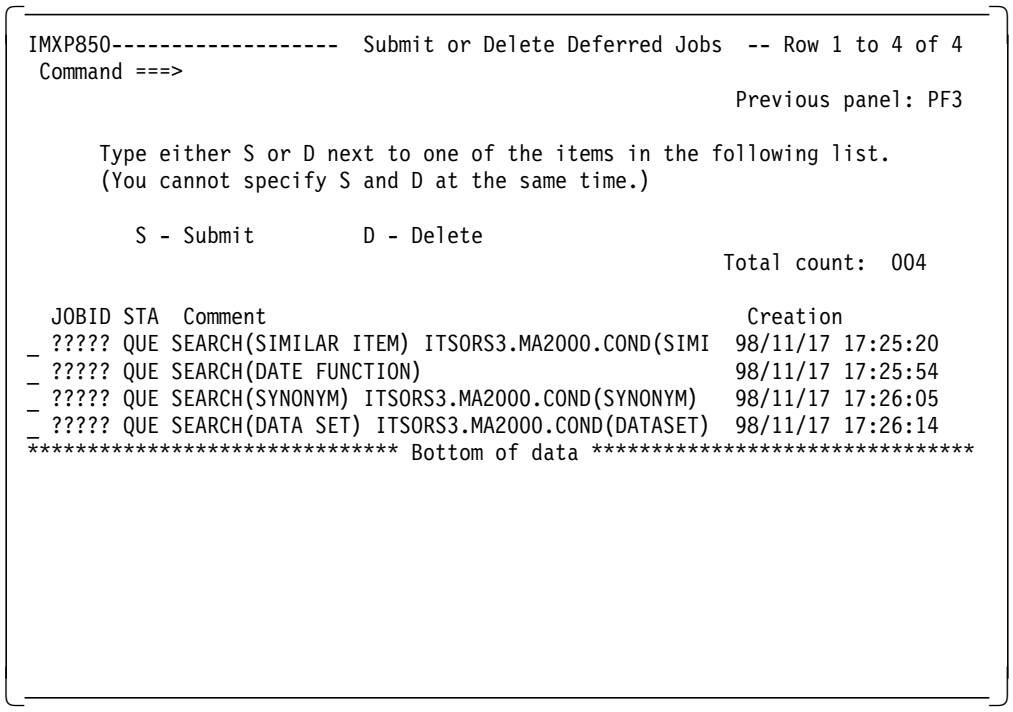


Figure 39. MA2000 Deferred Jobs Panel

4.3.4.1 Similar Item Search

For the similar item search, you must specify patterns of variable names that you are looking for. These patterns are called *search conditions*. You have to provide your search conditions either on the *Search* condition panel (Figure 37 on page 107) or in the condition file that you specify on this panel.

You can use the following wild card characters to specify a pattern:

***** An asterisk holds a place for zero or more characters. For PL/I source, it can also include a period used to qualify an element of a structure. If you specify a standalone *, all the basic variables of all the structures and all the elementary items defined in all the programs of your system are retrieved. Two consecutive asterisks (**) are not allowed in the search condition. The search will not find anything, but no error message will be issued.

? A question mark holds place for exactly one character. For PL/I source, it does not include a period. If you are looking for all variables that have a name of 7 characters you can specify ??????? as a search condition.

You can combine * and ? in one search condition.

If you retrieve basic variables of a structure, the variable names listed in the output are always fully qualified.

As an example, Figure 40 on page 112 shows the condition file we used for the similar item search. As output, we created a starting-point file including possible date formats. This starting-point file is to be used for the Year 2000 analysis.

```
*DATE*
*DTE*
*YY*
*YYYY*
*YR*
*YEAR*
*CC*
*CEN*
*GREGORIAN*
*JULIAN*
*DOB*
*YY*MM*DD*
*YMD*
*Y2*
*Y4*
```

Figure 40. Condition File for Similar Item Search for Dates

In non-English-speaking countries these conditions may be different. Please note that we did not specify *MM* or *DD*. For Year 2000 remediation, we are only interested in the year part of a date variable.

Figure 41 shows the resulting starting-point file.

```
*,TARDTE3,WORK-DATES,1,56
I,TARDTE3,WRK-DATE OF WORK-DATES,1,6,YYXXXX
I,TARDTE3,WRK-DATE-YY OF WRK-DATE OF WORK-DATES,1,2,YY
*,TARDTE3,WRK-DATE-MM OF WRK-DATE OF WORK-DATES,1,2
*,TARDTE3,WRK-DATE-DD OF WRK-DATE OF WORK-DATES,1,2
I,TARDTE3,WRK-DATE-RED OF WORK-DATES,1,6,YYXXXX
I,TARDTE3,WRK-DTE-YY OF WRK-DATE-RED OF WORK-DATES,1,2,YY
*,TARDTE3,WRK-DTE-MM OF WRK-DATE-RED OF WORK-DATES,1,2
*,TARDTE3,WRK-DTE-MM-NUM OF WRK-DATE-RED OF WORK-DATES,1,2
*,TARDTE3,WRK-DTE-DD OF WRK-DATE-RED OF WORK-DATES,1,2
*,TARDTE3,WRK-DTE-DD-NUM OF WRK-DATE-RED OF WORK-DATES,1,2
*,TARDTE3,WRK-DATE-YYDDD OF WORK-DATES,1,5
I,TARDTE3,WRK-DATE-YY OF WRK-DATE-YYDDD OF WORK-DATES,1,2,YY
*,TARDTE3,WRK-DATE-DDD OF WRK-DATE-YYDDD OF WORK-DATES,1,3
*,TARDTE3,WRK-DATE-YYDDD-RED OF WORK-DATES,1,5
*,TARDTE3,WRK-DATE-DDD-NUM OF WRK-DATE-YYDDD-RED OF WORK-DATES,1,3
I,TARDTE3,WRK-YEAR-YYYY OF WORK-DATES,1,4,YYXX
*,TARDTE3,WRK-YEAR-YY OF WRK-YEAR-YYYY OF WORK-DATES,1,2
*,      ATTENTION
I,TARDTE3,WRK-YEAR-YYYY-NUM OF WORK-DATES,1,4,YYYY,YYXX
I,TARDTE3,WRK-DATE-MMXDDXYY OF WORK-DATES,1,8,XX/XX/YY
I,TARDTE3,WRK-DATE-MM OF WRK-DATE-MMXDDXYY OF WORK-DATES,1,2,YY
*,TARDTE3,WRK-DATE-DD OF WRK-DATE-MMXDDXYY OF WORK-DATES,1,2
I,TARDTE3,WRK-DATE-YY OF WRK-DATE-MMXDDXYY OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-DATE-YYXDDXMM OF WORK-DATES,1,8,YY/XX/XX
I,TARDTE3,WRK-DATE-YY OF WRK-DATE-YYXDDXMM OF WORK-DATES,1,2,YY
*,TARDTE3,WRK-DATE-DD OF WRK-DATE-YYXDDXMM OF WORK-DATES,1,2
*,TARDTE3,WRK-DATE-MM OF WRK-DATE-YYXDDXMM OF WORK-DATES,1,2
I,TARDTE3,INPUT-DATE,1,8,YYYYXXXX
I,TARMU3,EMP-DATE-JOINED OF EMPLOYEE-MASTER-RECORD,1,5,YYXXX
```

Figure 41 (Part 1 of 3). Starting-Point File Generated by Similar Item Search

```
*,      ATTENTION
I,TARMU3,EMP-DATE-TERMINATED OF EMPLOYEE-MASTER-RECORD,1,6,YYXXXX,YYYYXX
I,TARMU3,EMP-DATE-MAINTAINED OF EMPLOYEE-MASTER-RECORD,1,5,YYXXX
I,TARMU3,EMP-BIRTH-DATE OF EMPLOYEE-MASTER-RECORD,1,5,YYXXX
I,TARMU3,WORK-TODAYS-MMDDYY OF WORK-VARS,1,8,YYYYXXXX
I,TARMU3,WORK-EIB-DATE OF WORK-VARS,1,4,YYYYXXX
I,TARMU3,WORK-EIB-DATE-CHAR OF WORK-VARS,1,7,YYYYXXX
I,TARMU3,WORK-EIB-YYDDD OF WORK-EIB-DATE-CHAR OF WORK-VARS,1,5,YYXXX
I,TARMU3,WORK-JOINED-YYDDD OF WORK-VARS,1,5,YYXXX
*,TARMU3,WORK-JOINED-MMDDYY OF WORK-VARS,1,8
I,TARMU3,WORK-JOINED-YY OF WORK-JOINED-MMDDYY OF WORK-VARS,1,2,YY
I,TARMU3,WORK-TERMINATED-YYDDD OF WORK-VARS,1,5,YYXXX
*,TARMU3,TARDATE-PARAMETERS,1,54
I,TARMU3,TARDATE-DATE OF TARDATE-PARAMETERS,1,8,YYYYXXXX
*,TARMU3,TARDATE-DATE-RED1 OF TARDATE-PARAMETERS,1,8
I,TARMU3,TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF TARDATE-PARAMETERS,1,5,YYXXX
*,TARMU3,TARDATE-DATE-RED2 OF TARDATE-PARAMETERS,1,8
*,      ATTENTION
I,TARMU3,TARDATE-DATE-YYMMDD OF TARDATE-DATE-RED2 OF TARDATE-PARAMETERS,1,6,YYXXXX,YYYYXX
I,TARMU3,TARDATE-INPUT-FORMAT OF TARDATE-PARAMETERS,1,8,YYYYXXXX
I,TARMU3,TARDATE-OUTPUT-FORMAT OF TARDATE-PARAMETERS,1,8,YYYYXXXX
*,TARMU3,TARDATE-MESSAGE OF TARDATE-PARAMETERS,1,30
*,      ATTENTION
I,TARMU3,TARMU3MJDATEL OF TARMU3MI,1,2,YYYY,YYXX
*,TARMU3,TARMU3MJDATEF OF TARMU3MI,1,1
*,TARMU3,TARMU3MJDATEA OF FILLER OF TARMU3MI,1,1
I,TARMU3,TARMU3MJDATEI OF TARMU3MI,1,8,YYYYXXXX
*,      ATTENTION
I,TARMU3,TARMU3MBDATEL OF TARMU3MI,1,2,YYYY,YYXX
*,TARMU3,TARMU3MBDATEF OF TARMU3MI,1,1
*,TARMU3,TARMU3MBDATEA OF FILLER OF TARMU3MI,1,1
I,TARMU3,TARMU3MBDATEI OF TARMU3MI,1,8,YYYYXXXX
*,      ATTENTION
I,TARMU3,TARMU3MTDATEL OF TARMU3MI,1,2,YYYY,YYXX
*,TARMU3,TARMU3MTDATEF OF TARMU3MI,1,1
*,TARMU3,TARMU3MTDATEA OF FILLER OF TARMU3MI,1,1
I,TARMU3,TARMU3MTDATEI OF TARMU3MI,1,8,YYYYXXXX
*,      ATTENTION
I,TARMU3,TARMU3MSDATEL OF TARMU3MI,1,2,YYYY,YYXX
*,TARMU3,TARMU3MSDATEF OF TARMU3MI,1,1
*,TARMU3,TARMU3MSDATEA OF FILLER OF TARMU3MI,1,1
I,TARMU3,TARMU3MSDATEI OF TARMU3MI,1,8,YYYYXXXX
I,TARMU3,TARMU3MJDATEO OF TARMU3MO,1,8,YYYYXXXX
I,TARMU3,TARMU3MBDATEO OF TARMU3MO,1,8,YYYYXXXX
I,TARMU3,TARMU3MTDATEO OF TARMU3MO,1,8,YYYYXXXX
I,TARMU3,TARMU3MSDATEO OF TARMU3MO,1,8,YYYYXXXX
I,TARMU3B,EMP-DATE-JOINED OF EMPLOYEE-MASTER-RECORD,1,5,YYXXX
*,      ATTENTION
I,TARMU3B,EMP-DATE-TERMINATED OF EMPLOYEE-MASTER-RECORD,1,6,YYXXXX,YYYYXX
I,TARMU3B,EMP-DATE-MAINTAINED OF EMPLOYEE-MASTER-RECORD,1,5,YYXXX
I,TARMU3B,EMP-BIRTH-DATE OF EMPLOYEE-MASTER-RECORD,1,5,YYXXX
I,TARMU3B,INP-EMP-DATE-JOINED OF INPUT-RECORD,1,8,YYYYXXXX
I,TARMU3B,INP-EMP-DATE-TERMINATED OF INPUT-RECORD,1,8,YYYYXXXX
I,TARMU3B,INP-EMP-BIRTH-DATE OF INPUT-RECORD,1,8,YYYYXXXX
I,TARMU3B,OUT-EMP-DATE-JOINED OF OUTPUT-RECORD,1,8,YYYYXXXX
I,TARMU3B,OUT-EMP-DATE-TERMINATED OF OUTPUT-RECORD,1,8,YYYYXXXX
I,TARMU3B,OUT-EMP-BIRTH-DATE OF OUTPUT-RECORD,1,8,YYYYXXXX
I,TARMU3B,WORK-TODAYS-MMDDYY OF WORK-VARS,1,8,YYYYXXXX
I,TARMU3B,WORK-EIB-DATE OF WORK-VARS,1,7,YYYYXXX
I,TARMU3B,WORK-EIB-DATE-RED OF WORK-VARS,1,7,YYYYXXX
I,TARMU3B,WORK-EIB-YYDDD OF WORK-EIB-DATE-RED OF WORK-VARS,1,5,YYXXX
I,TARMU3B,WORK-JOINED-YYDDD OF WORK-VARS,1,5,YYXXX
*,TARMU3B,WORK-JOINED-MMDDYY OF WORK-VARS,1,8
I,TARMU3B,WORK-JOINED-YY OF WORK-JOINED-MMDDYY OF WORK-VARS,1,2,YY
```

Figure 41 (Part 2 of 3). Starting-Point File Generated by Similar Item Search

```
I,TARMU3B,WORK-TERMINATED-YYDDD OF WORK-VARS,1,5,YYXXX
*,TARMU3B,TARDATE-PARAMETERS,1,54
I,TARMU3B,TARDATE-DATE OF TARDATE-PARAMETERS,1,8,YYYYXXXX
*,TARMU3B,TARDATE-DATE-RED1 OF TARDATE-PARAMETERS,1,8
I,TARMU3B,TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF TARDATE-PARAMETERS,1,5,YYXXX
*,TARMU3B,TARDATE-DATE-RED2 OF TARDATE-PARAMETERS,1,8
*,      ATTENTION
I,TARMU3B,TARDATE-DATE-YYMMDD OF TARDATE-DATE-RED2 OF TARDATE-PARAMETERS,1,6,YYXXXX,YYYYXX
I,TARMU3B,TARDATE-INPUT-FORMAT OF TARDATE-PARAMETERS,1,8,YYYYXXXX
I,TARMU3B,TARDATE-OUTPUT-FORMAT OF TARDATE-PARAMETERS,1,8,YYYYXXXX
*,TARMU3B,TARDATE-MESSAGE OF TARDATE-PARAMETERS,1,30
*,      ATTENTION
I,TARMU5,WORK-DATE-YYMMDD,1,6,YYXXXX,YYYYXX
I,TARMU5,RPT-HD1-DATE OF REPORT-HEAD1,1,8,XX/XX/YY
I,TARMU5,RPT-HD1-YY OF RPT-HD1-DATE OF REPORT-HEAD1,1,2,YY
*,TARMU5,TARDATE-PARAMETERS,1,54
I,TARMU5,TARDATE-DATE OF TARDATE-PARAMETERS,1,8,YYYYXXXX
*,TARMU5,TARDATE-DATE-RED1 OF TARDATE-PARAMETERS,1,8
I,TARMU5,TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF TARDATE-PARAMETERS,1,5,YYXXX
I,TARMU5,TARDATE-INPUT-FORMAT OF TARDATE-PARAMETERS,1,8,YYYYXXXX
I,TARMU5,TARDATE-OUTPUT-FORMAT OF TARDATE-PARAMETERS,1,8,YYYYXXXX
*,TARMU5,TARDATE-MESSAGE OF TARDATE-PARAMETERS,1,30
I,TARMU6,EMPLOYEE-DATE-JOINED OF EMPLOYEE-MASTER-RECORD,1,5,YYXXX
*,      ATTENTION
I,TARMU6,EMPLOYEE-DATE-TERMINATED OF EMPLOYEE-MASTER-RECORD,1,6,YYXXXX,YYYYXX
*,      ATTENTION
I,TARMU6,WORK-DATE-YYMMDD,1,6,YYXXXX,YYYYXX
*,      ATTENTION
I,TARMU6,WORK-DATE-YYDDD,1,6,YYXXXX,YYYYXX
I,TARMU6,RPT-HD1-DATE OF REPORT-HEAD1,1,8,XX/XX/YY
I,TARMU6,RPT-HD1-YY OF RPT-HD1-DATE OF REPORT-HEAD1,1,2,YY
*,TARMU6,TARDATE-PARAMETERS,1,54
I,TARMU6,TARDATE-DATE OF TARDATE-PARAMETERS,1,8,YYYYXXXX
*,TARMU6,TARDATE-DATE-RED1 OF TARDATE-PARAMETERS,1,8
I,TARMU6,TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF TARDATE-PARAMETERS,1,5,YYXXX
*,TARMU6,TARDATE-DATE-RED3 OF TARDATE-PARAMETERS,1,8
*,      ATTENTION
I,TARMU6,TARDATE-DATE-MMDD OF TARDATE-DATE-RED3 OF TARDATE-PARAMETERS,1,6,YYXXXX,YYYYXX
I,TARMU6,TARDATE-DATE-YY OF TARDATE-DATE-RED3 OF TARDATE-PARAMETERS,1,2,YY
I,TARMU6,TARDATE-INPUT-FORMAT OF TARDATE-PARAMETERS,1,8,YYYYXXXX
I,TARMU6,TARDATE-OUTPUT-FORMAT OF TARDATE-PARAMETERS,1,8,YYYYXXXX
*,TARMU6,TARDATE-MESSAGE OF TARDATE-PARAMETERS,1,30
```

Figure 41 (Part 3 of 3). Starting-Point File Generated by Similar Item Search

All variables that meet the search condition have been retrieved. But some are put in comment because MA2000 recognizes them from the format or the context as not containing years. The first item in the list (WORK-DATES), for example, has a length of 56 and therefore contains no date. The next item (WRK-DATE) has a length of 6 and is interpreted as having the format YYXXXX (structure layout 2 2 2). The first subitem of WRK-DATE is therefore considered to be the year part. WRK-DATE-YY has the format YY. The next two subitems (WRK-DATE-MM and WRK-DATE-DD) are considered to be the nonyear part of WRK-DATE. Therefore they are put as comment even if they meet the search condition *DATE*.

This does not always work. If WRK-DATE would be called WRK-AREA, for example, it would not be retrieved. As a consequence, both WRK-DATE-MM and WRK-DATE-DD would be incorrectly retrieved with a format of YY in addition to WRK-DATE-YY.

It is therefore very important that you check the generated starting-point file, item by item, and decide whether an item should be in a comment line or not. You also have to decide which date format is correct wherever more than one format is proposed. These items are indicated by a comment like

```
*,      ATTENTION
```

You can use the nonretrieval ID code option on the search condition panel (Figure 37 on page 107) to create NI items in the starting-point file. During the analysis, all the items with item type NI are ignored. But this works only if you have the same item twice in the starting-point file: once with an item type I, and once with a type NI.

We used the conditions shown in Figure 42 to create an additional starting-point file with NI entries. Again, check the generated output to make sure that this is what you really want.

```
*DATE*MM*
*DATE*DD*
*DTE*MM*
*DTE*DD*
```

Figure 42. Condition File for Similar Item Search (Nonretrieval ID)

You can also use this technique to exclude, for example, anything like *UPDATE* or to exclude all the CICS map variables.

Figure 43 shows the output generated with this condition file. As expected, it includes also correct items such as variables with a name (and format) like YYDDD and YYMMDD. We therefore deleted these correct items manually from the starting-point file.

```
NI,TARDTE3,WRK-DATE-YYDDD OF WORK-DATES
NI,TARDTE3,WRK-DATE-YYDDD-RED OF WORK-DATES
NI,TARDTE3,WRK-DATE-MMXDDXYY OF WORK-DATES
NI,TARDTE3,WRK-DATE-YYXDDXMM OF WORK-DATES
NI,TARDTE3,WRK-DATE-MM OF WRK-DATE OF WORK-DATES
NI,TARDTE3,WRK-DATE-DD OF WRK-DATE OF WORK-DATES
NI,TARDTE3,WRK-DTE-MM OF WRK-DATE-RED OF WORK-DATES
NI,TARDTE3,WRK-DTE-MM-NUM OF WRK-DATE-RED OF WORK-DATES
NI,TARDTE3,WRK-DTE-DD OF WRK-DATE-RED OF WORK-DATES
NI,TARDTE3,WRK-DTE-DD-NUM OF WRK-DATE-RED OF WORK-DATES
NI,TARDTE3,WRK-DATE-DDD OF WRK-DATE-YYDDD OF WORK-DATES
NI,TARDTE3,WRK-DATE-DDD-NUM OF WRK-DATE-YYDDD-RED OF WORK-DATES
NI,TARDTE3,WRK-DATE-MM OF WRK-DATE-MMXDDXYY OF WORK-DATES
NI,TARDTE3,WRK-DATE-DD OF WRK-DATE-MMXDDXYY OF WORK-DATES
NI,TARDTE3,WRK-DATE-DD OF WRK-DATE-YYXDDXMM OF WORK-DATES
NI,TARDTE3,WRK-DATE-MM OF WRK-DATE-YYXDDXMM OF WORK-DATES
NI,TARMU3,TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF TARDATE-PARAMETERS
NI,TARMU3,TARDATE-DATE-YYMMDD OF TARDATE-DATE-RED2 OF TARDATE-PARAMETERS
NI,TARMU3B,TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF TARDATE-PARAMETERS
NI,TARMU3B,TARDATE-DATE-YYMMDD OF TARDATE-DATE-RED2 OF TARDATE-PARAMETERS
```

Figure 43 (Part 1 of 2). NI Items Generated by Nonretrieval ID Search

```
NI,TARMU5,WORK-DATE-YYMMDD
NI,TARMU5,TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF TARDATE-PARAMETERS
NI,TARMU6,WORK-DATE-YYMMDD
NI,TARMU6,WORK-DATE-YYDDD
NI,TARMU6,TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF TARDATE-PARAMETERS
NI,TARMU6,TARDATE-DATE-MMDD OF TARDATE-DATE-RED3 OF TARDATE-PARAMETERS
```

Figure 43 (Part 2 of 2). NI Items Generated by Nonretrieval ID Search

4.3.4.2 DATE Function Search

The DATE function search does not require any condition. All the items of your system or domain are retrieved that have a system date assigned, as well as the source or target variables (or both) of a date-related intrinsic function (COBOL):

CICS
EIBDATE
FORMATTIME (all formats)

DB2
DATE (all formats)
TIMESTAMP

COBOL
DATE
DATE YYYYMMDD
DAY
DAY YYYYDDD
CURRENT-DATE
DATEVAL
YEARWINDOW
DATE-OF-INTEGER (target argument)
DATE-TO-YYYYMMDD (source/target arguments)
DAY-OF-INTEGER (target argument)
DAY-TO-YYYYDDD (source/target arguments)
YEAR-TO-YYYY (source/target arguments)
INTEGER-OF-DATE (source argument)
INTEGER-OF-DAY (source argument)

PL/I
DATE
DATETIME

If you are performing a Year 2000 analysis, the DATE function items do not have to be in your starting-point file. The Year 2000 analysis automatically considers the items that would be retrieved by the DATE function search in addition to the items listed in your starting-point file.

However, we used the DATE function option to create additional input for the synonym search (see 4.3.4.3, “Synonym Search” on page 117). Figure 44 on page 117 shows the starting-point file created by the DATE function search against our sample system. It should not be necessary to edit these items because they always contain dates with a defined format.

```
I,TARMU3,WORK-EIB-DATE OF WORK-VARS,1,4,XXYYXXX
I,TARMU3B,WORK-EIB-YYDDD OF WORK-EIB-DATE-RED OF WORK-VARS,1,5,YYXXX
I,TARMU5,WORK-DATE-YYMMDD,1,6,YYXXXX
I,TARMU6,WORK-DATE-YYMMDD,1,6,YYXXXX
I,TARMU6,WORK-DATE-YYDDD,1,6,YYXXX
```

Figure 44. Starting-Point Items Generated by the DATE Function Search

4.3.4.3 Synonym Search

The synonym search function analyzes only the data definition part of your programs. It retrieves all the variables specified as a condition as well as overlaid items of these variables. Overlaid items are:

- DEFINED items in PL/I including position
- REDEFINES in COBOL
- BASED items with the same base in PL/I

The items specified as conditions must not contain any wild cards. We took the starting-point file as it was after the similar and date function search and edited it for use as a condition file for the synonym search. We deleted all the nonretrieval ID (NI) lines and eliminated the following:

- Item types (I)
- Program names
- Offset specifications
- Length specifications
- Date formats

If you have a huge starting-point file, this can be a lot of work. You might want to write a small program to do this editing.

After you perform the synonym search, check the generated starting-point file again as you did after the similar-item search. Decide on the correct date format wherever you find the `ATTENTION` comment line.

In our sample system, the synonym search did not retrieve any item that was not in the condition file, and therefore in the starting-point file generated by the similar item search. This is because all the REDEFINES also have names that matched the similar-item search patterns. During the last step (see 4.3.5, “Consolidating the Starting-Point Files” on page 119), we remove all the duplicate starting-points from the starting-point file.

4.3.4.4 Data Set Search

We performed a data set starting-point function already in 4.3.3, “Using the Starting Point File Creation Option” on page 103. The difference between the data set search option and the data set starting-point creation option is that if you use the search option, you can limit the search by specifying conditions. The generated starting-point file, however, still requires that you manually edit it in order to provide the position within the data set record.

Because we did use the starting-point file creation option, which retrieves all the data sets of the entire system, we did not need to execute the data set search. However, if you are using the data set search for any reason, you have to be aware that the search conditions for the data set search follow slightly different

rules than the conditions for the similar item search. This is the reason why we briefly discuss these rules here.

If you are using a domain file, you have to specify it in another entry field on the **Search** menu (see Figure 36 on page 106).

As with the similar item search you can use asterisks (*) and question marks (?) as wild cards in your conditions (see 4.3.4.1, “Similar Item Search” on page 111). But wild cards cannot stand for periods used to separate the qualifiers of a data set name from each other.

The rules are as follows:

- All the lower level qualifiers following your pattern specification will be retrieved.
- Higher level qualifiers preceding your pattern specification will not be retrieved.
- If you specify several qualifiers by using periods in the pattern, at least the specified number of qualifiers must match for data set names to be retrieved. The first two rules apply in addition.

Table 27 provides some examples to illustrate the above rules:

Table 27. Samples of Pattern Specifications for Data Set Search

Pattern Specification	Data Set Names Retrieved	Data Set Names Not Retrieved
HLQ	HLQ HLQ.SECLQ HLQ.SECLQ.THIRDLQ HLQ.SECLQ.THIRDLQ.FOURTHLQ	FIRST.HLQ FIRST.HLQ.SECLQ HLQ1.SECLQ
HLQ.*	HLQ HLQ.SECLQ HLQ.SECLQ.THIRDLQ HLQ.SECLQ.THIRDLQ.FOURTHLQ	FIRST.HLQ FIRST.HLQ.SECLQ HLQ1.SECLQ
HLQ*	HLQ HLQ.SECLQ HLQ.SECLQ.THIRDLQ HLQ1.SECLQ HLQ2.SECLQ.THIRDLQ	FIRST.HLQ FIRST.HLQ.SECLQ
HLQ?	HLQ1 HLQ1.SECLQ HLQ2.SECLQ.THIRDLQ	HLQ HLQ.SECLQ HLQ12.SECLQ FIRST.HLQ1
*.SECLQ	HLQ.SECLQ HLQ1.SECLQ HLQ.SECLQ.THIRDLQ HLQ2.SECLQ.THIRDLQ.FOURTHLQ	FIRST.HLQ.SECLQ
HLQ.*.*	HLQ.SECLQ.THIRDLQ HLQ.SECLQ.THIRDLQ.FOURTHLQ	HLQ.SECLQ FIRST.HLQ.SECLQ.THIRDLQ
..THIRDLQ	HLQ.SECLQ.THIRDLQ HLQ.SECLQ.THIRDLQ.FOURTHLQ	SECLQ.THIRDLQ FIRST.HLQ.SECLQ.THIRDLQ

4.3.5 Consolidating the Starting-Point Files

During the starting-point creation and the search procedures we created several starting-point files. These files sometimes contain at least some of the same entries. The next step in creating the final starting-point file to be used for the Year 2000 analysis is to combine all the various starting-point files and eliminate all the duplicate entries.

To do this, you can run the sort program with the corresponding sort options. But first you have to make sure that all your starting-point files have the same data set properties. As mentioned in 4.3.3, "Using the Starting Point File Creation Option" on page 103, some functions require the output starting-point file to have a record length of 80 bytes. You have to copy these files to a data set with the same record length as your starting-point files generated by the search functions. In our example, we used partitioned data sets. We therefore copied the members generated by the data set and CICS file starting-point creation functions to the starting-point file we used for the search functions.

Figure 45 shows the JCL we used to create our master starting-point file.

```
//ITSORS3M JOB ,
//             MSGLEVEL=(1,1),MSGCLASS=H,TIME=(,59),
//             NOTIFY=&SYSUID,USER=&SYSUID
//*
//MERGE   EXEC PGM=ICEMAN
//SYSPRINT DD SYSOUT=*
//SYSOUT   DD SYSOUT=*
//SORTIN   DD DSN=ITSORS3.MA2000.SPOINT(SIMILAR),DISP=SHR
//         DD DSN=ITSORS3.MA2000.SPOINT(DATEFUN),DISP=SHR
//         DD DSN=ITSORS3.MA2000.SPOINT(SYNONYM),DISP=SHR
//         DD DSN=ITSORS3.MA2000.SPOINT(DATASET),DISP=SHR
//         DD DSN=ITSORS3.MA2000.SPOINT(CICSFIL),DISP=SHR
//SORTOUT  DD DSN=ITSORS3.MA2000.SPOINT(YEAR2000),DISP=SHR
//SYSIN    DD *
           SORT FIELDS=(1,256,CH,A)
           SUM FIELDS=NONE
/*
```

Figure 45. Using DFSORT to Merge Starting-Point Files

Make sure that you don't have any records in the input members that span more than one line.

If you didn't check the generated starting-point files after every search and starting-point file creation function, you have to do it now before you run the Year 2000 analysis. If you are not sure whether a specific item should be in the starting-point file or not, or if you don't know which of the proposed date formats is correct, you may have to check the program source.

With our relatively few and short sample programs, we already generated a lot of starting-points. With hundreds of starting-points, it can be a lot of work to check every item and to find out the date formats, especially if you didn't develop the programs yourself. If you are not sure about an item, don't lose too much time checking the programs, just delete the item from the starting-point file. After all this is the reason why you are running the Year 2000 analysis: to find the dates. Usually with a smaller starting-point file, you can also find all the date variables in your programs. It is more important to have correct items and

maybe the correct date formats in the starting-point file than to have as many items as possible. The Millennium Language Extensions discussed in this book can also help you to find variables that you might have missed during the find phase because the starting-point file was not complete. And finally, someone is going to test the converted programs during the test phase.

Figure 46 shows our merged starting-point file.

```
C,DEPMAST,35,3
C,EMPMAST,136,5
C,EMPMAST,141,6
C,EMPMAST,147,5
C,EMPMAST,152,5
C,EMPMAST,157,3
D,ITSORS3.DEPT.MASTER.ONLINE,35,3
D,ITSORS3.EMP.MASTER.ONLINE,136,5
D,ITSORS3.EMP.MASTER.ONLINE,141,6
D,ITSORS3.EMP.MASTER.ONLINE,147,5
D,ITSORS3.EMP.MASTER.ONLINE,152,5
D,ITSORS3.EMP.MASTER.ONLINE,157,3
D,ITSORS3.TARMU3B.INPUT,136,8
D,ITSORS3.TARMU3B.INPUT,144,8
D,ITSORS3.TARMU3B.INPUT,152,8
D,ITSORS3.TARMU3B.INPUT,160,8
I,TARDTE3,WRK-DATE OF WORK-DATES,1,6,YYXXXX
I,TARDTE3,WRK-DATE-DD OF WRK-DATE OF WORK-DATES,1,2
I,TARDTE3,WRK-DATE-DD OF WRK-DATE OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-DATE-DD OF WRK-DATE-MMXDDXYY OF WORK-DATES,1,2
I,TARDTE3,WRK-DATE-DD OF WRK-DATE-MMXDDXYY OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-DATE-DD OF WRK-DATE-YYXDDXMM OF WORK-DATES,1,2
I,TARDTE3,WRK-DATE-DD OF WRK-DATE-YYXDDXMM OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-DATE-DDD OF WRK-DATE-YYDDD OF WORK-DATES,1,3
I,TARDTE3,WRK-DATE-DDD-NUM OF WRK-DATE-YYDDD-RED OF WORK-DATES,1,3
I,TARDTE3,WRK-DATE-MM OF WRK-DATE OF WORK-DATES,1,2
I,TARDTE3,WRK-DATE-MM OF WRK-DATE OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-DATE-MM OF WRK-DATE-MMXDDXYY OF WORK-DATES,1,2
I,TARDTE3,WRK-DATE-MM OF WRK-DATE-MMXDDXYY OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-DATE-MM OF WRK-DATE-YYXDDXMM OF WORK-DATES,1,2
I,TARDTE3,WRK-DATE-MM OF WRK-DATE-YYXDDXMM OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-DATE-MMXDDXYY OF WORK-DATES,1,8,XX/XX/YY
I,TARDTE3,WRK-DATE-MMXDDXYY OF WORK-DATES,1,8,YY/XX/XX
I,TARDTE3,WRK-DATE-RED OF WORK-DATES,1,6,YYXXXX
I,TARDTE3,WRK-DATE-YY OF WRK-DATE OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-DATE-YY OF WRK-DATE-MMXDDXYY OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-DATE-YY OF WRK-DATE-YYDDD OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-DATE-YY OF WRK-DATE-YYXDDXMM OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-DATE-YYDDD OF WORK-DATES,1,5,YYXXX
I,TARDTE3,WRK-DATE-YYDDD-RED OF WORK-DATES,1,5
I,TARDTE3,WRK-DATE-YYDDD-RED OF WORK-DATES,1,5,YYXXX
I,TARDTE3,WRK-DATE-YYXDDXMM OF WORK-DATES,1,8,YY/XX/XX
I,TARDTE3,WRK-DTE-DD OF WRK-DATE-RED OF WORK-DATES,1,2
I,TARDTE3,WRK-DTE-DD OF WRK-DATE-RED OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-DTE-DD-NUM OF WRK-DATE-RED OF WORK-DATES,1,2
I,TARDTE3,WRK-DTE-DD-NUM OF WRK-DATE-RED OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-DTE-MM OF WRK-DATE-RED OF WORK-DATES,1,2
I,TARDTE3,WRK-DTE-MM OF WRK-DATE-RED OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-DTE-MM-NUM OF WRK-DATE-RED OF WORK-DATES,1,2
I,TARDTE3,WRK-DTE-MM-NUM OF WRK-DATE-RED OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-DTE-YY OF WRK-DATE-RED OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-YEAR-YY OF WRK-YEAR-YYYY OF WORK-DATES,1,2,YY
I,TARDTE3,WRK-YEAR-YYYY OF WORK-DATES,1,4,YYXX
I,TARDTE3,WRK-YEAR-YYYY OF WORK-DATES,1,4,YYYY
I,TARDTE3,WRK-YEAR-YYYY-NUM OF WORK-DATES,1,4,YYYY
I,TARMU3,EMP-BIRTH-DATE OF EMPLOYEE-MASTER-RECORD,1,5,YYXXX
```

Figure 46 (Part 1 of 3). Master Starting-Point File

```
I,TARMU3,EMP-DATE-JOINED OF EMPLOYEE-MASTER-RECORD,1,5,YYXXX
I,TARMU3,EMP-DATE-MAINTAINED OF EMPLOYEE-MASTER-RECORD,1,5,YYXXX
I,TARMU3,EMP-DATE-TERMINATED OF EMPLOYEE-MASTER-RECORD,1,6,YYXXXX
I,TARMU3,TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF TARDATE-PARAMETERS,1,5,YYXXX
I,TARMU3,TARDATE-DATE-YYMMDD OF TARDATE-DATE-RED2 OF TARDATE-PARAMETERS,1,6,YYXXXX
I,TARMU3,WORK-EIB-DATE OF WORK-VARS,1,4,XXYYXXX
I,TARMU3,WORK-EIB-DATE OF WORK-VARS,1,4,YYYYXXX
I,TARMU3,WORK-EIB-YYDDD OF WORK-EIB-DATE-CHAR OF WORK-VARS,1,5,YYXXX
I,TARMU3,WORK-JOINED-YY OF WORK-JOINED-MMDDYY OF WORK-VARS,1,2,YY
I,TARMU3,WORK-JOINED-YYDDD OF WORK-VARS,1,5,YYXXX
I,TARMU3,WORK-TERMINATED-YYDDD OF WORK-VARS,1,5,YYXXX
I,TARMU3,WORK-TODAYS-MMDDYY OF WORK-VARS,1,8,XX/XX/YY
I,TARMU3,WORK-TODAYS-MMDDYY OF WORK-VARS,1,8,YYYYXXXX
I,TARMU3B,EMP-BIRTH-DATE OF EMPLOYEE-MASTER-RECORD,1,5,YYXXX
I,TARMU3B,EMP-DATE-JOINED OF EMPLOYEE-MASTER-RECORD,1,5,YYXXX
I,TARMU3B,EMP-DATE-MAINTAINED OF EMPLOYEE-MASTER-RECORD,1,5,YYXXX
I,TARMU3B,EMP-DATE-TERMINATED OF EMPLOYEE-MASTER-RECORD,1,6,YYXXXX
I,TARMU3B,INP-EMP-BIRTH-DATE OF INPUT-RECORD,1,8,XX/XX/YY
I,TARMU3B,INP-EMP-BIRTH-DATE OF INPUT-RECORD,1,8,YYYYXXXX
I,TARMU3B,INP-EMP-DATE-JOINED OF INPUT-RECORD,1,8,XX/XX/YY
I,TARMU3B,INP-EMP-DATE-JOINED OF INPUT-RECORD,1,8,YYYYXXXX
I,TARMU3B,INP-EMP-DATE-TERMINATED OF INPUT-RECORD,1,8,XX/XX/YY
I,TARMU3B,INP-EMP-DATE-TERMINATED OF INPUT-RECORD,1,8,YYYYXXXX
I,TARMU3B,OUT-EMP-BIRTH-DATE OF OUTPUT-RECORD,1,8,XX/XX/YY
I,TARMU3B,OUT-EMP-BIRTH-DATE OF OUTPUT-RECORD,1,8,YYYYXXXX
I,TARMU3B,OUT-EMP-DATE-JOINED OF OUTPUT-RECORD,1,8,XX/XX/YY
I,TARMU3B,OUT-EMP-DATE-JOINED OF OUTPUT-RECORD,1,8,YYYYXXXX
I,TARMU3B,OUT-EMP-DATE-TERMINATED OF OUTPUT-RECORD,1,8,XX/XX/YY
I,TARMU3B,OUT-EMP-DATE-TERMINATED OF OUTPUT-RECORD,1,8,YYYYXXXX
I,TARMU3B,TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF TARDATE-PARAMETERS,1,5,YYXXX
I,TARMU3B,TARDATE-DATE-YYMMDD OF TARDATE-DATE-RED2 OF TARDATE-PARAMETERS,1,6,YYXXXX
I,TARMU3B,WORK-EIB-DATE OF WORK-VARS,1,7,YYYYXXX
I,TARMU3B,WORK-EIB-YYDDD OF WORK-EIB-DATE-RED OF WORK-VARS,1,5,YYXXX
I,TARMU3B,WORK-JOINED-YY OF WORK-JOINED-MMDDYY OF WORK-VARS,1,2,YY
I,TARMU3B,WORK-JOINED-YYDDD OF WORK-VARS,1,5,YYXXX
I,TARMU3B,WORK-TERMINATED-YYDDD OF WORK-VARS,1,5,YYXXX
I,TARMU3B,WORK-TODAYS-MMDDYY OF WORK-VARS,1,8,XX/XX/YY
I,TARMU3B,WORK-TODAYS-MMDDYY OF WORK-VARS,1,8,YYYYXXXX
I,TARMU5,RPT-HD1-DATE OF REPORT-HEAD1,1,8,XX/XX/YY
I,TARMU5,RPT-HD1-DATE OF REPORT-HEAD1,1,8,YY/XX/XX
I,TARMU5,RPT-HD1-DD OF RPT-HD1-DATE OF REPORT-HEAD1,1,2,YY
I,TARMU5,RPT-HD1-MM OF RPT-HD1-DATE OF REPORT-HEAD1,1,2,YY
I,TARMU5,RPT-HD1-YY OF RPT-HD1-DATE OF REPORT-HEAD1,1,2,YY
I,TARMU5,TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF TARDATE-PARAMETERS,1,5,YYXXX
I,TARMU5,WORK-DATE-YYMMDD,1,6,YYXXXX
I,TARMU6,EMPLOYEE-DATE-JOINED OF EMPLOYEE-MASTER-RECORD,1,5,YYXXX
I,TARMU6,EMPLOYEE-DATE-TERMINATED OF EMPLOYEE-MASTER-RECORD,1,6,YYXXXX
I,TARMU6,RPT-HD1-DATE OF REPORT-HEAD1,1,8,XX/XX/YY
I,TARMU6,RPT-HD1-DATE OF REPORT-HEAD1,1,8,YY/XX/XX
I,TARMU6,RPT-HD1-DD OF RPT-HD1-DATE OF REPORT-HEAD1,1,2,YY
I,TARMU6,RPT-HD1-MM OF RPT-HD1-DATE OF REPORT-HEAD1,1,2,YY
I,TARMU6,RPT-HD1-YY OF RPT-HD1-DATE OF REPORT-HEAD1,1,2,YY
I,TARMU6,TARDATE-DATE-MMDD OF TARDATE-DATE-RED3 OF TARDATE-PARAMETERS,1,6
I,TARMU6,TARDATE-DATE-YY OF TARDATE-DATE-RED3 OF TARDATE-PARAMETERS,1,2,YY
I,TARMU6,TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF TARDATE-PARAMETERS,1,5,YYXXX
I,TARMU6,WORK-DATE-YYDDD,1,6,YYXXX
I,TARMU6,WORK-DATE-YYMMDD,1,6,YYXXXX
NI,TARDTE3,WRK-DATE-DD OF WRK-DATE OF WORK-DATES
NI,TARDTE3,WRK-DATE-DD OF WRK-DATE-MMXDDXYY OF WORK-DATES
NI,TARDTE3,WRK-DATE-DD OF WRK-DATE-YYXDDXMM OF WORK-DATES
NI,TARDTE3,WRK-DATE-DDD OF WRK-DATE-YYDDD OF WORK-DATES
NI,TARDTE3,WRK-DATE-DDD-NUM OF WRK-DATE-YYDDD-RED OF WORK-DATES
```

Figure 46 (Part 2 of 3). Master Starting-Point File

```
NI,TARDTE3,WRK-DATE-MM OF WRK-DATE OF WORK-DATES
NI,TARDTE3,WRK-DATE-MM OF WRK-DATE-MMXDDXYY OF WORK-DATES
NI,TARDTE3,WRK-DATE-MM OF WRK-DATE-YYXDDXMM OF WORK-DATES
NI,TARDTE3,WRK-DATE-YYDDD-RED OF WORK-DATES
NI,TARDTE3,WRK-DTE-DD OF WRK-DATE-RED OF WORK-DATES
NI,TARDTE3,WRK-DTE-DD-NUM OF WRK-DATE-RED OF WORK-DATES
NI,TARDTE3,WRK-DTE-MM OF WRK-DATE-RED OF WORK-DATES
NI,TARDTE3,WRK-DTE-MM-NUM OF WRK-DATE-RED OF WORK-DATES
NI,TARMU6,TARDATE-DATE-MMDD OF TARDATE-DATE-RED3 OF TARDATE-PARAMETERS
```

Figure 46 (Part 3 of 3). Master Starting-Point File

4.4 Running the Year 2000 Analysis

Once you have your starting-point file, you can perform a Year 2000 analysis by selecting **Option 6** from the MA2000 Main Menu. The Year 2000 analysis panel is displayed as shown in Figure 47. As with the search option, you can specify a domain file in order to limit the amount of programs and data sets to be analyzed. Because our sample application contains only a few programs and jobs, we did not specify a domain.

```
IMXP500------------------------  Year 2000 Analysis  ------------------------
 Command ==>
                                                          Previous panel: PF3

    Specify the following items and press Enter.  An impact analysis
    is done with the Year 2000 analysis option.


 Starting point file: 'ITSORS3.MA2000.SPOINT(YEAR2000)'
 Volume name:         ______   (if not cataloged)


    Domain specification: N (Y/N)  (See note.)
    Data set name:    ____________________________________________________
    Volume name:      ______   (if not cataloged)

    Create a date identification file: Y (Y/N)
    Data set name:    'ITSORS3.MA2000.DIF.YEAR2000'
                      (Allocation is not required. Only PS is allowed.)

    Note: If you specify N at Domain specification, all analysis members
          will be the target for the impact analysis.
```

Figure 47. MA2000 Year 2000 Analysis Panel

As a starting-point file, we specified the consolidated file as discussed in the previous section.

In addition to the summary and detailed reports MA2000 is able to generate a date identification file. This date identification file is used by CCCA as input in order to automatically add Millennium Language Extension syntax to your programs. The data set name you specify for the date identification file must not exist. MA2000 always allocates it as a sequential data set.

The next panel is the MA2000 job execution panel. It is the same as the job execution panel used for the search option. Please see Figure 38 on page 110 for a discussion of its fields.

4.5 Processing the Analysis Output

All the individual and multiple retrieval functions in MA2000 generate various reports as output. Other output depending on the function can be:

- Starting-point file as output of the search function
- Date identification file as output of the Year 2000 analysis function.

MA2000 lists all the functions you perform. You can access these lists by selecting **Option 7** from the MA2000 Main Menu for a list of the results of your individual retrieval functions and **Option 8** for a list of the results of your multiple retrieval function. From the list, you can execute specific actions against the output of the functions that you performed. This is called *retrieval result management*.

The panels for the individual retrieval result management and the multiple retrieval result management look almost the same, with one exception: action **P** to accumulate results is not available on the individual retrieval result management panel. An example of an individual retrieval is the search function. An example of a multiple retrieval is the Year 2000 function.

Figure 48 shows an example of the multiple retrieval result management panel.

```
IMXP801---------------  Retrieval Result Management (Multipl Row 1 to 12 of 12
 Command ===>                                                 SCROLL ===>  CSR
                                                            Previous panel: PF3

   Type one of the following commands next to an item in the list:
      B -View (details)     R -Print (details)    M -Create a domain
      S -View (summary)     T -Print (summary)    X -Create a PC export file
      D -Delete   C -Cancel   P -Accumulate results
      (C and D can be specified for multiple items)       Total count:  012
1   2         3
  JOBID STA Comment                                      Executed       SYS ID
_ 08679 END IMPACT(INFLU) MULTI ITSORS3.MA2000.SPOINT(I 98/11/13 11:28 SYSITSO
_ 10230 END YEAR 2000 ANALYSIS ITSORS3.MA2000.SPOINT(YE 98/11/13 15:33 SYSITSO
_ 17451 END IMPACT(EQUIV) MULTI ITSORS3.MA2000.SPOINT(E 98/11/16 15:24 SYSITSO
_ 21806 END YEAR 2000 ANALYSIS ITSORS3.MA2000.SPOINT(EM 98/11/17 14:50 SYSITSO
_ 21813 END YEAR 2000 ANALYSIS ITSORS3.MA2000.SPOINT(DE 98/11/17 14:52 SYSITSO
_ 21863 END YEAR 2000 ANALYSIS ITSORS3.MA2000.SPOINT(JO 98/11/17 15:04 SYSITSO
_ 21877 END YEAR 2000 ANALYSIS ITSORS3.MA2000.SPOINT(AL 98/11/17 15:08 SYSITSO
_ 21883 END IMPACT(CAUSE) MULTI ITSORS3.MA2000.SPOINT(C 98/11/17 15:10 SYSITSO
_ 21910 END YEAR 2000 ANALYSIS ITSORS3.MA2000.SPOINT(MA 98/11/17 15:16 SYSITSO
_ 09899 END Y2K ANALYSIS WITH SP YEAR2000               98/11/23 16:44 SYSITSO
_ 09911 END Y2K ANALYSIS WITH SP MINI                   98/11/23 16:46 SYSITSO
_ 09924 END Y2K ANALYSIS WITH SP TARDATE                98/11/23 16:47 SYSITSO
****************************** Bottom of data ********************************
```

Figure 48. Retrieval Result Management (Multiple Retrieval) Panel

1 The actions you can perform against the results of the retrieval functions you executed are listed in the first half of the panel:

- **B** and **S** to view the reports
- **R** and **T** to print the reports

- **D** to delete the entry from this list and delete the summary and detail reports as well as all the intermediate data sets from the DASD. The reports have a name like

  ```
  userid.SUMRY.#date.Ttime
  userid.DETAL2.#date.Ttime
  ```

  The intermediate data sets are called

  ```
  userid.CSVWK.#date.Ttime
  userid.DOMAN.#date.Ttime
  userid.PRGIN.#date.Ttime
  userid.RESLT.#date.Ttime
  userid.RESLT2.#date.Ttime
  ```

- **C** to cancel a job while it is still executing. The **STA** column (for *status*) contains **EXE** before a job has ended. If you cancel a job through the **Retrieval Result Management panel**, all the intermediate data and reports created so far are deleted from the DASD.
- **X** to create a file that can be downloaded to the workstation and loaded into a spreadsheet program. The content of this PC export file is the information contained in the detailed report.
- **P** to create intermediate data that can be used by MA2000 to combine the results of several multiple impact analysis runs.
- **M** to create a domain file containing all the programs and jobs that are referenced in the corresponding retrieval output.

We discuss some of these actions in the following sections.

2 The **JOBID** is the actual JES2 job ID of the job you executed to perform the retrieval function. Deleting a job from the spool after it ended has no impact on the corresponding entry in the **Retrieval Result Management** list nor does it remove any report or intermediate file from your DASD. However, if you delete an entry from the **Retrieval Result Management** list, the job output is automatically removed from the spool.

3 MA2000 takes the comment from the job execution panel. See Figure 38 on page 110 for a discussion about these comments.

4.5.1 Summary Reports

You can print (action T) or view (action S) a summary report. If you choose to print it, you have to specify a job card and the SYSOUT class for the print job on the next panel, as shown in Figure 49 on page 125. The print output definition statements are defined in the IMXR1 system parameter. See 3.2.2.4, “Parameter Members for the Retrieval Environment” on page 29 and Table 5 on page 30 for an explanation of this parameter member.

```
IMXP720----------------------  Print (Summary Report)  ----------------------
 Command ===>
                                                          Previous panel: PF3
                                                          Initial panel:  PF4
    To print a summary report, specify the following items and
    press Enter:

    SYSOUT class: *

    Job statement information:

  ==>  //ITSORS3P JOB ,
  ==>  //  MSGCLASS=H,MSGLEVEL=(1,1),NOTIFY=ITSORS3,
  ==>  //  TIME=(,59),REGION=64M
  ==>  //*
  ==>  //*
```

Figure 49. Print Summary Report Panel

Figure 50 on page 126 shows an example of a summary report. It contains the following information:

- Retrieval parameters
  - Type of retrieval
  - User ID performing the retrieval
  - Date and time when the retrieval was performed
  - System
  - Subsystem
  - Domain file used for the retrieval
  - Conditions (for search functions)
  - Starting-point file (for analysis functions)
  - Comment
- Messages
  - Error and other messages issued by the retrieval. In our example we were using a domain that did not contain all the programs referenced in the starting-point file.
- Statistics
- List of retrieved items
  - Program name
  - Item name
  - Attribute of item:

    CHA Alphanumeric variable or nonnumeric literal
    STR Structure

DEC Numeric variable
NUM Numeric literal

– Length of item

```
Year 2000 analysis (Summary)      USER:ITSORS3    DATE:1998/11/23 16:44:19
Information:    1) Target system: SYSITSO
                2) View:          SUBSITSO
                3) Input domain:  ITSORS3.MA2000.DOMAIN(TARMU3)
                4) Condition:     Program name:           ********
                                  File name:              ITSORS3.MA2000.SPOINT(YEAR2000)
                                  Comment:                Y2K ANALYSIS WITH SP YEAR2000
Result:     IMXIM203W    The starting point MEMBER TARMU3B is out of domain.
            IMXIM203W    The starting point MEMBER TARMU5 is out of domain.
            IMXIM203W    The starting point MEMBER TARMU6 is out of domain.
            IMXIM204I    Impact out of domain.  PGM MEMBER: TARMU3B
            IMXIM204I    Impact out of domain.  PGM MEMBER: TARMU6

  Number of Programs:          2 TARDTE3  TARMU3
  Number of COPYs:             2 TARDTE3C TARMU3M
  Number of %INCLUDEs:         0
  Number of SQL COPYs:         0
  Number of SQL INCLs:         0
  Number of EASY MACROs:       0
  Number of JOBs:              3 TARMU3B  TARMU5   TARMU6
  Number of Procedures:        2 TARMU5   TARMU6
  Number of Lines:           133
  Number of Items:            50
  Number of Frequency:       379

Program    Attr Length  Item                                Frequency
----------+---+-----+-----------------------------------+----
TARDTE3    CHA     3 "/"                                     1
           CHA     4 "00"                                    1
           CHA     4 "99"                                    1
           CHA     8 INPUT-DATE                             25
           STR    56 WORK-DATES                              1
           STR     6 WRK-DATE                                6
           STR     8 WRK-DATE-MMXDDXYY                       8
           STR     6 WRK-DATE-RED                            2
           CHA     2 WRK-DATE-YY                            22
           STR     5 WRK-DATE-YYDDD                          8
           STR     8 WRK-DATE-YYXDDXMM                       4
           CHA     2 WRK-DTE-YY                              4
           CHA     2 WRK-YEAR-YY                             3
           STR     4 WRK-YEAR-YYYY                           3
           CHA     4 WRK-YEAR-YYYY-NUM                       4
TARMU3     CHA     8 "02/29/"                                1
           DEC     4 EIBDATE                                 1
           CHA     5 EMP-BIRTH-DATE                          4
           CHA     5 EMP-DATE-JOINED                         4
           CHA     5 EMP-DATE-MAINTAINED                     3
           CHA     6 EMP-DATE-TERMINATED                     6
           DEC     3 EMP-SECURITY-EXP                        3
           STR   200 EMPLOYEE-MASTER-RECORD                 24
           CHA     8 TARDATE-DATE                           138
           STR     8 TARDATE-DATE-RED1                       1
           STR     8 TARDATE-DATE-RED2                       1
           CHA     5 TARDATE-DATE-YYDDD                      7
           CHA     6 TARDATE-DATE-YYMMDD                     4
           STR    54 TARDATE-PARAMETERS                      1
           CHA     8 TARMU3MBDATEI                           4
```

Figure 50 (Part 1 of 2). Summary Report (Year 2000 Analysis)

```
CHA     8 TARMU3MBDATEO                          3
STR   284 TARMU3MI                               1
CHA     8 TARMU3MJDATEI                          5
CHA     8 TARMU3MJDATEO                          4
STR   284 TARMU3MO                              13
CHA     8 TARMU3MSDATEI                          4
CHA     8 TARMU3MSDATEO                          5
CHA     8 TARMU3MTDATEI                          5
CHA     8 TARMU3MTDATEO                          5
DEC     4 WORK-EIB-DATE                          4
STR     7 WORK-EIB-DATE-CHAR                     2
CHA     5 WORK-EIB-YYDDD                         5
CHA     6 WORK-JOINED-MMDD                       3
STR     8 WORK-JOINED-MMDDYY                     7
CHA     2 WORK-JOINED-YY                         3
CHA     5 WORK-JOINED-YYDDD                      5
CHA     5 WORK-TERMINATED-YYDDD                  4
CHA     8 WORK-TODAYS-MMDDYY                     4
STR    44 WORK-VARS                              1
NUM     1 1                                      1
```

Figure 50 (Part 2 of 2). Summary Report (Year 2000 Analysis)

The summary report provides a clear list of all the items that have been retrieved. In case of a Year 2000 analysis, this is the list of the items that could possibly contain a date or somehow influence a date variable. You have to find these items and literals in your program source and check whether there is anything to change in order to make the program Year 2000 ready. If you need more information about the items listed in the summary report, such as the source statements where the items are used, you have to check the detailed report which we discuss in the next section.

The comprehensiveness and correctness of the summary report depends on the quality of your starting-point file.

4.5.2 Detailed Reports

You can print (action R) or view (action B) a detailed report. If you choose to print it, you have to specify a job card and the SYSOUT class for the print job on the next panel, as shown in Figure 51 on page 128. The print output definition statements are defined in the IMXR1 system parameter. See 3.2.2.4, “Parameter Members for the Retrieval Environment” on page 29 and Table 5 on page 30 for an explanation of this parameter member.

You can further choose if you want to print the entire report or just the source statements, and if you want to print it on a printer or write it to a file. If you select the file option, you have to specify the name of the data set you want the report to be written to. The data set name you specify must not exist. MA2000 always allocates it as a sequential data set.

If you write the entire report to a file, MA2000 just copies the report file with the name userid.DETAL2.#date.Ttime to the data set you specify. Therefore writing a report to a file makes sense only if you want to extract the source statements from a report.

```
IMXP710----------------------  Print (Detailed Report)  ----------------------
 Command ===>
                                                          Previous panel: PF3
                                                          Initial panel:  PF4
    To print a detailed report, specify the following items and press Enter.
    If you select Y for the "Output to a file" option, the report is
    written into a file.

      Report type:            1  (1: Whole report  2: Only source statements)
      SYSOUT class:           A
      Output to a file:       Y  (Y/N)
      Output data set name:   'ITSORS3.MA2000.DETAIL.YEAR2000
                           (Allocation is not required.  Only PS is allowed.)

    Job statement information:

  ==>  //ITSORS3P JOB ,
  ==>  //  MSGCLASS=H,MSGLEVEL=(1,1),NOTIFY=ITSORS3,
  ==>  //  TIME=(,59),REGION=64M
  ==>  //*
  ==>  //*
```

Figure 51. Print Detailed Report Panel

Figure 52 on page 129 shows an example of a summary report. It contains the following information:

- Retrieval parameters
  - Type of retrieval
  - UserID performing the retrieval
  - Date and time when the retrieval was performed
  - System
  - Subsystem
  - Domain file used for the retrieval
  - Conditions (for search functions)
  - Starting-point file (for analysis functions)
  - Comment
- Messages
  - Error and other messages issued by the retrieval. In our example, we were using a domain that did not contain all the programs referenced in the starting-point file.
- List of retrieved items: all occurrences of an item are listed. For search functions, the detailed reports do not contain reason codes.
  - JCL member or program name. After a job or program name, there are additional messages that apply to the specific job or program. In our example there are some date format specifications in the starting-point file that caused a conflict during the analysis. In the sample report below, we put some of these error messages in bold.
  - Data set name or program item name

- Attribute of item:

  CHA Alphanumeric variable or nonnumeric literal
  STR Structure
  DEC Numeric variable
  NUM Numeric literal

- Length of item
- Position and length of date within item
- Program line of item occurrence
- Copybook or include where item is defined
- Type of action performed with the specific item in the corresponding source statement. In the heading of the list, this is called *Reason*. But this reason code does not always explain why the item has been retrieved. In other words, it does not say what the relation is between the retrieved item and any of the starting-points. There is one exception: the reason code STA means *starting-point*. So the reason why the specific item is in the list is because it is mentioned as a starting-point in the starting-point file.

  For an explanation of all the reason codes, please see Chapter 5.6.5, "Reason Codes," in the *Maintenance 2000 Guide and Reference Version 1, Release 3*. Note that there are different reason codes with different meanings depending on the type of analysis.

- Program source statements and JCL statements where retrieved items and data sets are referenced. The source and JCL statements are listed together for each member.

```
Year 2000 analysis               USER:ITSORS3 DATE:1998/11/23 16:44:19
Information:    1) Target system: SYSITSO
                2) View:          SUBSITSO
                3) Input domain:  ITSORS3.MA2000.DOMAIN(TARMU3)
                4) Condition:     Program name:           ********
                                  File name:              ITSORS3.MA2000.SPOINT(YEAR2000)
                                  Comment:                Y2K ANALYSIS WITH SP YEAR2000
Result:     IMXIM203W    The starting point MEMBER TARMU3B is out of domain.
            IMXIM203W    The starting point MEMBER TARMU5 is out of domain.
            IMXIM203W    The starting point MEMBER TARMU6 is out of domain.
            IMXIM204I    Impact out of domain.  PGM MEMBER: TARMU3B
            IMXIM204I    Impact out of domain.  PGM MEMBER: TARMU6

Job     (TARMU3B)
Step                   Effects     Impacts
Data set                Attr LengthPos   LengthLine No. PROC       Reason
-----------------------+---+-----+-----+-----+-------+-----------+--------------------------------------------------
TARMU3B
 +RUNIT TARMU3B PGM=TARMU3B
```

Figure 52 (Part 1 of 14). Detailed Report (Year 2000 Analysis)

```
 EMPMAST  ITSORS3.EMP.MASTER.ONLINE
                                  136     5 00310000            STA
                                  141     6 00310000            STA
                                  147     5 00310000            STA
                                  152     5 00310000            STA
                                  157     3 00310000            STA
 DEPTMAST ITSORS3.DEPT.MASTER.ONLINE
                                   35     3 00320000            STA
 INPFILE  ITSORS3.TARMU3B.INPUT
                                  136     8 00330007            STA
                                  144     8 00330007            STA
                                  152     8 00330007            STA
                                  160     8 00330007            STA
         ---------1---------2---------3---------4---------5---------6---------7---------8
        >//EMPMAST  DD  DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR                    00310000<
        >//DEPTMAST DD  DSN=&HLQ..DEPT.MASTER.ONLINE,DISP=SHR                   00320000<
        >//INPFILE  DD  DSN=&HLQ..TARMU3B.INPUT,DISP=SHR                        00330007<
Program     (TARDTE3)
Items                  Effects     Impacts              COPY
Procedure               Attr LengthPos   LengthLine No. %INCLUDE   Reason
----------------------+---+-----+-----+-----+-------+-----------+-----------------------------------------------------
TARDTE3
 IMXIM751W    The formats XX and YY of WRK-DATE-MM are incompatible .
 IMXIM204I    Impact out of domain.  PGM MEMBER: TARMU3B
 IMXIM204I    Impact out of domain.  PGM MEMBER: TARMU5
 IMXIM204I    Impact out of domain.  PGM MEMBER: TARMU6
 IMXIM753W    WRK-DATE-MMXDDXYY is not affected by INPUT-DATE beacuse of having conflicting formats . Line:000071
 IMXIM753W    WRK-DATE-YYXDDXMM is not affected by INPUT-DATE beacuse of having conflicting formats . Line:000076
 IMXIM753W    WRK-DATE-YYDDD is not affected by INPUT-DATE beacuse of having conflicting formats . Line:000081
 IMXIM753W    WRK-DATE is not affected by INPUT-DATE beacuse of having conflicting formats . Line:000085
 IMXIM753W    INPUT-DATE is not affected by WRK-DATE-MMXDDXYY beacuse of having conflicting formats . Line:000150
 IMXIM753W    INPUT-DATE is not affected by WRK-DATE-MMXDDXYY beacuse of having conflicting formats . Line:000071
 IMXIM753W    WRK-DATE-MMXDDXYY is not affected by INPUT-DATE beacuse of having conflicting formats . Line:000150

+TARDTE3
1  WORK-DATES   (YYXXXXYY-YYYY-XX-YY*)
                       STR    56             000001   TARDTE3C   USE
  3  WRK-DATE 1 (YYXXXX)
                       STR     6             000003   TARDTE3C   DCL
     +TARDTE3
                       STR     6     1     2 000003   TARDTE3C   STA
                       STR     6     1     2 000134              INS => INPUT-DATE
                       STR     6     1     2 000085              INS <= INPUT-DATE
                       STR     6     1     6 000085              INS <= INPUT-DATE
                       STR     6     1     6 000134              INS => INPUT-DATE
    5  WRK-DATE-YY 1,1 (YY)
                       CHA     2             000004   TARDTE3C   CHI WRK-DATE OF WORK-DATES
       +TARDTE3
                       CHA     2     1     2 000004   TARDTE3C   STA
                       CHA     2     1     2 000082              INS <= WRK-DATE-YY OF WRK-DATE-YYDDD OF WORK-DATES
                       CHA     2     1     2 000187              INS <= WRK-DATE-YY OF WRK-DATE-YYDDD OF WORK-DATES
                       CHA     2     1     2 000077              INS <= WRK-DATE-YY OF WRK-DATE-YYXDDXMM OF WORK-DA
                       CHA     2     1     2 000147              INS => WRK-DATE-YY OF WRK-DATE-MMXDDXYY OF WORK-DA
                       CHA     2     1     2 000072              INS <= WRK-DATE-YY OF WRK-DATE-MMXDDXYY OF WORK-DA
  3  WRK-DATE-RED 1 (YYXXXX)
                       STR     6             000008   TARDTE3C   DEF => WRK-DATE OF WORK-DATES
     +TARDTE3
                       STR     6     1     2 000008   TARDTE3C   STA
    5  WRK-DTE-YY 1,1 (YY)
                       CHA     2             000009   TARDTE3C   CHI WRK-DATE-RED OF WORK-DATES
       +TARDTE3
                       CHA     2     1     2 000009   TARDTE3C   STA
                       CHA     2     1     2 000167              INS => WRK-YEAR-YY OF WRK-YEAR-YYYY OF WORK-DATES
                       CHA     2     1     2 000109              REF <= "00"
```

Figure 52 (Part 2 of 14). Detailed Report (Year 2000 Analysis)

```
3  WRK-DATE-YYDDD 7 (YYXXX)
                     STR     5             000015   TARDTE3C   DCL
   +TARDTE3
                     STR     5     1     2 000015   TARDTE3C   STA
                     STR     5     1     2 000081              INS <= INPUT-DATE
                     STR     5     1     2 000144              INS => INPUT-DATE
                     STR     5     1     2 000100              REF <= NUMERIC
                     STR     5     1     5 000081              INS <= INPUT-DATE
                     STR     5     1     5 000144              INS => INPUT-DATE
                     STR     5     1     5 000100              REF <= NUMERIC
  5  WRK-DATE-YY 7,1 (YY)
                     CHA     2             000016   TARDTE3C   CHI WRK-DATE-YYDDD OF WORK-DATES
     +TARDTE3
                     CHA     2     1     2 000016   TARDTE3C   STA
                     CHA     2     1     2 000082              INS => WRK-DATE-YY OF WRK-DATE OF WORK-DATES
                     CHA     2     1     2 000187              INS => WRK-DATE-YY OF WRK-DATE OF WORK-DATES
                     CHA     2     1     2 000073              INS <= WRK-DATE-YY OF WRK-DATE-MMXDDXYY OF WORK-DA
                     CHA     2     1     2 000078              INS <= WRK-DATE-YY OF WRK-DATE-YYXDDXMM OF WORK-DA
3  WRK-YEAR-YYYY 12 (YYYY)
                     STR     4             000023   TARDTE3C   CHI WORK-DATES
   +TARDTE3
                     STR     4     1     2 000023   TARDTE3C   STA
                     STR     4     3     2 000023   TARDTE3C   STA
  5  WRK-YEAR-YY 14,3 (YY)
                     CHA     2             000025   TARDTE3C   CHI WRK-YEAR-YYYY OF WORK-DATES
     +TARDTE3
                     CHA     2     1     2 000025   TARDTE3C   STA
                     CHA     2     1     2 000167              INS <= WRK-DTE-YY OF WRK-DATE-RED OF WORK-DATES
3  WRK-YEAR-YYYY-NUM 12 (YYYY)
                     CHA     4             000026   TARDTE3C   DEF => WRK-YEAR-YYYY OF WORK-DATES
   +TARDTE3
                     CHA     4     1     4 000026   TARDTE3C   STA
                     CHA     4     3     2 000168              REF
                     CHA     4     1     2 000168              REF
3  WRK-DATE-MMXDDXYY 17 (XXXXXYYX)
                     STR     8             000029   TARDTE3C   CHI WORK-DATES
   +TARDTE3
                     STR     8     7     2 000029   TARDTE3C   STA
                     STR     8     1     8 000029   TARDTE3C   STA
                     STR     8     7     2 000150              INS => INPUT-DATE
                     STR     8     7     2 000071              INS <= INPUT-DATE
                     STR     8     1     8 000150              INS => INPUT-DATE
                     STR     8     1     8 000071              INS <= INPUT-DATE
                     STR     8     1     8 000148              INS <= "/"
  5  WRK-DATE-YY 23,7 (YY)
                     CHA     2             000034   TARDTE3C   CHI WRK-DATE-MMXDDXYY OF WORK-DATES
     +TARDTE3
                     CHA     2     1     2 000034   TARDTE3C   STA
                     CHA     2     1     2 000147              INS <= WRK-DATE-YY OF WRK-DATE OF WORK-DATES
                     CHA     2     1     2 000072              INS => WRK-DATE-YY OF WRK-DATE OF WORK-DATES
                     CHA     2     1     2 000073              INS => WRK-DATE-YY OF WRK-DATE-YYDDD OF WORK-DATES
3  WRK-DATE-YYXDDXMM 25 (YYXXXXYY)
                     STR     8             000036   TARDTE3C   DCL
   +TARDTE3
                     STR     8     1     2 000036   TARDTE3C   STA
                     STR     8     1     2 000076              INS <= INPUT-DATE
                     STR     8     1     8 000076              INS <= INPUT-DATE
  5  WRK-DATE-YY 25,1 (YY)
                     CHA     2             000037   TARDTE3C   CHI WRK-DATE-YYXDDXMM OF WORK-DATES
     +TARDTE3
                     CHA     2     1     2 000037   TARDTE3C   STA
                     CHA     2     1     2 000077              INS => WRK-DATE-YY OF WRK-DATE OF WORK-DATES
                     CHA     2     1     2 000078              INS => WRK-DATE-YY OF WRK-DATE-YYDDD OF WORK-DATES
```

Figure 52 (Part 3 of 14). Detailed Report (Year 2000 Analysis)

```
1  INPUT-DATE   (YYXXXXXX)
                       CHA     8             000057              DCL
   +TARDTE3
                       CHA     8     1     2 000134              INS <= WRK-DATE OF WORK-DATES
                       CHA     8     1     2 000062              CLL <= TARMU3(1)
                       CHA     8     1     2 000071              INS => WRK-DATE-MMXDDXYY OF WORK-DATES
                       CHA     8     1     2 000076              INS => WRK-DATE-YYXDDXMM OF WORK-DATES
                       CHA     8     1     2 000081              INS => WRK-DATE-YYDDD OF WORK-DATES
                       CHA     8     1     2 000085              INS => WRK-DATE OF WORK-DATES
                       CHA     8     1     2 000144              INS <= WRK-DATE-YYDDD OF WORK-DATES
                       CHA     8     1     2 000150              INS <= WRK-DATE-MMXDDXYY OF WORK-DATES
                       CHA     8     7     2 000150              INS <= WRK-DATE-MMXDDXYY OF WORK-DATES
                       CHA     8     7     2 000062              CLL <= TARMU3(1)
                       CHA     8     7     2 000071              INS => WRK-DATE-MMXDDXYY OF WORK-DATES
                       CHA     8     7     2 000076              INS => WRK-DATE-YYXDDXMM OF WORK-DATES
                       CHA     8     7     2 000081              INS => WRK-DATE-YYDDD OF WORK-DATES
                       CHA     8     7     2 000085              INS => WRK-DATE OF WORK-DATES
                       CHA     8     7     2 000134              INS <= WRK-DATE OF WORK-DATES
                       CHA     8     7     2 000144              INS <= WRK-DATE-YYDDD OF WORK-DATES
                       CHA     8     1     8 000062              PRC TARMU3(1)
                       CHA     8     1     8 000071              INS => WRK-DATE-MMXDDXYY OF WORK-DATES
                       CHA     8     1     8 000076              INS => WRK-DATE-YYXDDXMM OF WORK-DATES
                       CHA     8     1     8 000081              INS => WRK-DATE-YYDDD OF WORK-DATES
                       CHA     8     1     8 000085              INS => WRK-DATE OF WORK-DATES
                       CHA     8     1     8 000134              INS <= WRK-DATE OF WORK-DATES
                       CHA     8     1     8 000144              INS <= WRK-DATE-YYDDD OF WORK-DATES
                       CHA     8     1     8 000150              INS <= WRK-DATE-MMXDDXYY OF WORK-DATES
++CN "/"
     +TARDTE3
                       CHA     3     1     1 000148              INS => WRK-DATE-MMXDDXYY OF WORK-DATES
++CN "00"
     +TARDTE3
                       CHA     4     1     2 000109              REF => WRK-DTE-YY OF WRK-DATE-RED OF WORK-DATES
++CN "99"
     +TARDTE3
                       CHA     4     1     2 000109              REF => WRK-DTE-YY OF WRK-DATE-RED OF WORK-DATES
          ---------1---------2---------3---------4---------5---------6---------7---------8
 TARDTE3C>000001 01  WORK-DATES.                                                  00000001<
 TARDTE3C>000003     03  WRK-DATE.                                               00000003<
 TARDTE3C>000004         05  WRK-DATE-YY           PIC XX.                      00000004<
 TARDTE3C>000008     03  WRK-DATE-RED REDEFINES WRK-DATE.                       00000008<
 TARDTE3C>000009         05  WRK-DTE-YY            PIC XX.                      00000009<
 TARDTE3C>000015     03  WRK-DATE-YYDDD.                                        00000015<
 TARDTE3C>000016         05  WRK-DATE-YY           PIC XX.                      00000016<
 TARDTE3C>000023     03  WRK-YEAR-YYYY.                                         00000023<
 TARDTE3C>000025         05  WRK-YEAR-YY           PIC XX.                      00000025<
 TARDTE3C>000026     03  WRK-YEAR-YYYY-NUM REDEFINES WRK-YEAR-YYYY PIC 9(4).    00000026<
 TARDTE3C>000029     03  WRK-DATE-MMXDDXYY.                                     00000029<
 TARDTE3C>000034         05  WRK-DATE-YY           PIC XX.                      00000034<
 TARDTE3C>000036     03  WRK-DATE-YYXDDXMM.                                     00000036<
 TARDTE3C>000037         05  WRK-DATE-YY           PIC XX.                      00000037<
         >000057 01  INPUT-DATE                   PIC X(8).                    00000057<
         >000062 PROCEDURE DIVISION USING INPUT-DATE                            00000062<
         >000063                          INPUT-FORMAT                          00000063<
         >000064                          OUTPUT-FORMAT                         00000064<
         >000065                          MSG.                                  00000065<
         >000071         MOVE INPUT-DATE TO WRK-DATE-MMXDDXYY                   00000071<
         >000072         MOVE CORRESPONDING WRK-DATE-MMXDDXYY TO WRK-DATE       00000072<
         >000073         MOVE CORRESPONDING WRK-DATE-MMXDDXYY TO WRK-DATE-YYDDD 00000073<
         >000076            MOVE INPUT-DATE TO WRK-DATE-YYXDDXMM                00000076<
         >000077            MOVE CORRESPONDING WRK-DATE-YYXDDXMM TO WRK-DATE    00000077<
         >000078            MOVE CORRESPONDING WRK-DATE-YYXDDXMM TO WRK-DATE-YYDDD00000078<
         >000081               MOVE INPUT-DATE TO WRK-DATE-YYDDD                00000081<
         >000082               MOVE CORRESPONDING WRK-DATE-YYDDD TO WRK-DATE    00000082<
         >000085                  MOVE INPUT-DATE TO WRK-DATE                   00000085<
```

Figure 52 (Part 4 of 14). Detailed Report (Year 2000 Analysis)

```
         >000100       IF  WRK-DATE-YYDDD IS NUMERIC THEN                          00000100<
         >000109       IF  WRK-DTE-YY < "00" OR > "99" THEN                        00000109<
         >000134         MOVE WRK-DATE    TO INPUT-DATE                            00000134<
         >000144         MOVE WRK-DATE-YYDDD TO INPUT-DATE                         00000144<
         >000147         MOVE CORRESPONDING WRK-DATE TO WRK-DATE-MMXDDXYY           00000147<
         >000148         MOVE "/"            TO WRK-DATE-MMXDDXYY(3:1)             00000148<
         >000149                                WRK-DATE-MMXDDXYY(6:1)             00000149<
         >000150         MOVE WRK-DATE-MMXDDXYY TO INPUT-DATE                      00000150<
         >000167     MOVE WRK-DTE-YY  TO WRK-YEAR-YY                               00000167<
         >000168     DIVIDE WRK-YEAR-YYYY-NUM BY 4 GIVING WHOLE-NUMBER             00000168<
         >000169                                  REMAINDER WRK-LEAP               00000169<
         >000187         MOVE CORRESPONDING WRK-DATE-YYDDD TO WRK-DATE             00000187<
Job     (TARMU5)
Step                   Effects     Impacts
Data set                Attr LengthPos   LengthLine No. PROC       Reason
-----------------------+---+-----+-----+-----+-------+-----------+-------------------------------------------------
TARMU5
 IMXIM204I    Impact out of domain.  PGM MEMBER: TARMU5

 +RUNIT SORT PGM=ICEMAN
 SORTIN   ITSORS3.EMP.MASTER.ONLINE
                                    136     5 00270000 TARMU5     STA
                                    141     6 00270000 TARMU5     STA
                                    147     5 00270000 TARMU5     STA
                                    152     5 00270000 TARMU5     STA
                                    157     3 00270000 TARMU5     STA
                                    157     1 00070022            REF DD:SORTIN
                                    158     2 00070022            REF DD:SORTIN
 SORTOUT  &&SORTOUT
                                     37     3 00280000 TARMU5     SET => AT 00090022
                                     37     3 00090022            SET DD:SORTOUT => DD:SORTIN
          ---------1---------2---------3---------4---------5---------6---------7---------8
 TARMU5  >//SORTIN   DD  DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR                      00270000<
 TARMU5  >//SORTOUT  DD  DSN=&&SORTOUT,DISP=(NEW,PASS),SPACE=(CYL,(1,1)),          00280000<
 TARMU5  >//             DCB=(DSORG=PS,RECFM=FB,LRECL=200,BLKSIZE=4000)           00290000<
         >   SORT FIELDS=(157,1,Y2D,A,158,2,BI,A)                                 00070022<
         >   OUTREC FIELDS=(1,6,11,30,157,3,161X)                                 00090022<
Job     (TARMU5)
Step                   Effects     Impacts
Data set                Attr LengthPos   LengthLine No. PROC       Reason
-----------------------+---+-----+-----+-----+-------+-----------+-------------------------------------------------
TARMU5
 +RUNIT BEFORE PGM=IDCAMS
 EMPMAST  &&SORTOUT
                                     37     3 00400000 TARMU5     USD STEP:RUNIT DD:SORTOUT
          ---------1---------2---------3---------4---------5---------6---------7---------8
 TARMU5  >//EMPMAST  DD  DSN=&&SORTOUT,DISP=(OLD,PASS)                            00400000<
Job     (TARMU5)
Step                   Effects     Impacts
Data set                Attr LengthPos   LengthLine No. PROC       Reason
-----------------------+---+-----+-----+-----+-------+-----------+-------------------------------------------------
TARMU5
 +RUNIT TARMU5 PGM=TARMU5
 EMPMAST  &&SORTOUT
                                     37     3 00520000 TARMU5     USD STEP:RUNIT DD:SORTOUT
          ---------1---------2---------3---------4---------5---------6---------7---------8
 TARMU5  >//EMPMAST  DD  DSN=&&SORTOUT,DISP=(OLD,DELETE)                          00520000<
```

Figure 52 (Part 5 of 14). Detailed Report (Year 2000 Analysis)

```
Job     (TARMU6)
Step                   Effects     Impacts
Data set                Attr LengthPos   LengthLine No. PROC       Reason
-----------------------+---+-----+-----+-----+-------+-----------+-----------------------------------------------------
TARMU6
 +RUNIT TARMU6 PGM=TARMU6
 EMPMAST  ITSORS3.EMP.MASTER.ONLINE
                                   136     5 00360000 TARMU6     STA
                                   141     6 00360000 TARMU6     STA
                                   147     5 00360000 TARMU6     STA
                                   152     5 00360000 TARMU6     STA
                                   157     3 00360000 TARMU6     STA
         ---------1---------2---------3---------4---------5---------6---------7---------8
 TARMU6  >//EMPMAST  DD  DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR                    00360000<
Program     (TARMU3)
Items                  Effects     Impacts              COPY
Procedure               Attr LengthPos   LengthLine No. %INCLUDE   Reason
-----------------------+---+-----+-----+-----+-------+-----------+-----------------------------------------------------
TARMU3
 IMXIM753W    WORK-EIB-DATE-CHAR is not affected by WORK-EIB-DATE beacuse of having conflicting formats . Line:015100
 IMXIM751W    The formats XXXXYY and YYYYXXXX of WORK-TODAYS-MMDDYY are incompatible .
 IMXIM753W    WORK-TODAYS-MMDDYY is not affected by TARDATE-DATE beacuse of having conflicting formats . Line:015900
 IMXIM753W    TARMU3MJDATEO is not affected by WORK-TODAYS-MMDDYY beacuse of having conflicting formats . Line:036400
 IMXIM753W    TARDATE-DATE is not affected by WORK-TODAYS-MMDDYY beacuse of having conflicting formats . Line:015900
 IMXIM753W    WORK-EIB-YYDDD is not affected by TARDATE-DATE beacuse of having conflicting formats . Line:015200

+TARMU3
1  EMPLOYEE-MASTER-RECORD   (-YYXXXYYXXXXYYXXXYY*)
                         STR   200             003400              USE
   +TARMU3
                         STR   200   136     5 050700              DCW CFILE:EMPMAST
                         STR   200   136     5 051600              DCW CFILE:EMPMAST
                         STR   200   136     5 045800              INS <= SPACES
                         STR   200   152     5 050700              DCW CFILE:EMPMAST
                         STR   200   152     5 051600              DCW CFILE:EMPMAST
                         STR   200   152     5 045800              INS <= SPACES
                         STR   200   157     3 050700              DCW CFILE:EMPMAST
                         STR   200   157     3 051600              DCW CFILE:EMPMAST
                         STR   200   157     3 045800              INS <= SPACES
                         STR   200   141     6 050700              DCW CFILE:EMPMAST
                         STR   200   141     6 051600              DCW CFILE:EMPMAST
                         STR   200   141     6 045800              INS <= SPACES
                         STR   200   136     5 022300              DCR CFILE:EMPMAST
                         STR   200   136     5 045100              DCR CFILE:EMPMAST
                         STR   200   141     6 022300              DCR CFILE:EMPMAST
                         STR   200   141     6 045100              DCR CFILE:EMPMAST
                         STR   200   147     5 050700              DCW CFILE:EMPMAST
                         STR   200   147     5 051600              DCW CFILE:EMPMAST
                         STR   200   147     5 045800              INS <= SPACES
                         STR   200   147     5 022300              DCR CFILE:EMPMAST
                         STR   200   147     5 045100              DCR CFILE:EMPMAST
                         STR   200   152     5 022300              DCR CFILE:EMPMAST
                         STR   200   152     5 045100              DCR CFILE:EMPMAST
  3  EMP-DATE-JOINED 136 (YYXXX)
                         CHA     5             004400              DCL
     +TARMU3
                         CHA     5     1     5 004400              STA
                         CHA     5     1     5 025400              INS => TARDATE-DATE OF TARDATE-PARAMETERS
                         CHA     5     1     5 047300              INS <= TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF
```

Figure 52 (Part 6 of 14). Detailed Report (Year 2000 Analysis)

```
  3  EMP-DATE-TERMINATED 141 (YYXXXX)
                       CHA     6             004600             DCL
     +TARMU3
                       CHA     6     1     6 004600             STA
                       CHA     6     1     6 049100             INS <= TARDATE-DATE-YYMMDD OF TARDATE-DATE-RED2 OF
                       CHA     6     1     6 027300             INS => TARDATE-DATE OF TARDATE-PARAMETERS
                       CHA     6     1     6 049300             INS <= ZEROS
                       CHA     6     1     6 027200             REF <= ZEROS
  3  EMP-DATE-MAINTAINED 147 (YYXXX)
                       CHA     5             004800             DCL
     +TARMU3
                       CHA     5     1     5 004800             STA
                       CHA     5     1     5 050400             INS <= WORK-EIB-YYDDD OF WORK-EIB-DATE-CHAR OF WOR
  3  EMP-BIRTH-DATE 152 (YYXXX)
                       CHA     5             005000             DCL
     +TARMU3
                       CHA     5     1     5 005000             STA
                       CHA     5     1     5 048100             INS <= TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF
                       CHA     5     1     5 026300             INS => TARDATE-DATE OF TARDATE-PARAMETERS
  3  EMP-SECURITY-EXP 157 (YYXXXXYY)
                       DEC     3             005200             DCL
     +TARMU3
                       DEC     3     1     3 050200             INS <= TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF
                       DEC     3     1     3 028500             INS => TARDATE-DATE OF TARDATE-PARAMETERS
1  WORK-VARS   (-XXYYXXX-)
                       STR    44             009800             USE
  3  WORK-TODAYS-MMDDYY 1 (XXXXYYXX)
                       CHA     8             009900             DCL
     +TARMU3
                       CHA     8     1     8 009900             STA
                       CHA     8     1     8 036400             INS => TARMU3MJDATEO OF TARMU3MO
                       CHA     8     1     8 015900             INS <= TARDATE-DATE OF TARDATE-PARAMETERS
  3  WORK-EIB-DATE 11 (XXYYXXX)
                       DEC     4             010100             DCL
     +TARMU3
                       DEC     4     1     4 015000             INS <= EIBDATE
                       DEC     4     1     4 015100             INS => WORK-EIB-DATE-CHAR OF WORK-VARS
                       DEC     4     1     4 010100             STA
  3  WORK-EIB-DATE-CHAR 15 (YYYYXXX)
                       STR     7             010200             CHI WORK-VARS
     +TARMU3
                       STR     7     3     5 015100             INS <= WORK-EIB-DATE OF WORK-VARS
    5  WORK-EIB-YYDDD 17,3 (YYXXX)
                       CHA     5             010500             DCL
       +TARMU3
                       CHA     5     1     5 010500             STA
                       CHA     5     1     5 050400             INS => EMP-DATE-MAINTAINED OF EMPLOYEE-MASTER-RECO
                       CHA     5     1     5 015200             INS => TARDATE-DATE OF TARDATE-PARAMETERS
                       CHA     5     1     5 036600             INS => WORK-JOINED-YYDDD OF WORK-VARS
  3  WORK-JOINED-YYDDD 22 (YYXXX)
                       CHA     5             010600             DCL
     +TARMU3
                       CHA     5     1     5 010600             STA
                       CHA     5     1     5 036000             INS <= TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF
                       CHA     5     1     5 036600             INS <= WORK-EIB-YYDDD OF WORK-EIB-DATE-CHAR OF WOR
                       CHA     5     1     5 041200             REF => WORK-TERMINATED-YYDDD OF WORK-VARS
  3  WORK-JOINED-MMDDYY 27 (XXXXXXYY)
                       STR     8             010700             CHI WORK-VARS
     +TARMU3
                       STR     8     1     8 036100             INS <= TARMU3MJDATEI OF TARMU3MI
                       STR     8     1     8 044500             INS => TARMU3MSDATEO OF TARMU3MO
                       STR     8     1     8 036400             INS <= WORK-TODAYS-MMDDYY OF WORK-VARS
                       STR     8     7     2 044500             INS => TARMU3MSDATEO OF TARMU3MO
                       STR     8     7     2 036100             INS <= TARMU3MJDATEI OF TARMU3MI
                       STR     8     7     2 036400             INS <= WORK-TODAYS-MMDDYY OF WORK-VARS
```

Figure 52 (Part 7 of 14). Detailed Report (Year 2000 Analysis)

```
    5  WORK-JOINED-MMDD 27,1
                         CHA     6             010800             CHI WORK-JOINED-MMDDYY OF WORK-VARS
       +TARMU3
                         CHA     6     1     6 044300             INS <= "02/28/"
                         CHA     6     1     6 044200             REF <= "02/29/"
    5  WORK-JOINED-YY 33,7 (YY)
                         CHA     2             010900             CHI WORK-JOINED-MMDDYY OF WORK-VARS
       +TARMU3
                         CHA     2     1     2 010900             STA
                         CHA     2     1     2 044100             INS UPDATED
  3  WORK-TERMINATED-YYDDD 40 (YYXXX)
                         CHA     5             011100             DCL
     +TARMU3
                         CHA     5     1     5 011100             STA
                         CHA     5     1     5 041100             INS <= TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF
                         CHA     5     1     5 041200             REF <= WORK-JOINED-YYDDD OF WORK-VARS
1  TARDATE-PARAMETERS   (YYXXX-)
                         STR    54             011800             USE
  3  TARDATE-DATE 1 (YYXXXXYY)
                         CHA     8             011900             CHI TARDATE-PARAMETERS
     +TARMU3
                         CHA     8     1     8 025400             INS <= EMP-DATE-JOINED OF EMPLOYEE-MASTER-RECORD
                         CHA     8     1     8 015500             CLL TARDTE3(1)
                         CHA     8     1     8 025700             CLL TARDTE3(1)
                         CHA     8     1     8 026600             CLL TARDTE3(1)
                         CHA     8     1     8 027600             CLL TARDTE3(1)
                         CHA     8     1     8 028800             CLL TARDTE3(1)
                         CHA     8     1     8 034800             CLL TARDTE3(1)
                         CHA     8     1     8 037300             CLL TARDTE3(1)
                         CHA     8     1     8 039800             CLL TARDTE3(1)
                         CHA     8     1     8 042800             CLL TARDTE3(1)
                         CHA     8     1     8 046900             CLL TARDTE3(1)
                         CHA     8     1     8 047700             CLL TARDTE3(1)
                         CHA     8     1     8 048700             CLL TARDTE3(1)
                         CHA     8     1     8 049800             CLL TARDTE3(1)
                         CHA     8     1     8 015900             INS => WORK-TODAYS-MMDDYY OF WORK-VARS
                         CHA     8     1     8 026100             INS => TARMU3MJDATEO OF TARMU3MO
                         CHA     8     1     8 027000             INS => TARMU3MBDATEO OF TARMU3MO
                         CHA     8     1     8 028000             INS => TARMU3MTDATEO OF TARMU3MO
                         CHA     8     1     8 029200             INS => TARMU3MSDATEO OF TARMU3MO
                         CHA     8     1     8 015200             INS <= WORK-EIB-YYDDD OF WORK-EIB-DATE-CHAR OF WOR
                         CHA     8     1     8 026300             INS <= EMP-BIRTH-DATE OF EMPLOYEE-MASTER-RECORD
                         CHA     8     1     8 027300             INS <= EMP-DATE-TERMINATED OF EMPLOYEE-MASTER-RECO
                         CHA     8     1     8 028500             INS <= EMP-SECURITY-EXP OF EMPLOYEE-MASTER-RECORD
                         CHA     8     1     8 034500             INS <= TARMU3MJDATEI OF TARMU3MI
                         CHA     8     1     8 037000             INS <= TARMU3MBDATEI OF TARMU3MI
                         CHA     8     1     8 039500             INS <= TARMU3MTDATEI OF TARMU3MI
                         CHA     8     1     8 042500             INS <= TARMU3MSDATEI OF TARMU3MI
                         CHA     8     1     8 046600             INS <= TARMU3MJDATEI OF TARMU3MI
                         CHA     8     1     8 047400             INS <= TARMU3MBDATEI OF TARMU3MI
                         CHA     8     1     8 048400             INS <= TARMU3MTDATEI OF TARMU3MI
                         CHA     8     1     8 049500             INS <= TARMU3MSDATEI OF TARMU3MI
                         CHA     8     1     5 015500             CLL TARDTE3(1)
                         CHA     8     1     5 025700             CLL TARDTE3(1)
                         CHA     8     1     5 026600             CLL TARDTE3(1)
                         CHA     8     1     5 027600             CLL TARDTE3(1)
                         CHA     8     1     5 028800             CLL TARDTE3(1)
                         CHA     8     1     5 034800             CLL TARDTE3(1)
                         CHA     8     1     5 037300             CLL TARDTE3(1)
                         CHA     8     1     5 039800             CLL TARDTE3(1)
                         CHA     8     1     5 042800             CLL TARDTE3(1)
```

Figure 52 (Part 8 of 14). Detailed Report (Year 2000 Analysis)

```
CHA     8     1     5 046900             CLL TARDTE3(1)
CHA     8     1     5 047700             CLL TARDTE3(1)
CHA     8     1     5 048700             CLL TARDTE3(1)
CHA     8     1     5 049800             CLL TARDTE3(1)
CHA     8     1     5 015900             INS => WORK-TODAYS-MMDDYY OF WORK-VARS
CHA     8     1     5 026100             INS => TARMU3MJDATEO OF TARMU3MO
CHA     8     1     5 027000             INS => TARMU3MBDATEO OF TARMU3MO
CHA     8     1     5 028000             INS => TARMU3MTDATEO OF TARMU3MO
CHA     8     1     5 029200             INS => TARMU3MSDATEO OF TARMU3MO
CHA     8     1     5 015200             INS <= WORK-EIB-YYDDD OF WORK-EIB-DATE-CHAR OF WOR
CHA     8     1     5 025400             INS <= EMP-DATE-JOINED OF EMPLOYEE-MASTER-RECORD
CHA     8     1     5 026300             INS <= EMP-BIRTH-DATE OF EMPLOYEE-MASTER-RECORD
CHA     8     1     5 027300             INS <= EMP-DATE-TERMINATED OF EMPLOYEE-MASTER-RECO
CHA     8     1     5 028500             INS <= EMP-SECURITY-EXP OF EMPLOYEE-MASTER-RECORD
CHA     8     1     5 034500             INS <= TARMU3MJDATEI OF TARMU3MI
CHA     8     1     5 037000             INS <= TARMU3MBDATEI OF TARMU3MI
CHA     8     1     5 039500             INS <= TARMU3MTDATEI OF TARMU3MI
CHA     8     1     5 042500             INS <= TARMU3MSDATEI OF TARMU3MI
CHA     8     1     5 046600             INS <= TARMU3MJDATEI OF TARMU3MI
CHA     8     1     5 047400             INS <= TARMU3MBDATEI OF TARMU3MI
CHA     8     1     5 048400             INS <= TARMU3MTDATEI OF TARMU3MI
CHA     8     1     5 049500             INS <= TARMU3MSDATEI OF TARMU3MI
CHA     8     1     6 015500             CLL TARDTE3(1)
CHA     8     1     6 025700             CLL TARDTE3(1)
CHA     8     1     6 026600             CLL TARDTE3(1)
CHA     8     1     6 027600             CLL TARDTE3(1)
CHA     8     1     6 028800             CLL TARDTE3(1)
CHA     8     1     6 034800             CLL TARDTE3(1)
CHA     8     1     6 037300             CLL TARDTE3(1)
CHA     8     1     6 039800             CLL TARDTE3(1)
CHA     8     1     6 042800             CLL TARDTE3(1)
CHA     8     1     6 046900             CLL TARDTE3(1)
CHA     8     1     6 047700             CLL TARDTE3(1)
CHA     8     1     6 048700             CLL TARDTE3(1)
CHA     8     1     6 049800             CLL TARDTE3(1)
CHA     8     1     6 015900             INS => WORK-TODAYS-MMDDYY OF WORK-VARS
CHA     8     1     6 026100             INS => TARMU3MJDATEO OF TARMU3MO
CHA     8     1     6 027000             INS => TARMU3MBDATEO OF TARMU3MO
CHA     8     1     6 028000             INS => TARMU3MTDATEO OF TARMU3MO
CHA     8     1     6 029200             INS => TARMU3MSDATEO OF TARMU3MO
CHA     8     1     6 015200             INS <= WORK-EIB-YYDDD OF WORK-EIB-DATE-CHAR OF WOR
CHA     8     1     6 025400             INS <= EMP-DATE-JOINED OF EMPLOYEE-MASTER-RECORD
CHA     8     1     6 026300             INS <= EMP-BIRTH-DATE OF EMPLOYEE-MASTER-RECORD
CHA     8     1     6 027300             INS <= EMP-DATE-TERMINATED OF EMPLOYEE-MASTER-RECO
CHA     8     1     6 028500             INS <= EMP-SECURITY-EXP OF EMPLOYEE-MASTER-RECORD
CHA     8     1     6 034500             INS <= TARMU3MJDATEI OF TARMU3MI
CHA     8     1     6 037000             INS <= TARMU3MBDATEI OF TARMU3MI
CHA     8     1     6 039500             INS <= TARMU3MTDATEI OF TARMU3MI
CHA     8     1     6 042500             INS <= TARMU3MSDATEI OF TARMU3MI
CHA     8     1     6 046600             INS <= TARMU3MJDATEI OF TARMU3MI
CHA     8     1     6 047400             INS <= TARMU3MBDATEI OF TARMU3MI
CHA     8     1     6 048400             INS <= TARMU3MTDATEI OF TARMU3MI
CHA     8     1     6 049500             INS <= TARMU3MSDATEI OF TARMU3MI
CHA     8     1     2 015500             PRC TARDTE3(1)
CHA     8     1     2 025700             PRC TARDTE3(1)
CHA     8     1     2 026600             PRC TARDTE3(1)
CHA     8     1     2 027600             PRC TARDTE3(1)
```

Figure 52 (Part 9 of 14). Detailed Report (Year 2000 Analysis)

```
                        CHA     8     1     2 028800              PRC TARDTE3(1)
                        CHA     8     1     2 034800              PRC TARDTE3(1)
                        CHA     8     1     2 037300              PRC TARDTE3(1)
                        CHA     8     1     2 039800              PRC TARDTE3(1)
                        CHA     8     1     2 042800              PRC TARDTE3(1)
                        CHA     8     1     2 046900              PRC TARDTE3(1)
                        CHA     8     1     2 047700              PRC TARDTE3(1)
                        CHA     8     1     2 048700              PRC TARDTE3(1)
                        CHA     8     1     2 049800              PRC TARDTE3(1)
                        CHA     8     7     2 015500              PRC TARDTE3(1)
                        CHA     8     7     2 025700              CLL TARDTE3(1)
                        CHA     8     7     2 026600              CLL TARDTE3(1)
                        CHA     8     7     2 027600              CLL TARDTE3(1)
                        CHA     8     7     2 028800              CLL TARDTE3(1)
                        CHA     8     7     2 034800              CLL TARDTE3(1)
                        CHA     8     7     2 037300              CLL TARDTE3(1)
                        CHA     8     7     2 039800              CLL TARDTE3(1)
                        CHA     8     7     2 042800              CLL TARDTE3(1)
                        CHA     8     7     2 046900              CLL TARDTE3(1)
                        CHA     8     7     2 047700              CLL TARDTE3(1)
                        CHA     8     7     2 048700              CLL TARDTE3(1)
                        CHA     8     7     2 049800              CLL TARDTE3(1)
                        CHA     8     7     2 015900              INS => WORK-TODAYS-MMDDYY OF WORK-VARS
                        CHA     8     7     2 026100              INS => TARMU3MJDATEO OF TARMU3MO
                        CHA     8     7     2 027000              INS => TARMU3MBDATEO OF TARMU3MO
                        CHA     8     7     2 028000              INS => TARMU3MTDATEO OF TARMU3MO
                        CHA     8     7     2 029200              INS => TARMU3MSDATEO OF TARMU3MO
                        CHA     8     7     2 015200              INS <= WORK-EIB-YYDDD OF WORK-EIB-DATE-CHAR OF WOR
                        CHA     8     7     2 025400              INS <= EMP-DATE-JOINED OF EMPLOYEE-MASTER-RECORD
                        CHA     8     7     2 026300              INS <= EMP-BIRTH-DATE OF EMPLOYEE-MASTER-RECORD
                        CHA     8     7     2 027300              INS <= EMP-DATE-TERMINATED OF EMPLOYEE-MASTER-RECO
                        CHA     8     7     2 028500              INS <= EMP-SECURITY-EXP OF EMPLOYEE-MASTER-RECORD
                        CHA     8     7     2 034500              INS <= TARMU3MJDATEI OF TARMU3MI
                        CHA     8     7     2 037000              INS <= TARMU3MBDATEI OF TARMU3MI
                        CHA     8     7     2 039500              INS <= TARMU3MTDATEI OF TARMU3MI
                        CHA     8     7     2 042500              INS <= TARMU3MSDATEI OF TARMU3MI
                        CHA     8     7     2 046600              INS <= TARMU3MJDATEI OF TARMU3MI
                        CHA     8     7     2 047400              INS <= TARMU3MBDATEI OF TARMU3MI
                        CHA     8     7     2 048400              INS <= TARMU3MTDATEI OF TARMU3MI
                        CHA     8     7     2 049500              INS <= TARMU3MSDATEI OF TARMU3MI
  3  TARDATE-DATE-RED1 1 (YYXXXXXX)
                        STR     8             012000              DEF => TARDATE-DATE OF TARDATE-PARAMETERS
    5  TARDATE-DATE-YYDDD 1,1 (YYXXX)
                        CHA     5             012100              CHI TARDATE-DATE-RED1 OF TARDATE-PARAMETERS
       +TARMU3
                        CHA     5     1     5 012100              STA
                        CHA     5     1     5 036000              INS => WORK-JOINED-YYDDD OF WORK-VARS
                        CHA     5     1     5 041100              INS => WORK-TERMINATED-YYDDD OF WORK-VARS
                        CHA     5     1     5 047300              INS => EMP-DATE-JOINED OF EMPLOYEE-MASTER-RECORD
                        CHA     5     1     5 048100              INS => EMP-BIRTH-DATE OF EMPLOYEE-MASTER-RECORD
                        CHA     5     1     5 050200              INS => EMP-SECURITY-EXP OF EMPLOYEE-MASTER-RECORD
  3  TARDATE-DATE-RED2 1 (YYXXXXXX)
                        STR     8             012300              DEF => TARDATE-DATE OF TARDATE-PARAMETERS
    5  TARDATE-DATE-YYMMDD 1,1 (YYXXXX)
                        CHA     6             012400              CHI TARDATE-DATE-RED2 OF TARDATE-PARAMETERS
       +TARMU3
                        CHA     6     1     6 012400              STA
                        CHA     6     1     6 049100              INS => EMP-DATE-TERMINATED OF EMPLOYEE-MASTER-RECO
                        CHA     6     1     5 049100              INS => EMP-DATE-TERMINATED OF EMPLOYEE-MASTER-RECO
```

Figure 52 (Part 10 of 14). Detailed Report (Year 2000 Analysis)

```
1  TARMU3MI
                        STR   284             000001   TARMU3M    DEF <= TARMU3MO
  2  TARMU3MJDATEI 180 (YYXXXXXX)
                        CHA     8             000050   TARMU3M    CHI TARMU3MI
     +TARMU3
                        CHA     8     1     8 034500              INS => TARDATE-DATE OF TARDATE-PARAMETERS
                        CHA     8     1     8 036100              INS => WORK-JOINED-MMDDYY OF WORK-VARS
                        CHA     8     1     8 046600              INS => TARDATE-DATE OF TARDATE-PARAMETERS
                        CHA     8     1     8 034400              REF <= SPACES
  2  TARMU3MBDATEI 192 (YYXXXXXX)
                        CHA     8             000056   TARMU3M    CHI TARMU3MI
     +TARMU3
                        CHA     8     1     8 037000              INS => TARDATE-DATE OF TARDATE-PARAMETERS
                        CHA     8     1     8 047400              INS => TARDATE-DATE OF TARDATE-PARAMETERS
                        CHA     8     1     8 036900              REF <= SPACES
  2  TARMU3MTDATEI 204 (YYXXXXXX)
                        CHA     8             000062   TARMU3M    CHI TARMU3MI
     +TARMU3
                        CHA     8     1     8 039500              INS => TARDATE-DATE OF TARDATE-PARAMETERS
                        CHA     8     1     8 048400              INS => TARDATE-DATE OF TARDATE-PARAMETERS
                        CHA     8     1     8 039400              REF <= SPACES
                        CHA     8     1     8 048300              REF <= SPACES
  2  TARMU3MSDATEI 216 (YYXXXXXX)
                        CHA     8             000068   TARMU3M    CHI TARMU3MI
     +TARMU3
                        CHA     8     1     8 042500              INS => TARDATE-DATE OF TARDATE-PARAMETERS
                        CHA     8     1     8 049500              INS => TARDATE-DATE OF TARDATE-PARAMETERS
                        CHA     8     1     8 042400              REF <= SPACES
1  TARMU3MO   (-YYXXXX-YYXXXX-YYXX*)
                        STR   284             000075   TARMU3M    USE
   +TARMU3
                        STR   284   180     8 021500              DCW MAP:TARMU3M
                        STR   284   180     8 016100              INS <= LOW-VALUES
                        STR   284   180     8 022900              INS <= LOW-VALUES
                        STR   284   216     8 021500              DCW MAP:TARMU3M
                        STR   284   216     8 016100              INS <= LOW-VALUES
                        STR   284   216     8 022900              INS <= LOW-VALUES
                        STR   284   192     8 021500              DCW MAP:TARMU3M
                        STR   284   192     8 016100              INS <= LOW-VALUES
                        STR   284   192     8 022900              INS <= LOW-VALUES
                        STR   284   204     8 021500              DCW MAP:TARMU3M
                        STR   284   204     8 016100              INS <= LOW-VALUES
                        STR   284   204     8 022900              INS <= LOW-VALUES
  2  TARMU3MJDATEO 180 (YYXXXXXX)
                        CHA     8             000100   TARMU3M    DCL
     +TARMU3
                        CHA     8     1     8 026100              INS <= TARDATE-DATE OF TARDATE-PARAMETERS
                        CHA     8     1     8 023500              INS <= SPACES
                        CHA     8     1     8 036400              INS <= WORK-TODAYS-MMDDYY OF WORK-VARS
  2  TARMU3MBDATEO 192 (YYXXXXXX)
                        CHA     8             000103   TARMU3M    DCL
     +TARMU3
                        CHA     8     1     8 027000              INS <= TARDATE-DATE OF TARDATE-PARAMETERS
                        CHA     8     1     8 023500              INS <= SPACES
  2  TARMU3MTDATEO 204 (YYXXXXXX)
                        CHA     8             000106   TARMU3M    DCL
     +TARMU3
                        CHA     8     1     8 028000              INS <= TARDATE-DATE OF TARDATE-PARAMETERS
                        CHA     8     1     8 041000              INS => TARMU3MSDATEO OF TARMU3MO
                        CHA     8     1     8 023500              INS <= SPACES
                        CHA     8     1     8 028200              INS <= SPACES
```

Figure 52 (Part 11 of 14). Detailed Report (Year 2000 Analysis)

```
  2  TARMU3MSDATEO 216 (YYXXXXYY)
                       CHA     8             000109   TARMU3M    DCL
     +TARMU3
                       CHA     8     1     8 029200              INS <= TARDATE-DATE OF TARDATE-PARAMETERS
                       CHA     8     1     8 044500              INS <= WORK-JOINED-MMDDYY OF WORK-VARS
                       CHA     8     1     8 023500              INS <= SPACES
                       CHA     8     1     8 041000              INS <= TARMU3MTDATEO OF TARMU3MO
1  EIBDATE   (XXYYXXX)
   +TARMU3
                       DEC     4     1     4 015000              INS => WORK-EIB-DATE OF WORK-VARS
++CN "02/29/"
     +TARMU3
                       CHA     8     1     6 044200              REF => WORK-JOINED-MMDD OF WORK-JOINED-MMDDYY OF W
++CN 1
     +TARMU3
                       NUM     1     1     2 044100              CLC WORK-JOINED-YY OF WORK-JOINED-MMDDYY OF WORK-V
          ---------1---------2---------3---------4---------5---------6---------7---------8
         >003400 01  EMPLOYEE-MASTER-RECORD.                                        00000034<
         >004400     03  EMP-DATE-JOINED            PIC 9(5).                    00000044<
         >004600     03  EMP-DATE-TERMINATED        PIC 9(6).                    00000046<
         >004800     03  EMP-DATE-MAINTAINED        PIC 9(5).                    00000048<
         >005000     03  EMP-BIRTH-DATE             PIC 9(5).                    00000050<
         >005200     03  EMP-SECURITY-EXP           PIC 9(5) COMP-3.             00000052<
         >009800 01  WORK-VARS.                                                  00000098<
         >009900     03  WORK-TODAYS-MMDDYY         PIC 9(8).                    00000099<
         >010100     03  WORK-EIB-DATE              PIC 9(7) COMP-3.             00000101<
         >010200     03  WORK-EIB-DATE-CHAR.                                     00000102<
         >010500         05  WORK-EIB-YYDDD         PIC X(5).                    00000105<
         >010600     03  WORK-JOINED-YYDDD          PIC 9(5).                    00000106<
         >010700     03  WORK-JOINED-MMDDYY.                                     00000107<
         >010800         05  WORK-JOINED-MMDD       PIC X(6).                    00000108<
         >010900         05  WORK-JOINED-YY         PIC 99.                      00000109<
         >011100     03  WORK-TERMINATED-YYDDD      PIC 9(5).                    00000111<
         >011800 01  TARDATE-PARAMETERS.                                         00000118<
         >011900     03  TARDATE-DATE               PIC X(8).                    00000119<
         >012000     03  TARDATE-DATE-RED1 REDEFINES TARDATE-DATE.               00000120<
         >012100         05  TARDATE-DATE-YYDDD     PIC 9(5).                    00000121<
         >012300     03  TARDATE-DATE-RED2 REDEFINES TARDATE-DATE.               00000123<
         >012400         05  TARDATE-DATE-YYMMDD    PIC 9(6).                    00000124<
 TARMU3M >000001 01  TARMU3MI.                                                        <
 TARMU3M >000050     02  TARMU3MJDATEI  PIC X(8).                                    <
 TARMU3M >000056     02  TARMU3MBDATEI  PIC X(8).                                    <
 TARMU3M >000062     02  TARMU3MTDATEI  PIC X(8).                                    <
 TARMU3M >000068     02  TARMU3MSDATEI  PIC X(8).                                    <
 TARMU3M >000075 01  TARMU3MO REDEFINES TARMU3MI.                                  <
 TARMU3M >000100     02  TARMU3MJDATEO  PIC X(8).                                    <
 TARMU3M >000103     02  TARMU3MBDATEO  PIC X(8).                                    <
 TARMU3M >000106     02  TARMU3MTDATEO  PIC X(8).                                    <
 TARMU3M >000109     02  TARMU3MSDATEO  PIC X(8).                                    <
         >015000     MOVE EIBDATE        TO  WORK-EIB-DATE.                       00000150<
         >015100     MOVE WORK-EIB-DATE  TO  WORK-EIB-DATE-CHAR.                  00000150<
         >015200     MOVE WORK-EIB-YYDDD TO  TARDATE-DATE.                        00000151<
         >015500     CALL "TARDTE3" USING TARDATE-DATE                            00000154<
         >015600                          TARDATE-INPUT-FORMAT                    00000155<
         >015700                          TARDATE-OUTPUT-FORMAT                   00000156<
         >015800                          TARDATE-MESSAGE.                        00000157<
         >015900     MOVE TARDATE-DATE   TO  WORK-TODAYS-MMDDYY.                  00000158<
         >016100     MOVE LOW-VALUES     TO  TARMU3MO.                            00000160<
```

Figure 52 (Part 12 of 14). Detailed Report (Year 2000 Analysis)

```
>021500     EXEC CICS SEND MAP("TARMU3M") FROM(TARMU3MO)                 00000214<
>021600                  CURSOR FREEKB END-EXEC.                         00000215<
>022300     EXEC CICS READ FILE("EMPMAST") INTO(EMPLOYEE-MASTER-RECORD)  00000222<
>022400          LENGTH(EMP-LENGTH)                                      00000223<
>022500          RIDFLD(EMP-ID)                                          00000224<
>022600          RESP(RESPONSE)                                          00000225<
>022700          END-EXEC.                                               00000226<
>022900     MOVE LOW-VALUES     TO  TARMU3MO.                           00000228<
>023500         MOVE SPACES               TO TARMU3MDEPO                00000234<
>023600                                      TARMU3MNAMEO               00000235<
>023700                                      TARMU3MADDR1O              00000236<
>023800                                      TARMU3MADDR2O              00000237<
>023900                                      TARMU3MADDR3O              00000238<
>024000                                      TARMU3MZIPO                00000239<
>024100                                      TARMU3MJDATEO              00000240<
>024200                                      TARMU3MBDATEO              00000241<
>024300                                      TARMU3MTDATEO              00000242<
>024400                                      TARMU3MSDATEO              00000243<
>025400         MOVE EMP-DATE-JOINED      TO TARDATE-DATE               00000253<
>025700         CALL "TARDTE3" USING TARDATE-DATE                       00000256<
>025800                              TARDATE-INPUT-FORMAT               00000257<
>025900                              TARDATE-OUTPUT-FORMAT              00000258<
>026000                              TARDATE-MESSAGE                    00000259<
>026100         MOVE TARDATE-DATE         TO TARMU3MJDATEO              00000260<
>026300         MOVE EMP-BIRTH-DATE       TO TARDATE-DATE               00000262<
>026600         CALL "TARDTE3" USING TARDATE-DATE                       00000265<
>026700                              TARDATE-INPUT-FORMAT               00000266<
>026800                              TARDATE-OUTPUT-FORMAT              00000267<
>026900                              TARDATE-MESSAGE                    00000268<
>027000         MOVE TARDATE-DATE         TO TARMU3MBDATEO              00000269<
>027200         IF  EMP-DATE-TERMINATED > ZEROS THEN                    00000271<
>027300             MOVE EMP-DATE-TERMINATED  TO TARDATE-DATE           00000272<
>027600             CALL "TARDTE3" USING TARDATE-DATE                   00000275<
>027700                                  TARDATE-INPUT-FORMAT           00000276<
>027800                                  TARDATE-OUTPUT-FORMAT          00000277<
>027900                                  TARDATE-MESSAGE                00000278<
>028000             MOVE TARDATE-DATE         TO TARMU3MTDATEO          00000279<
>028200             MOVE SPACES               TO TARMU3MTDATEO          00000281<
>028500         MOVE EMP-SECURITY-EXP     TO TARDATE-DATE               00000284<
>028800         CALL "TARDTE3" USING TARDATE-DATE                       00000287<
>028900                              TARDATE-INPUT-FORMAT               00000288<
>029000                              TARDATE-OUTPUT-FORMAT              00000289<
>029100                              TARDATE-MESSAGE                    00000290<
>029200         MOVE TARDATE-DATE         TO TARMU3MSDATEO              00000291<
>034400     IF  TARMU3MJDATEI > SPACES     THEN                         00000343<
>034500         MOVE TARMU3MJDATEI   TO TARDATE-DATE                    00000344<
>034800         CALL "TARDTE3" USING TARDATE-DATE                       00000347<
>034900                              TARDATE-INPUT-FORMAT               00000348<
>035000                              TARDATE-OUTPUT-FORMAT              00000349<
>035100                              TARDATE-MESSAGE                    00000350<
>036000             MOVE TARDATE-DATE-YYDDD TO WORK-JOINED-YYDDD        00000359<
>036100             MOVE TARMU3MJDATEI      TO WORK-JOINED-MMDDYY       00000360<
>036400         MOVE WORK-TODAYS-MMDDYY TO TARMU3MJDATEO                00000363<
>036500                                    WORK-JOINED-MMDDYY           00000364<
>036600         MOVE WORK-EIB-YYDDD     TO WORK-JOINED-YYDDD            00000365<
>036900     IF  TARMU3MBDATEI > SPACES     THEN                         00000368<
>037000         MOVE TARMU3MBDATEI   TO TARDATE-DATE                    00000369<
>037300         CALL "TARDTE3" USING TARDATE-DATE                       00000372<
>037400                              TARDATE-INPUT-FORMAT               00000373<
>037500                              TARDATE-OUTPUT-FORMAT              00000374<
>037600                              TARDATE-MESSAGE                    00000375<
```

Figure 52 (Part 13 of 14). Detailed Report (Year 2000 Analysis)

```
>039400     IF  TARMU3MTDATEI > SPACES THEN                                00000393<
>039500         MOVE TARMU3MTDATEI   TO TARDATE-DATE                       00000394<
>039800         CALL "TARDTE3" USING TARDATE-DATE                          00000397<
>039900                              TARDATE-INPUT-FORMAT                  00000398<
>040000                              TARDATE-OUTPUT-FORMAT                 00000399<
>040100                              TARDATE-MESSAGE                       00000400<
>041000             MOVE TARMU3MTDATEO      TO TARMU3MSDATEO               00000409<
>041100             MOVE TARDATE-DATE-YYDDD TO WORK-TERMINATED-YYDDD       00000410<
>041200             IF  WORK-TERMINATED-YYDDD < WORK-JOINED-YYDDD          00000411<
>042400     IF  TARMU3MSDATEI > SPACES     THEN                            00000423<
>042500         MOVE TARMU3MSDATEI   TO TARDATE-DATE                       00000424<
>042800         CALL "TARDTE3" USING TARDATE-DATE                          00000427<
>042900                              TARDATE-INPUT-FORMAT                  00000428<
>043000                              TARDATE-OUTPUT-FORMAT                 00000429<
>043100                              TARDATE-MESSAGE                       00000430<
>044100         COMPUTE WORK-JOINED-YY = WORK-JOINED-YY + 1                00000440<
>044200         IF  WORK-JOINED-MMDD = "02/29/" THEN                       00000441<
>044300             MOVE "02/28/"  TO WORK-JOINED-MMDD                     00000442<
>044500         MOVE WORK-JOINED-MMDDYY TO TARMU3MSDATEO                   00000444<
>045100     EXEC CICS READ FILE("EMPMAST") INTO(EMPLOYEE-MASTER-RECORD)    00000450<
>045200          LENGTH(EMP-LENGTH)                                        00000451<
>045300          RIDFLD(EMP-ID)                                            00000452<
>045400          RESP(RESPONSE)                                            00000453<
>045500          UPDATE                                                    00000454<
>045600          END-EXEC.                                                 00000455<
>045800     MOVE SPACES             TO  EMPLOYEE-MASTER-RECORD.            00000457<
>046600     MOVE TARMU3MJDATEI      TO  TARDATE-DATE                       00000465<
>046900     CALL "TARDTE3" USING        TARDATE-DATE                       00000468<
>047000                                 TARDATE-INPUT-FORMAT               00000469<
>047100                                 TARDATE-OUTPUT-FORMAT              00000470<
>047200                                 TARDATE-MESSAGE.                   00000471<
>047300     MOVE TARDATE-DATE-YYDDD TO  EMP-DATE-JOINED.                   00000472<
>047400     MOVE TARMU3MBDATEI      TO  TARDATE-DATE                       00000473<
>047700     CALL "TARDTE3" USING        TARDATE-DATE                       00000476<
>047800                                 TARDATE-INPUT-FORMAT               00000477<
>047900                                 TARDATE-OUTPUT-FORMAT              00000478<
>048000                                 TARDATE-MESSAGE.                   00000479<
>048100     MOVE TARDATE-DATE-YYDDD TO  EMP-BIRTH-DATE.                    00000480<
>048300     IF  TARMU3MTDATEI > SPACES THEN                                00000482<
>048400         MOVE TARMU3MTDATEI   TO TARDATE-DATE                       00000483<
>048700         CALL "TARDTE3" USING    TARDATE-DATE                       00000486<
>048800                                 TARDATE-INPUT-FORMAT               00000487<
>048900                                 TARDATE-OUTPUT-FORMAT              00000488<
>049000                                 TARDATE-MESSAGE                    00000489<
>049100         MOVE TARDATE-DATE-YYMMDD TO  EMP-DATE-TERMINATED           00000490<
>049300         MOVE ZEROS           TO  EMP-DATE-TERMINATED               00000492<
>049500     MOVE TARMU3MSDATEI      TO  TARDATE-DATE                       00000494<
>049800     CALL "TARDTE3" USING        TARDATE-DATE                       00000497<
>049900                                 TARDATE-INPUT-FORMAT               00000498<
>050000                                 TARDATE-OUTPUT-FORMAT              00000499<
>050100                                 TARDATE-MESSAGE.                   00000500<
>050200     MOVE TARDATE-DATE-YYDDD TO  EMP-SECURITY-EXP.                  00000501<
>050400     MOVE WORK-EIB-YYDDD     TO  EMP-DATE-MAINTAINED.               00000503<
>050700         EXEC CICS WRITE FILE("EMPMAST")                            00000506<
>050800                   FROM(EMPLOYEE-MASTER-RECORD)                     00000507<
>050900                   LENGTH(EMP-LENGTH)                               00000508<
>051000                   RIDFLD(EMP-ID)                                   00000509<
>051100                   RESP(RESPONSE)                                   00000510<
>051200                   END-EXEC                                         00000511<
>051600         EXEC CICS REWRITE FILE("EMPMAST")                          00000515<
>051700                   FROM(EMPLOYEE-MASTER-RECORD)                     00000516<
>051800                   LENGTH(EMP-LENGTH)                               00000517<
>051900                   RESP(RESPONSE)                                   00000518<
>052000                   END-EXEC                                         00000519<
```

Figure 52 (Part 14 of 14). Detailed Report (Year 2000 Analysis)

The detailed report provides information additional to the information in the summary report. The source statements especially can immediately help to find out if an item needs to be looked at more closely in order to make the program Year 2000 ready. But the most important information in the detailed report that

you have to check are the messages. They tell you if there is anything wrong with your starting-point file. If you find such messages in the detailed report, you need to correct your starting-point file and run the analysis again.

4.5.3 PC Export File

The PC export file contains the same information as the detailed report, except for the retrieval parameters, the messages, and the reason codes. All occurrences are listed. The PC export file has a table layout, which means that every record has the same structure. The columns are divided by tab characters (X'05'). That allows you to read the file with any workstation-based spreadsheet program.

Unlike the detailed report, in the PC export file, each corresponding source statement is listed in the same row as the item itself. The record layout of the PC export file is as follows:

- JCL member or program name.
- Language (CBL, PLI, JCL)
- Data set name or program item name. If an item belongs to a structure it is not qualified.
- Attribute of item
- Length of item
- Position and length of date within item
- Program line of item occurrence
- Copybook, include, or program source where item is defined
- Source statements where item is referenced

If you have your report data in a spreadsheet, you can easily further analyze the information. You can hide specific columns or rows and delete duplicate entries. You can better check if a variable contains a date because you see the source statement where it is referenced in the same place. Similarily, you can also verify whether or not your starting-point file contains correct data only.

In order to create a PC export file and import it into a spreadsheet program, do the following:

1. Specify action X on the **Retrieval Result Management** panel (Figure 48 on page 123) next to the row indicating the results you want to export, and press Enter.
2. On the next panel (Figure 53 on page 144) specify a valid job card and the data set name for the PC export file. The data set must not exist. MA2000 always allocates it as a sequential data set.

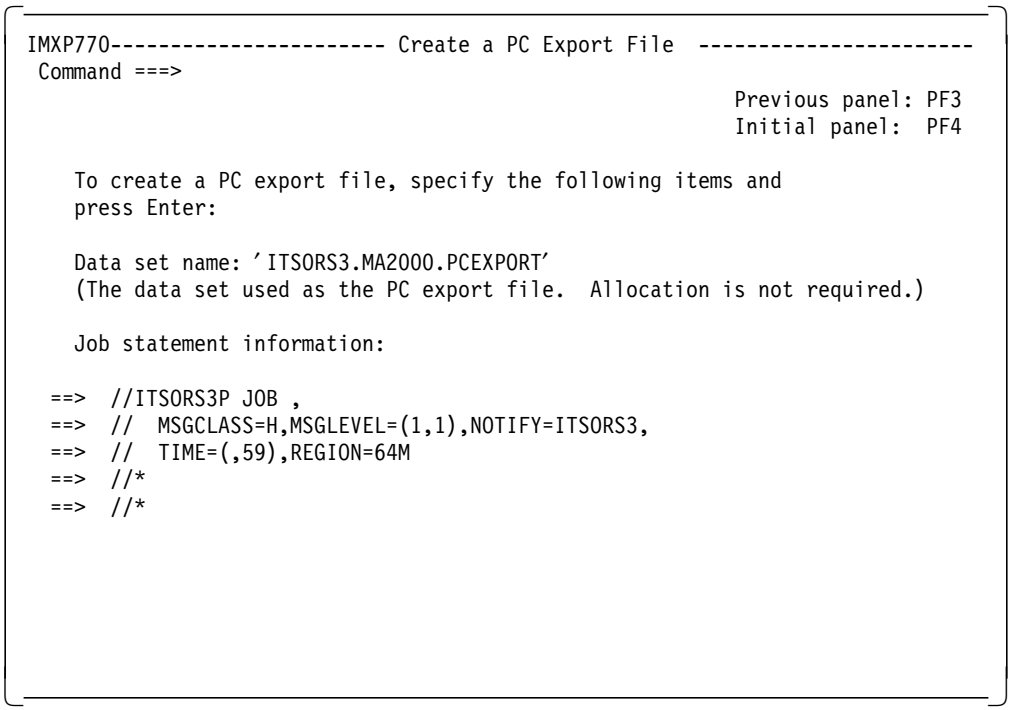


Figure 53. Create a PC Export File Panel

3. Download the file to your workstation in text mode with ASCII translation.
4. Start your spreadsheet program, for example *Lotus 1-2-3*.
5. Open your downloaded PC export file. Because the file is in text format and not in spreadsheet format, you have to specify a parsing option when you open the file. Specify *Start a new column at each* **Tab** because MA2000 inserts a tab character to separate the columns in each row.
6. Save the file as a spreadsheet file.

Figure 54 on page 145 shows a part of our sample report with the PC export file format.

```
SYSITSO  SUBSITSO  Y2K ANALYSIS WITH SP YEAR2000
Member   Language  Item/DS Name               Attribute  Length  Position  Length  Line no.1  Line no.2  Reference member  Statement
TARDTE3  CBL       WORK-DATES                 STR        56                          000001     00001      TARDTE3C          000001 01 WORK-DATES.
TARDTE3  CBL       WRK-DATE                   STR        6                           000003     00003      TARDTE3C          000003 03 WRK-DATE.
TARDTE3  CBL       WRK-DATE                   STR        6         1       2         000003     00003      TARDTE3C          000003 03 WRK-DATE.
TARDTE3  CBL       WRK-DATE-YY                CHA        2                           000004     00004      TARDTE3C          000004 05 WRK-DATE-YY   PIC XX.
TARDTE3  CBL       WRK-DATE-YY                CHA        2         1       2         000004     00004      TARDTE3C          000004 05 WRK-DATE-YY   PIC XX.
TARDTE3  CBL       WRK-DATE-RED               STR        6                           000008     00008      TARDTE3C          000008 03 WRK-DATE-RED REDEFINES WRK-DATE.
TARDTE3  CBL       WRK-DATE-RED               STR        6         1       2         000008     00008      TARDTE3C          000008 03 WRK-DATE-RED REDEFINES WRK-DATE.
TARDTE3  CBL       WRK-DTE-YY                 CHA        2                           000009     00009      TARDTE3C          000009 05 WRK-DTE-YY    PIC XX.
TARDTE3  CBL       WRK-DTE-YY                 CHA        2         1       2         000009     00009      TARDTE3C          000009 05 WRK-DTE-YY    PIC XX.
TARDTE3  CBL       WRK-DATE-YYDDD             STR        5                           000015     00015      TARDTE3C          000015 03 WRK-DATE-YYDDD.
TARDTE3  CBL       WRK-DATE-YYDDD             STR        5         1       2         000015     00015      TARDTE3C          000015 03 WRK-DATE-YYDDD.
TARDTE3  CBL       WRK-DATE-YY                CHA        2                           000016     00016      TARDTE3C          000016 05 WRK-DATE-YY   PIC XX.
   ⋮
TARDTE3  CBL       INPUT-DATE                 CHA        8         1       8         000062     00062      TARDTE3           000062 PROCEDURE DIVISION USING INPUT-DATE
TARDTE3  CBL       INPUT-DATE                 CHA        8         7       2         000062     00062      TARDTE3           000062 PROCEDURE DIVISION USING INPUT-DATE
TARDTE3  CBL       INPUT-DATE                 CHA        8         1       2         000063     00063      TARDTE3           000063         INPUT-FORMAT
TARDTE3  CBL       INPUT-DATE                 CHA        8         1       8         000063     00063      TARDTE3           000063         INPUT-FORMAT
TARDTE3  CBL       INPUT-DATE                 CHA        8         7       2         000063     00063      TARDTE3           000063         INPUT-FORMAT
TARDTE3  CBL       INPUT-DATE                 CHA        8         1       2         000064     00064      TARDTE3           000064         OUTPUT-FORMAT
TARDTE3  CBL       INPUT-DATE                 CHA        8         1       8         000064     00064      TARDTE3           000064         OUTPUT-FORMAT
TARDTE3  CBL       INPUT-DATE                 CHA        8         7       2         000064     00064      TARDTE3           000064         OUTPUT-FORMAT
TARDTE3  CBL       INPUT-DATE                 CHA        8         1       2         000065     00065      TARDTE3           000065         MSG.
TARDTE3  CBL       INPUT-DATE                 CHA        8         1       8         000065     00065      TARDTE3           000065         MSG.
TARDTE3  CBL       INPUT-DATE                 CHA        8         7       2         000065     00065      TARDTE3           000065         MSG.
TARDTE3  CBL       WRK-DATE-MMXDDXYY          STR        8         1       8         000071     00071      TARDTE3           000071 MOVE INPUT-DATE TO WRK-DATE-MMXDDXYY
TARDTE3  CBL       WRK-DATE-MMXDDXYY          STR        8         7       2         000071     00071      TARDTE3           000071 MOVE INPUT-DATE TO WRK-DATE-MMXDDXYY
TARDTE3  CBL       INPUT-DATE                 CHA        8         1       2         000071     00071      TARDTE3           000071 MOVE INPUT-DATE TO WRK-DATE-MMXDDXYY
TARDTE3  CBL       INPUT-DATE                 CHA        8         1       8         000071     00071      TARDTE3           000071 MOVE INPUT-DATE TO WRK-DATE-MMXDDXYY
TARDTE3  CBL       INPUT-DATE                 CHA        8         7       2         000071     00071      TARDTE3           000071 MOVE INPUT-DATE TO WRK-DATE-MMXDDXYY
TARDTE3  CBL       WRK-DATE-YY                CHA        2         1       2         000072     00072      TARDTE3           000072 MOVE CORRESPONDING WRK-DATE-MMXDDXYY TO WRK-DATE
TARDTE3  CBL       WRK-DATE-YY                CHA        2         1       2         000072     00072      TARDTE3           000072 MOVE CORRESPONDING WRK-DATE-MMXDDXYY TO WRK-DATE
TARDTE3  CBL       WRK-DATE-YY                CHA        2         1       2         000073     00073      TARDTE3           000073 MOVE CORRESPONDING WRK-DATE-MMXDDXYY TO WRK-DATE-YYDDD
TARDTE3  CBL       WRK-DATE-YY                CHA        2         1       2         000073     00073      TARDTE3           000073 MOVE CORRESPONDING WRK-DATE-MMXDDXYY TO WRK-DATE-YYDDD
TARDTE3  CBL       WRK-DATE-YYXDDXMM          STR        8         1       2         000076     00076      TARDTE3           000076 MOVE INPUT-DATE TO WRK-DATE-YYXDDXMM
   ⋮
TARDTE3  CBL       WRK-DATE                   STR        6         1       6         000134     00134      TARDTE3           000134 MOVE WRK-DATE TO INPUT-DATE
TARDTE3  CBL       INPUT-DATE                 CHA        8         1       2         000134     00134      TARDTE3           000134 MOVE WRK-DATE TO INPUT-DATE
TARDTE3  CBL       INPUT-DATE                 CHA        8         1       8         000134     00134      TARDTE3           000134 MOVE WRK-DATE TO INPUT-DATE
TARDTE3  CBL       INPUT-DATE                 CHA        8         7       2         000134     00134      TARDTE3           000134 MOVE WRK-DATE TO INPUT-DATE
TARDTE3  CBL       WRK-DATE-YYDDD             STR        5         1       2         000144     00144      TARDTE3           000144 MOVE WRK-DATE-YYDDD TO INPUT-DATE
TARDTE3  CBL       WRK-DATE-YYDDD             STR        5         1       5         000144     00144      TARDTE3           000144 MOVE WRK-DATE-YYDDD TO INPUT-DATE
TARDTE3  CBL       INPUT-DATE                 CHA        8         1       2         000144     00144      TARDTE3           000144 MOVE WRK-DATE-YYDDD TO INPUT-DATE
TARDTE3  CBL       INPUT-DATE                 CHA        8         1       8         000144     00144      TARDTE3           000144 MOVE WRK-DATE-YYDDD TO INPUT-DATE
TARDTE3  CBL       INPUT-DATE                 CHA        8         7       2         000144     00144      TARDTE3           000144 MOVE WRK-DATE-YYDDD TO INPUT-DATE
TARDTE3  CBL       WRK-DATE-YY                CHA        2         1       2         000147     00147      TARDTE3           000147 MOVE CORRESPONDING WRK-DATE TO WRK-DATE-MMXDDXYY
TARDTE3  CBL       WRK-DATE-YY                CHA        2         1       2         000147     00147      TARDTE3           000147 MOVE CORRESPONDING WRK-DATE TO WRK-DATE-MMXDDXYY
TARDTE3  CBL       WRK-DATE-MMXDDXYY          STR        8         1       8         000148     00148      TARDTE3           000148 MOVE "/"   TO WRK-DATE-MMXDDXYY(3:1)
TARDTE3  CBL       "/"                        CHA        3         1       1         000148     00148      TARDTE3           000148 MOVE "/"   TO WRK-DATE-MMXDDXYY(3:1)
TARDTE3  CBL       WRK-DATE-MMXDDXYY          STR        8         1       8         000149     00149      TARDTE3           000149              WRK-DATE-MMXDDXYY(6:1)
TARDTE3  CBL       "/"                        CHA        3         1       1         000149     00149      TARDTE3           000149              WRK-DATE-MMXDDXYY(6:1)
TARDTE3  CBL       WRK-DATE-MMXDDXYY          STR        8         1       8         000150     00150      TARDTE3           000150 MOVE WRK-DATE-MMXDDXYY TO INPUT-DATE
TARDTE3  CBL       WRK-DATE-MMXDDXYY          STR        8         7       2         000150     00150      TARDTE3           000150 MOVE WRK-DATE-MMXDDXYY TO INPUT-DATE
TARDTE3  CBL       INPUT-DATE                 CHA        8         1       2         000150     00150      TARDTE3           000150 MOVE WRK-DATE-MMXDDXYY TO INPUT-DATE
```

Figure 54 (Part 1 of 3). PC Export File (Year 2000 Analysis)

```
TARDTE3  CBL      INPUT-DATE                CHA      8        1      8      000150    00150     TARDTE3          000150 MOVE WRK-DATE-MMXDDXYY TO INPUT-DATE
TARDTE3  CBL      INPUT-DATE                CHA      8        7      2      000150    00150     TARDTE3          000150 MOVE WRK-DATE-MMXDDXYY TO INPUT-DATE
TARDTE3  CBL      WRK-DTE-YY                CHA      2        1      2      000167    00167     TARDTE3          000167 MOVE WRK-DTE-YY TO WRK-YEAR-YY
TARDTE3  CBL      WRK-YEAR-YY               CHA      2        1      2      000167    00167     TARDTE3          000167 MOVE WRK-DTE-YY TO WRK-YEAR-YY
TARDTE3  CBL      WRK-YEAR-YYYY-NUM         CHA      4        1      2      000168    00168     TARDTE3          000168 DIVIDE WRK-YEAR-YYYY-NUM BY 4 GIVING WHOLE-NUMBER
TARDTE3  CBL      WRK-YEAR-YYYY-NUM         CHA      4        3      2      000168    00168     TARDTE3          000168 DIVIDE WRK-YEAR-YYYY-NUM BY 4 GIVING WHOLE-NUMBER
TARDTE3  CBL      WRK-YEAR-YYYY-NUM         CHA      4        1      2      000169    00169     TARDTE3          000169                 REMAINDER WRK-LEAP
TARDTE3  CBL      WRK-YEAR-YYYY-NUM         CHA      4        3      2      000169    00169     TARDTE3          000169                 REMAINDER WRK-LEAP
TARDTE3  CBL      WRK-DATE-YY               CHA      2        1      2      000187    00187     TARDTE3          000187 MOVE CORRESPONDING WRK-DATE-YYDDD TO WRK-DATE
TARDTE3  CBL      WRK-DATE-YY               CHA      2        1      2      000187    00187     TARDTE3          000187 MOVE CORRESPONDING WRK-DATE-YYDDD TO WRK-DATE
TARMU3   CBL      TARMU3MI                  STR      284                    000001    00001     TARMU3M          000001 01 TARMU3MI.
TARMU3   CBL      EMPLOYEE-MASTER-RECORD    STR      200                    003400    00034     TARMU3           003400 01 EMPLOYEE-MASTER-RECORD.
TARMU3   CBL      EMP-DATE-JOINED           CHA      5                      004400    00044     TARMU3           004400 03 EMP-DATE-JOINED PIC 9(5).
TARMU3   CBL      EMP-DATE-JOINED           CHA      5        1      5      004400    00044     TARMU3           004400 03 EMP-DATE-JOINED PIC 9(5).
TARMU3   CBL      EMP-DATE-TERMINATED       CHA      6                      004600    00046     TARMU3           004600 03 EMP-DATE-TERMINATED PIC 9(6).
TARMU3   CBL      EMP-DATE-TERMINATED       CHA      6        1      6      004600    00046     TARMU3           004600 03 EMP-DATE-TERMINATED PIC 9(6).
TARMU3   CBL      EMP-DATE-MAINTAINED       CHA      5                      004800    00048     TARMU3           004800 03 EMP-DATE-MAINTAINED PIC 9(5).
TARMU3   CBL      EMP-DATE-MAINTAINED       CHA      5        1      5      004800    00048     TARMU3           004800 03 EMP-DATE-MAINTAINED PIC 9(5).
TARMU3   CBL      EMP-BIRTH-DATE            CHA      5                      005000    00050     TARMU3           005000 03 EMP-BIRTH-DATE  PIC 9(5).
TARMU3   CBL      EMP-BIRTH-DATE            CHA      5        1      5      005000    00050     TARMU3           005000 03 EMP-BIRTH-DATE  PIC 9(5).
TARMU3   CBL      TARMU3MJDATEI             CHA      8                      000050    00050     TARMU3M          000050 02 TARMU3MJDATEI PIC X(8).
TARMU3   CBL      EMP-SECURITY-EXP          DEC      3                      005200    00052     TARMU3           005200 03 EMP-SECURITY-EXP PIC 9(5) COMP-3.
TARMU3   CBL      TARMU3MBDATEI             CHA      8                      000056    00056     TARMU3M          000056 02 TARMU3MBDATEI PIC X(8).
TARMU3   CBL      TARMU3MTDATEI             CHA      8                      000062    00062     TARMU3M          000062 02 TARMU3MTDATEI PIC X(8).
TARMU3   CBL      TARMU3MSDATEI             CHA      8                      000068    00068     TARMU3M          000068 02 TARMU3MSDATEI PIC X(8).
TARMU3   CBL      TARMU3MO                  STR      284                    000075    00075     TARMU3M          000075 01 TARMU3MO REDEFINES TARMU3MI.
TARMU3   CBL      WORK-VARS                 STR      44                     009800    00098     TARMU3           009800 01 WORK-VARS.
TARMU3   CBL      WORK-TODAYS-MMDDYY        CHA      8                      009900    00099     TARMU3           009900 03 WORK-TODAYS-MMDDYY PIC 9(8).
   ⋮
TARMU3   CBL      TARDATE-DATE              CHA      8        1      2      015500    00156     TARMU3           015500 CALL "TARDTE3" USING TARDATE-DATE
TARMU3   CBL      TARDATE-DATE              CHA      8        1      5      015500    00156     TARMU3           015500 CALL "TARDTE3" USING TARDATE-DATE
TARMU3   CBL      TARDATE-DATE              CHA      8        1      6      015500    00156     TARMU3           015500 CALL "TARDTE3" USING TARDATE-DATE
TARMU3   CBL      TARDATE-DATE              CHA      8        1      8      015500    00156     TARMU3           015500 CALL "TARDTE3" USING TARDATE-DATE
TARMU3   CBL      TARDATE-DATE              CHA      8        7      2      015500    00156     TARMU3           015500 CALL "TARDTE3" USING TARDATE-DATE
TARMU3   CBL      TARDATE-DATE              CHA      8        1      2      015600    00157     TARMU3           015600         TARDATE-INPUT-FORMAT
TARMU3   CBL      TARDATE-DATE              CHA      8        1      5      015600    00157     TARMU3           015600         TARDATE-INPUT-FORMAT
TARMU3   CBL      TARDATE-DATE              CHA      8        1      6      015600    00157     TARMU3           015600         TARDATE-INPUT-FORMAT
TARMU3   CBL      TARDATE-DATE              CHA      8        1      8      015600    00157     TARMU3           015600         TARDATE-INPUT-FORMAT
TARMU3   CBL      TARDATE-DATE              CHA      8        7      2      015600    00157     TARMU3           015600         TARDATE-INPUT-FORMAT
TARMU3   CBL      TARDATE-DATE              CHA      8        1      2      015700    00158     TARMU3           015700         TARDATE-OUTPUT-FORMAT
TARMU3   CBL      TARDATE-DATE              CHA      8        1      5      015700    00158     TARMU3           015700         TARDATE-OUTPUT-FORMAT
TARMU3   CBL      TARDATE-DATE              CHA      8        1      6      015700    00158     TARMU3           015700         TARDATE-OUTPUT-FORMAT
TARMU3   CBL      TARDATE-DATE              CHA      8        1      8      015700    00158     TARMU3           015700         TARDATE-OUTPUT-FORMAT
TARMU3   CBL      TARDATE-DATE              CHA      8        7      2      015700    00158     TARMU3           015700         TARDATE-OUTPUT-FORMAT
TARMU3   CBL      TARDATE-DATE              CHA      8        1      2      015800    00159     TARMU3           015800         TARDATE-MESSAGE.
TARMU3   CBL      TARDATE-DATE              CHA      8        1      5      015800    00159     TARMU3           015800         TARDATE-MESSAGE.
TARMU3   CBL      TARDATE-DATE              CHA      8        1      6      015800    00159     TARMU3           015800         TARDATE-MESSAGE.
TARMU3   CBL      TARDATE-DATE              CHA      8        1      8      015800    00159     TARMU3           015800         TARDATE-MESSAGE.
TARMU3   CBL      TARDATE-DATE              CHA      8        7      2      015800    00159     TARMU3           015800         TARDATE-MESSAGE.
TARMU3   CBL      WORK-TODAYS-MMDDYY        CHA      8        1      8      015900    00160     TARMU3           015900 MOVE TARDATE-DATE TO WORK-TODAYS-MMDDYY.
TARMU3   CBL      TARDATE-DATE              CHA      8        1      5      015900    00160     TARMU3           015900 MOVE TARDATE-DATE TO WORK-TODAYS-MMDDYY.
TARMU3   CBL      TARDATE-DATE              CHA      8        1      6      015900    00160     TARMU3           015900 MOVE TARDATE-DATE TO WORK-TODAYS-MMDDYY.
TARMU3   CBL      TARDATE-DATE              CHA      8        1      8      015900    00160     TARMU3           015900 MOVE TARDATE-DATE TO WORK-TODAYS-MMDDYY.
TARMU3   CBL      TARDATE-DATE              CHA      8        7      2      015900    00160     TARMU3           015900 MOVE TARDATE-DATE TO WORK-TODAYS-MMDDYY.
TARMU3   CBL      TARMU3MO                  STR      284      180    8      016100    00162     TARMU3           016100 MOVE LOW-VALUES TO TARMU3MO.
```

Figure 54 (Part 2 of 3). PC Export File (Year 2000 Analysis)

```
TARMU3   CBL   TARMU3MO                  STR   284   192   8   016100   00162   TARMU3    016100 MOVE LOW-VALUES TO TARMU3MO.
TARMU3   CBL   TARMU3MO                  STR   284   204   8   016100   00162   TARMU3    016100 MOVE LOW-VALUES TO TARMU3MO.
TARMU3   CBL   TARMU3MO                  STR   284   216   8   016100   00162   TARMU3    016100 MOVE LOW-VALUES TO TARMU3MO.
TARMU3   CBL   TARMU3MO                  STR   284   180   8   021500   00216   TARMU3    021500 EXEC CICS SEND MAP("TARMU3M") FROM(TARMU3MO)
TARMU3   CBL   TARMU3MO                  STR   284   192   8   021500   00216   TARMU3    021500 EXEC CICS SEND MAP("TARMU3M") FROM(TARMU3MO)
TARMU3   CBL   TARMU3MO                  STR   284   204   8   021500   00216   TARMU3    021500 EXEC CICS SEND MAP("TARMU3M") FROM(TARMU3MO)
TARMU3   CBL   TARMU3MO                  STR   284   216   8   021500   00216   TARMU3    021500 EXEC CICS SEND MAP("TARMU3M") FROM(TARMU3MO)
TARMU3   CBL   TARMU3MO                  STR   284   180   8   021600   00217   TARMU3    021600  CURSOR FREEKB END-EXEC.
TARMU3   CBL   TARMU3MO                  STR   284   192   8   021600   00217   TARMU3    021600  CURSOR FREEKB END-EXEC.
TARMU3   CBL   TARMU3MO                  STR   284   204   8   021600   00217   TARMU3    021600  CURSOR FREEKB END-EXEC.
TARMU3   CBL   TARMU3MO                  STR   284   216   8   021600   00217   TARMU3    021600  CURSOR FREEKB END-EXEC.
TARMU3   CBL   EMPLOYEE-MASTER-RECORD    STR   200   136   5   022400   00225   TARMU3    022400 LENGTH(EMP-LENGTH)
TARMU3   CBL   EMPLOYEE-MASTER-RECORD    STR   200   141   6   022400   00225   TARMU3    022400 LENGTH(EMP-LENGTH)
TARMU3   CBL   EMPLOYEE-MASTER-RECORD    STR   200   147   5   022400   00225   TARMU3    022400 LENGTH(EMP-LENGTH)
TARMU3   CBL   EMPLOYEE-MASTER-RECORD    STR   200   152   5   022400   00225   TARMU3    022400 LENGTH(EMP-LENGTH)
TARMU3   CBL   EMPLOYEE-MASTER-RECORD    STR   200   136   5   022500   00226   TARMU3    022500 RIDFLD(EMP-ID)
TARMU3   CBL   EMPLOYEE-MASTER-RECORD    STR   200   141   6   022500   00226   TARMU3    022500 RIDFLD(EMP-ID)
TARMU3   CBL   EMPLOYEE-MASTER-RECORD    STR   200   147   5   022500   00226   TARMU3    022500 RIDFLD(EMP-ID)
TARMU3   CBL   EMPLOYEE-MASTER-RECORD    STR   200   152   5   022500   00226   TARMU3    022500 RIDFLD(EMP-ID)
TARMU3   CBL   EMPLOYEE-MASTER-RECORD    STR   200   136   5   022600   00227   TARMU3    022600 RESP(RESPONSE)
TARMU3   CBL   EMPLOYEE-MASTER-RECORD    STR   200   141   6   022600   00227   TARMU3    022600 RESP(RESPONSE)
TARMU3   CBL   EMPLOYEE-MASTER-RECORD    STR   200   147   5   022600   00227   TARMU3    022600 RESP(RESPONSE)
TARMU3   CBL   EMPLOYEE-MASTER-RECORD    STR   200   152   5   022600   00227   TARMU3    022600 RESP(RESPONSE)
TARMU3   CBL   EMPLOYEE-MASTER-RECORD    STR   200   136   5   022700   00228   TARMU3    022700 END-EXEC.
TARMU3   CBL   EMPLOYEE-MASTER-RECORD    STR   200   141   6   022700   00228   TARMU3    022700 END-EXEC.
TARMU3   CBL   EMPLOYEE-MASTER-RECORD    STR   200   147   5   022700   00228   TARMU3    022700 END-EXEC.
  ⋮
TARMU3B  JCL   ITSORS3.EMP.MASTER.ONLINE                 136   5   00310000   00030   TARMU3B   //EMPMAST DD DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR
TARMU3B  JCL   ITSORS3.EMP.MASTER.ONLINE                 141   6   00310000   00030   TARMU3B   //EMPMAST DD DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR
TARMU3B  JCL   ITSORS3.EMP.MASTER.ONLINE                 147   5   00310000   00030   TARMU3B   //EMPMAST DD DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR
TARMU3B  JCL   ITSORS3.EMP.MASTER.ONLINE                 152   5   00310000   00030   TARMU3B   //EMPMAST DD DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR
TARMU3B  JCL   ITSORS3.EMP.MASTER.ONLINE                 157   3   00310000   00030   TARMU3B   //EMPMAST DD DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR
TARMU3B  JCL   ITSORS3.DEPT.MASTER.ONLINE                 35   3   00320000   00031   TARMU3B   //DEPTMAST DD DSN=&HLQ..DEPT.MASTER.ONLINE,DISP=SHR
TARMU3B  JCL   ITSORS3.TARMU3B.INPUT                     136   8   00330007   00032   TARMU3B   //INPFILE DD DSN=&HLQ..TARMU3B.INPUT,DISP=SHR
TARMU3B  JCL   ITSORS3.TARMU3B.INPUT                     144   8   00330007   00032   TARMU3B   //INPFILE DD DSN=&HLQ..TARMU3B.INPUT,DISP=SHR
TARMU3B  JCL   ITSORS3.TARMU3B.INPUT                     152   8   00330007   00032   TARMU3B   //INPFILE DD DSN=&HLQ..TARMU3B.INPUT,DISP=SHR
TARMU3B  JCL   ITSORS3.TARMU3B.INPUT                     160   8   00330007   00032   TARMU3B   //INPFILE DD DSN=&HLQ..TARMU3B.INPUT,DISP=SHR
TARMU5   JCL   ITSORS3.EMP.MASTER.ONLINE                 157   1   00070022   00007   TARMU5       SORT FIELDS=(157,1,Y2D,A,158,2,BI,A)
TARMU5   JCL   ITSORS3.EMP.MASTER.ONLINE                 158   2   00070022   00007   TARMU5       SORT FIELDS=(157,1,Y2D,A,158,2,BI,A)
TARMU5   JCL   &&SORTOUT                                  37   3   00090022   00009   TARMU5       OUTREC FIELDS=(1,6,11,30,157,3,161X)
TARMU5   JCL   ITSORS3.EMP.MASTER.ONLINE                 136   5   00270000   00027   TARMU5    //SORTIN DD DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR
TARMU5   JCL   ITSORS3.EMP.MASTER.ONLINE                 141   6   00270000   00027   TARMU5    //SORTIN DD DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR
TARMU5   JCL   ITSORS3.EMP.MASTER.ONLINE                 147   5   00270000   00027   TARMU5    //SORTIN DD DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR
TARMU5   JCL   ITSORS3.EMP.MASTER.ONLINE                 152   5   00270000   00027   TARMU5    //SORTIN DD DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR
TARMU5   JCL   ITSORS3.EMP.MASTER.ONLINE                 157   3   00270000   00027   TARMU5    //SORTIN DD DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR
TARMU5   JCL   &&SORTOUT                                  37   3   00280000   00028   TARMU5    //SORTOUT DD DSN=&&SORTOUT,DISP=(NEW,PASS),SPACE=(CYL,(1,1)),
TARMU5   JCL   &&SORTOUT                                  37   3   00290000   00029   TARMU5    // DCB=(DSORG=PS,RECFM=FB,LRECL=200,BLKSIZE=4000)
TARMU5   JCL   &&SORTOUT                                  37   3   00400000   00035   TARMU5    //EMPMAST DD DSN=&&SORTOUT,DISP=(OLD,PASS)
TARMU5   JCL   &&SORTOUT                                  37   3   00520000   00044   TARMU5    //EMPMAST DD DSN=&&SORTOUT,DISP=(OLD,DELETE)
TARMU6   JCL   ITSORS3.EMP.MASTER.ONLINE                 136   5   00360000   00028   TARMU6    //EMPMAST DD DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR
TARMU6   JCL   ITSORS3.EMP.MASTER.ONLINE                 141   6   00360000   00028   TARMU6    //EMPMAST DD DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR
TARMU6   JCL   ITSORS3.EMP.MASTER.ONLINE                 147   5   00360000   00028   TARMU6    //EMPMAST DD DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR
TARMU6   JCL   ITSORS3.EMP.MASTER.ONLINE                 152   5   00360000   00028   TARMU6    //EMPMAST DD DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR
TARMU6   JCL   ITSORS3.EMP.MASTER.ONLINE                 157   3   00360000   00028   TARMU6    //EMPMAST DD DSN=&HLQ..EMP.MASTER.ONLINE,DISP=SHR
```

Figure 54 (Part 3 of 3). PC Export File (Year 2000 Analysis)

4.5.4 Accumulation of Results

Option 9 on the MA2000 main menu, **Accumulative result**, lets you combine the results of several multiple retrieval executions for a specific kind of element. The elements can be:

- Programs
- Jobs
- Includes
- Copybooks
- SQL includes in PL/I
- SQL includes in COBOL

You can print the results or write them to a file in the format of a detailed report or a PC export file.

MA2000 allocates some data sets with names like #ANAPFIX.SYS2000.systemID.* It uses these data sets as a buffer from where it retrieves the information needed to create the accumulative result file. If you want the results of a specific impact analysis execution to be in the accumulated report, you first have to add them to the accumulation buffer. To do that, type action P next to the corresponding results on the multiple **Retrieval Result Management** panel (Figure 48 on page 123) and provide a valid job card on the next panel as shown in Figure 55:

```
IMXP780---------------- Accumulate Year 2000 Analysis Results  ----------------
 Command ===>
                                                          Previous panel: PF3
                                                          Initial panel:  PF4

      To accumulate the retrieval result of the year 2000 analysis to
      the following system ID, press Enter.

          System ID: SYSITSO

      Job statement information:

  ==>  //ITSORS3A JOB ,
  ==>  //  MSGCLASS=H,MSGLEVEL=(1,1),NOTIFY=ITSORS3,
  ==>  //  TIME=(,59),REGION=64M
  ==>  //*
  ==>  //*
```

Figure 55. Accumulate Year 2000 Analysis Results Panel

Once the results from every analysis you have an interest in are in the buffer, you can execute the accumulative results function (Option 9 from the MA2000 Main Menu). From the accumulative results panel shown in Figure 56 on page 149 you have to select the type of element you want to get a report for. In addition, you can specify whether you want to retrieve all the elements or whether you want to select one single element from a list.

```
IMXP802--------------  Accumulative Results (Year 2000 Analysis)  -------------
 Menu ===>
                                                          Previous panel: PF3

       Select 1 (selective) or 2 (select all), and specify the type of
       output you want to obtain:

       Target system: SYSITSO


       1 Selective           Type: _

       2 Select all          Type: _

     Select the type option :
        P - PGM     T - TBL(PL/I)
        J - JOB     U - TBL(COBOL)
        I - INCLUDE O - OTHERS
        C - COPY
```

Figure 56. Accumulative Results Panel

On the last panel, shown in Figure 57 on page 150, you have to specify if you want to print the results or write them to a file. If you choose to create a file, you have to provide the name of a new data set that does not exist yet. MA2000 always allocates it as a sequential data set.

You can choose whether or not you want the entire detailed report or just the source statements. However, if you create a file and select the PC export file format, all data is written to the file.

```
IMXP785----------  Print or Export Accumulative Results (Selective)  ----------
 Command ===>
                                                          Previous panel: PF3
                                                          Initial panel:  PF4

    To print the selected members, specify the following items and
    press Enter.  (If you specify Y for the "Output to a file" option,
    the results are written into a file.)

    Report type:      1  (1: Whole report  2: Only source statements)
    Output class:     A
    Output to a file: Y (Y/N)   * If Y, specify the following items:
      Output type:     R (R: Detailed report  C: PC export file)
      Output data set: 'ITSORS3.MA2000.ACCU'
                       (Allocation is not required.  Only PS is allowed.)

    Job statement information:

  ==>  //ITSORS3A JOB ,
  ==>  //  MSGCLASS=H,MSGLEVEL=(1,1),NOTIFY=ITSORS3,
  ==>  //  TIME=(,59),REGION=64M
  ==>  //*
  ==>  //*
```

Figure 57. Print or Export Accumulative Results Panel

4.5.5 Date Identification File

If you apply the windowing technique in order to make your applications Year 2000 ready, the easiest and most reliable way to do it is to use IBM's Millennium Language Extensions. In this case, the date identification file is the most useful output produced by the Year 2000 analysis function of MA2000. Refer to 4.4, "Running the Year 2000 Analysis" on page 122 for information on how to create the date identification file.

The date identification file is used by CCCA to automatically add Millennium Language Extension syntax to your source programs and copybooks. We discuss the layout of the date identification file in detail in Chapter 6, "Using CCCA to Process the Results of the Find Phase" on page 165.

Use the reports and PC export file to verify the accuracy of your starting-point file. Make sure to add the correct date formats to the starting-point file. The comprehensiveness and correctness of the resulting date identification file depend on the quality of the starting-point file used for the Year 2000 analysis.

Chapter 5. Millennium Language Extensions to COBOL for OS/390 & VM

The term “Millennium Language Extensions” (MLE) is a collective reference to those facilities available in the COBOL compiler intended to provide assistance in the resolution of the problem posed by the pervasive use of two-digit year representations in application programs. This convention of using two digits rather than four to depict the year portion of a date relies upon the assumption that all such two-digit year fields represent years from 1900 to 1999. While this approach gave rise to few complications for most of the Twentieth Century, the advent of the next millennium makes it impossible to assume any longer that a two-digit year can automatically be assigned to this century. The mistaken inference that the two-digit year ″00″ represents the year ″1900″ rather than the year ″2000″ and therefore precedes rather than follows the two-digit year ″99″ will cause programs to yield incorrect results unless urgent steps are taken to correct this limitation.

The Millennium Language Extensions are designed to permit otherwise fully functional programs to overcome the restrictions imposed by their use of two-digit year fields and continue performing properly in the year 2000 and beyond, with minimal modifications to their existing code. This is accomplished by the extended use of the technique currently in use today, which is known as *century windowing*. A century window is a 100-year interval in which any two-digit year is uniquely represented. Rather than continuing the rigid application of the currently assumed century window of 1900-1999, however, this technique enables a two-digit year field to uniquely represent years within **any** given 100-year range.

If, for example, a two-digit year field contains the value 17, most current applications using two-digit year representations would apply the conventional century window of 1900-1999 in which all 2-digit years are assumed to lie and would infer the century value to be 19. The two-digit year would thus be interpreted as representing the year 1917. If, however, the century window range is defined to be 1956-2055, then the century value is interpreted to be 20 in accordance with where the two-digit year lies within the century window:

- Year values from 56 through 99 are interpreted as Years 1956-1999.
- Year values from 00 through 55 are interpreted as Years 2000-2055.

The Millennium Language Extensions to IBM′s COBOL compiler provide support for automated date windowing based upon information you supply about which dates should be windowed, and which century window should be applied when doing so. The compiler then automatically implements the associated windowing logic, in many cases without requiring any additional changes to the logic in the Procedure Division. Support for expanded 4-digit date fields to be used in conjunction with 2-digit windowed year fields allows you to implement the inevitable expansion of your 2-digit year fields incrementally at a pace dictated by your own requirements and resources.

5.1 MLE Within the Context of the VisualAge 2000 Find and Fix Paradigm

MLE is intended to address and accelerate the correction (″fixing″) of appropriate data items identified as candidates for windowing or expansion during the find process as defined in the VisualAge 2000 methodology.

Note

While MLE can help to reduce the scope of the logic changes in those programs for which windowing is an acceptable solution, MLE does not eliminate the necessity for the assessment, planning, analysis, implementation, and testing activities that constitute the various phases of the VisualAge 2000 methodology.

5.1.1 The Results of the Find Process as Input for MLE

Before specifying which data items represent what type of date to the compiler, the initial results of the find process should be subjected to a process of analysis whose results include:

- The elimination of those items that initially appeared to represent year fields requiring correction but in reality do not
- The identification of as many additional data items containing dates requiring correction as possible
- The identification of the date patterns of the date items
- The detection of those date data items for which century windowing represents an appropriate correction
- The identification of those date data items for which century windowing represents an unacceptable solution

Although MLE can also be used to expedite the find process by identifying potential date data items that might have been previously overlooked, this approach should be viewed as a complement to rather than a replacement for the find process referred to in this document and other items of VisualAge 2000 documentation. The technique of identifying unmistakable date data types to the compiler and allowing the compiler diagnostics to identify other candidates used in conjunction with these items is a helpful extension of the date identification process. However, it should be used as a reinforcement to rather than a substitution for the activities that make up the date identification processes such as those described in this book. Even though the programs of our sample application were relatively small and straightforward, we nevertheless adopted the approach of using the results of the find process as provided by MA2000 as a basis for our MLE-related activities rather than taking the simplistic alternative of allowing the compiler diagnostics to duplicate our results.

5.1.2 Using MLE to Fix Programs

While MLE offers you the possibility of extending the useful life of application programs for which the introduction of century windowing logic is an appropriate option, MLE cannot be used to rectify all potentially dangerous date items in all of your application programs. The reasons are these:

- MLE is applicable only for existing programs whose future usefulness is restricted only by their current inability to apply any other century window than one with a range of 1900-1999 when interpreting 2-digit year fields. MLE

should not be used in the repair of any other programs except those containing 2-digit year representations.

- In addition, not all the 2-digit year fields identified during the find process will be suitable candidates for century-windowing because of the context in which they are used in a program.
- Century windowing is also not an acceptable approach for programs whose date data items cannot be confined to a 100-year interval or for programs whose processing requirements cannot be met by the use of a single century window.
- When choosing MLE to assist you in the correction of programs in which a windowed date concept can be introduced, you must also be aware that required changes may not restrict themselves to the Data Division of a program. The identification of a date item to the compiler as a windowed date can potentially affect the behavior of all the statements in the program that reference these fields and necessitate changes in the Procedure Division of the program to resolve possible problems.
- MLE cannot be used for all levels of source code:
  - If your application programs have been compiled with the OS/VS COBOL compiler, they will require source conversion from the COBOL 68/74 standard to the COBOL 85 standard. The IBM COBOL and CICS Command Level Conversion Aid (CCCA) product can facilitate the conversion process dramatically, even inserting MLE date identification statements into your COBOL programs if desired. Specification of the program and date data item name, together with the appropriate date pattern is all that is required as input to this feature of CCCA. This task can be performed in conjunction with the conversion of COBOL 68/74 to the COBOL 85 level or as a separate activity when you wish simply to add MLE enhancements to source code already at the COBOL 85 level.

    See Chapter 6, “Using CCCA to Process the Results of the Find Phase” on page 165 for information on how to use CCCA to add MLE enhancements to your source code.
  - If your programs have been compiled with VS COBOL II Release 3 using the NOCMPR2 compiler option, you cannot use MLE. However, no conversion will be required.
- Although COBOL programs to which MLE windowing has been added can normally be used with any of the subsystems that currently support your applications, such as DB2, CICS, and VSAM, some functions specific to these subsystems do not permit the use of variables in whose data definitions the MLE enhancements have been used for date identification. Some examples include:
  - VSAM file primary or alternate keys
  - Search fields in database systems such as IMS or DB2
  - Sort fields in DB2 (ORDER BY...)
  - Key fields in CICS commands

  MLE does, however, offer functionality that allows the definition of variables identified to the compiler as dates to be superseded when necessary, although such functionality must be introduced explicitly in the Procedure Division of the affected program.

5.2 Software Prerequisites for MLE

The MLE features are available as a separate product which must be installed in addition to the IBM COBOL for OS/390 & VM or IBM COBOL for MVS & VM host compilers that support them:

- VisualAge COBOL Millennium Language Extensions for OS/390 & VM (5648-MLE) Version 1 Release 1
- VisualAge COBOL Millennium Language Extensions for MVS & VM (5654-MLE) Version 1 Release 1

In order for the respective MLE product to be enabled, PTFs are required for the base compiler, the run-time library, and the Debug Tool as well:

- APAR PQ12093 is necessary for COBOL for OS/390 & VM, resulting in a software level of Version 2 Release 1 Modification Level 1.
- APAR PQ12092 is necessary for COBOL for MVS & VM, resulting in a software level of Version 1 Release 2 Modification Level 2.
- APAR PQ12094 is necessary for Language Environment for OS/390, MVS and VM.
- APARs PQ06206 and PQ08277 are necessary for the IBM Debug Tool for MVS.
- APARs PQ13105 and PQ13149 are necessary for the IBM Debug Tool for VM.

Before attempting to incorporate the MLE enhancements in your application programs, you should first verify that your compilers are at the prerequisite level and that the appropriate MLE product has been installed.

5.3 The Advantages of Using MLE for Date Definitions and Century Windowing

Assuming that the introduction of windowing logic is suitable for date data items in your application programs, the use of MLE offers you the following benefits:

- The compiler supplies the logic necessary to perform century windowing automatically, largely eliminating the necessity of making manual modifications to code within the Procedure Division of a program.
- Because the windowing logic is implemented directly by compiler generated machine code, it is much more efficient than code written manually.
- The Millennium Language Extensions are easily activated and deactivated.
- The century window can be hard-coded or supplied as a value relative to the system date.
- Because changes to the Procedure Division of a program are kept to an absolute minimum by the definition of windowed dates, there are fewer changes to program logic and fewer code changes that must be tested.
- MLE augments the date identification process by identifying statements that involve date-sensitive processing and warns of inconsistencies or possible errors that might arise from the failure to define data items as dates now or in the future.
- By supporting a mixture of windowed (2-digit) and expanded (4-digit) dates, MLE can assist you in the gradual expansion of date fields by using the century window to automatically supply the century portion of the 4-digit date when a windowed date is copied to an expanded one. This feature eases

the transition to full field expansion by allowing you to expand the dates referenced in programs or stored in files and databases as your requirements and resources dictate.

5.4 The Limitations of Using MLE for Date Definitions and Century Windowing

These limitations hold:

- The start of the century window must lie within the 100-year interval of 1900-1999. This requirement reinforces the fact that the windowing feature of MLE is not to be implemented as a permanent solution to the problem posed by two-digit dates.
- Not all date formats are eligible for the automatic windowing capability of MLE. In order to qualify, a date field must contain a 2-digit year as the first or only part of the field. If the date to be windowed does not meet these requirements, the date cannot be defined to the compiler as a windowed date without the introduction of program code to isolate the year component from the rest of the date field. The remainder of the date field, if present, must be a minimum of two and a maximum of four characters or digits, although the content of the rest of the field is of no significance to MLE.
- Defining a date item to the compiler as a windowed date affects the behavior of every statement in the program that references this item. Many COBOL language elements are affected by the use of windowed date fields because they do not contain a complete item of information and their contents must be supplemented by information from the century window. In each case where a language element is affected by the definition of a data item as a date, a compiler diagnostic message is produced and corrective measures may be required to resolve the problem.
- MLE does not provide a fully specified set of date-oriented data types for new or existing applications. The only date-sensitive semantics provided by MLE involve the automatic expansion and contraction of the 2-digit year portion of dates on the basis of the century window in effect for the program. The remainder of the date field is not processed by MLE because of the potentially undesirable impact the introduction of such functionality might have on existing programs. Consequently, incrementing the windowed year field 99 will be interpreted as 1900 or 2000, depending upon the century window in effect, whereas incrementing 980430 by 1 will result in a value of 980431 and not 980501, if the extensions provided date data types.

5.5 The Components of MLE

The term MLE represents an aggregate of compiler options and COBOL language elements which have been introduced to assist you in achieving a solution to your Year 2000 date-related logic problems.

5.5.1 MLE-Related Compiler Options

The compiler options introduced in MLE are used to activate and deactivate special date-oriented processing of fields which have been designated as date fields and to define the century window to be applied when processing 2-digit windowed dates.

5.5.1.1 The DATEPROC Compiler Option

The DATEPROC option is used to enable the Millennium Language Extensions of the COBOL compiler. These extensions must be explicitly activated prior to compilation, however, as the default value for this option is NODATEPROC. The DATEPROC compiler option can be specified with a PROCESS (or CBL) statement preceding the Identification Division header of your COBOL program, as Figure 58 illustrates:

```
CBL DATEPROC(FLAG) YEARWINDOW(1935) NODYNAM QUOTE SOURCE LIST
IDENTIFICATION DIVISION.
PROGRAM-ID.   TARMU6E
```

Figure 58. Specifying the DATEPROC Compiler Option

If only the DATEPROC compiler option is specified, then the default value is DATEPROC(FLAG). When this option is in effect, the compiler produces diagnostic messages for **every** statement that defines or references a date field. While these messages will normally represent information-level messages, it is entirely possible for warning-level messages to be generated for COBOL code that appears syntactically correct and even produces correct results. This is particularly the case when using dates in connection with nondate fields. Error-level messages are also produced when there are inconsistencies or errors in the use of date-related program constructs.

After compiling and testing your program satisfactorily, the DATEPROC(NOFLAG) option can be used to produce a listing with the minimum of diagnostic messages. When this option is in effect, messages are generated only when there are errors or inconsistencies in the code.

Note

Use of the DATEPROC option requires that the NOCMPR2 option is also in effect.

Specifying the default value of NODATEPROC indicates that MLE features are not in effect for this compile unit if they are present. The ease with which this compiler option can be specified allows you to introduce MLE date-related constructs into your program, compile and test the program, and deactivate the changes until a specified future date by recompiling. Such an approach, however, assumes that no changes have been introduced in the Procedure Division of the program that would produce inconsistent or unpredictable results after the Millennium Language Extensions have been temporarily disabled.

5.5.1.2 The YEARWINDOW Compiler Option

The YEARWINDOW compiler option is used to define the first year of the century window which is to be applied to 2-digit windowed date fields by the compiler. A century window is a 100-year interval within which any 2-digit year is unique. An unsigned decimal number between 1900 and 1999 is used to establish a fixed century window. The value of the YEARWINDOW option in Figure 59 on page 157 defines a century window beginning in 1935 and ending in 2034:

```
CBL DATEPROC(FLAG) YEARWINDOW(1935) NODYNAM QUOTE SOURCE LIST
IDENTIFICATION DIVISION.
PROGRAM-ID.   TARMU6E
```

Figure 59. Specifying the YEARWINDOW Compiler Option (Fixed Window)

A negative integer from -1 through -99 is used to indicate a sliding window, the first year of which is calculated by subtracting the specified value from the current year of the current run-time date. Figure 60 illustrates how to specify a century window which begins 63 years before the year in which the program is executed:

```
CBL DATEPROC(FLAG) YEARWINDOW(-63) NODYNAM QUOTE SOURCE LIST
IDENTIFICATION DIVISION.
PROGRAM-ID.   TARMU6E
```

Figure 60. Specifying the YEARWINDOW Compiler Option (Sliding Window)

Note

The DATEPROC compiler option must be enabled in order for the YEARWINDOW option to take effect.

When the program is executed, two restrictions must be observed:

1. The first year of the 100-year interval defined by the YEARWINDOW option must lie in the range 1900-1999.
2. The current year must lie within the century window in effect for the compile unit.

   Only one century window is in effect at any given time for a particular program.

The ease with which the YEARWINDOW compiler option can be set also facilitates the testing of your programs. A value of 1900 for the YEARWINDOW option is ideal for performing regression testing after you have introduced MLE date constructs into your program and should produce the same results as you currently receive.

5.5.2 MLE Language Elements

The Millennium Language Extensions, when they are enabled by the DATEPROC compiler option, include the following COBOL language elements:

- Date format clause
- Intrinsic functions returning a date field
- Conceptual data items
- Date reinterpretation functions

5.5.2.1 Date Format

The DATE FORMAT clause is added exclusively to data definitions in the Data Division of a program in order to specify that the items contain a date in a form supported by MLE. The date format specified in the clause defines the date pattern of a date field, thereby indicating the location of the year component within the date. The number of digits specified for the year portion defines the

type of date represented by the field, and thereby indicates how it is to be treated by the compiler. The two possible types of date fields are:

- **windowed date field** — A windowed date field is a date field that contains a windowed year. A windowed year consists of 2 digits, representing a year within the specified century window.
- **expanded date field** — An expanded date field is a date field that contains an expanded year. An expanded year consists of 4 digits. The primary use of expanded date fields is to provide consistent results when they are used in conjunction with windowed date fields. MLE automatically fills in the leftmost 2 digits representing the century by applying the century window when the contents of a windowed date field are copied into an expanded date field. This support of expanded and windowed date fields in combination is intended for situations in which the migration from 2- to 4-digit dates is not complete.

The Expansion of Windowed Dates: If a windowed date field is used in one of the following contexts, it is treated as if it were first converted to expanded date format:

- Operands in arithmetic or conditional (IF, EVALUATE WHEN, SEARCH WHEN, or PERFORM UNTIL) expressions
- Operands in ADD or SUBTRACT statements
- A sending field in COMPUTE or MOVE statements
- A KEY argument of a SORT or MERGE statement

Given a century window starting year of 19xx, the year component (yy) of a numeric windowed date field is treated as if it were expanded as follows:

- If yy is less than xx, then add 2000 to yy
- If yy is equal to or greater than xx, then add 1900 to yy

This is illustrated by the example in Figure 61.

```
01  TARDATE-PARAMETERS.
    03  TARDATE-DATE        PIC X(8).
    03  TARDATE-DATE-RED3 REDEFINES TARDATE-DATE.
        05  TARDATE-DATE-MMDD       PIC X(6).
        05  TARDATE-DATE-YY         PIC 9(2)
                                    DATE FORMAT YY.
⋮
    ADD 1 TO  TARDATE-DATE-YY.
```

Figure 61. An Example of the Expansion of Windowed Date Fields

If the century window is 1935-2034 and the value of TARDATE-DATE-YY is 1999, then the computation takes place as if TARDATE-DATE-YY is first incremented by 1900 to result in 1999. The ADD operation is then performed, giving a result of 2000. The value 00 is then stored in windowed date field TARDATE-DATE-YY.

Alphanumeric windowed date fields are dealt with in a similar fashion, but a prefix of either ″19″ or ″20″ is added rather than performing the addition of 1900 or 2000.

Supported Date Formats: The date format clause is used to specify the date pattern of a date field. Table 28 on page 159 illustrates which patterns are supported.

Table 28. Date Formats Supported by MLE	
Date Format	**Contents**
YY	A windowed year
YYXX	A windowed year followed by 2 characters, perhaps representing a month
YYXXX	A windowed year followed by 3 characters, perhaps representing a day of the year
YYXXXX	A windowed year followed by 4 characters, perhaps representing a month and the day of the month
YYYY	An expanded year
YYYYXX	An expanded year followed by 2 characters
YYYYXXX	An expanded year followed by 3 characters
YYYYXXXX	An expanded year followed by 4 characters

Using Unsupported Date Patterns: The DATE FORMAT clause can only be used to define date fields whose year component is the first or only part of the field. If your application contains date fields that do not meet these requirements, some additional code must be supplied in order to set apart the portion of the date field which represents the year. Figure 62 illustrates a Gregorian date field whose pattern is unsupported by the DATE FORMAT clause:

```
01  LAST-TARDATE-DATE         PIC 9(6).
01  NEXT-TARDATE-DATE         PIC 9(6).
⋮
    ADD 1 TO LAST-TARDATE-DATE GIVING NEXT-TARDATE-DATE.
```

Figure 62. An Example of Unsupported Date Patterns

In this example, if LAST-TARDATE-DATE contained a value of 280352, the operation would result in a new value for NEXT-TARDATE-DATE of 280353. If, however, LAST-TARDATE-DATE had a value of 280399, the addition of 1 would result in a value of 280400, that is not the expected result. Figure 63 demonstrates the changes that must be made to isolate the year portion of the two Gregorian dates so that they can be identified to the compiler as windowed date fields

```
01  LAST-TARDATE-DATE.
    05  LAST-TARDATE-DATE-DDMM       PIC 9(4).
    05  LAST-TARDATE-DATE-YY         PIC 9(2) DATE FORMAT YY.
01  NEXT-TARDATE-DATE.
    05  NEXT-TARDATE-DATE-DDMM       PIC 9(4).
    05  NEXT-TARDATE-DATE-YY         PIC 9(2) DATE FORMAT YY.
⋮
    MOVE LAST-TARDATE-DATE-DDMM TO NEXT-TARDATE-DATE-DDMM.
    ADD 1 TO LAST-TARDATE-DATE GIVING NEXT-TARDATE-DATE.
```

Figure 63. Isolating the Year Portion of Unsupported Date Patterns

The data definitions of the date fields have been expanded so that the year component is the only part of the field identified as a date field by the DATE FORMAT clause. The same arithmetic operation will now produce the desired result of 280300.

5.5.2.2 Intrinsic Functions Returning a Date Field

The enabling of the Millennium Language Extensions also results in the returned values of the following intrinsic functions being treated as if they were expanded date fields with implicit date formats. If the DATEPROC compiler option is in effect, MLE reinterprets the return values of the following functions as a date field as shown in Table 29.

Table 29. Implicit Date Formats of Intrinsic Function Return Values

Function	Reinterpreted Return Value
DATE-OF-INTEGER	Expanded date field with implicit DATE FORMAT YYYYXXXX
DATE-TO-YYYYMMDD	Expanded date field with implicit DATE FORMAT YYYYXXXX
DAY-OF-INTEGER	Expanded date field with implicit DATE FORMAT YYYYXXX
DAY-TO-YYYYDDD	Expanded date field with implicit DATE FORMAT YYYYXXX
YEAR-TO-YYYY	Expanded date field with implicit DATE FORMAT YYYY
YEARWINDOW	Expanded date field with implicit DATE FORMAT YYYY

If the variables in which the values returned by these intrinsic functions are stored are not defined as date fields by means of a DATE FORMAT clause, a compiler diagnostic message is issued to highlight the inconsistency.

The YEARWINDOW intrinsic function has been introduced by the Millennium Language Extensions. When enabled, it returns the starting year of the century window specified by the YEARWINDOW compiler option. This value can be used to ensure that the century windows used by automatic date windowing, COBOL intrinsic functions, and Language Environment Callable Services are consistent. See Figure 89 on page 201 for an example.

5.5.2.3 Conceptual Data Items

Enabling MLE also results in the values returned by the conceptual data items DATE, DATE YYYYMMDD, DAY and DAY YYYYDDD being regarded as having an implicit DATE FORMAT. If the DATEPROC compiler option is in effect, MLE reinterprets the return values of the following functions as if they had a DATE FORMAT clause as Table 30 illustrates.

Table 30. Implicit Date Formats of The Accept Statement

Accept Statement	Implicit Date Format of Return Value
ACCEPT identifier FROM DATE	Implicit DATE FORMAT YYXXXX
ACCEPT identifier FROM DATE YYYYMMDD	Implicit DATE FORMAT YYYYXXXX
ACCEPT identifier FROM DAY	Implicit DATE FORMAT YYXXX
ACCEPT identifier FROM DAY YYYYDDD	Implicit DATE FORMAT YYYYXXX

Note

When you modify existing programs to take advantage of MLE, you must be aware that the compiler will generate an error-level diagnostic message if the data item into which the return value is accepted does not have a DATE FORMAT clause whose date format matches the date type specified in the ACCEPT statement.

5.5.2.4 MLE Intrinsic Functions

MLE introduced three new intrinsic functions as IBM COBOL language elements. Two of them, DATEVAL and UNDATE, are used to redefine date fields and nondates precisely as the usage of the data item in a program demands. The YEARWINDOW intrinsic function returns the value specified in the YEARWINDOW compiler option to the program.

The DATEVAL Intrinsic Function: The DATEVAL intrinsic function is used to convert a nondate into a date. Any of the following qualify as a nondate:

- An alphanumeric or numeric data item whose data definition does not include a DATE FORMAT clause
- A literal (991231,19560121)
- A reference-modified date field such as WORK-DATE-YYMMDD(1:2)
- A date field that has been converted to a nondate using the UNDATE intrinsic function
- The results of certain arithmetic operations that may include date field operands, such as the difference of two compatible date fields

Depending on the context in which you use a nondate in conjunction with a date field, the compiler is forced to make certain assumptions about the nondate field. The nondate is either assumed to be compatible with the date field in the number of year and nonyear characters or is treated as a simple numeric value contingent upon the type of statement involved:

- In comparison operations where a date field is compared to a nondate, the nondate is assumed to be compatible with the date field in the number of year and nonyear characters. Not only is the nondate considered to have the same number of year and nonyear digits as the date field, if the date-field is a windowed date, the compiler applies the assumed century window of 1900-1999 to the nondate. This can result in an inconsistent comparison if the YEARWINDOW compiler option does not specify the same century window. The DATEVAL function can be used to ensure that the compiler applies the same century window to the nondate as that applied to windowed dates in the program. As the compiler generates a warning-level message whenever a nondate is compared to a date, even when the assumed window of 1900-1999 is the correct assumption, it is advisable to make the year portion of the nondate explicit and thereby eliminate the warning-level message that accompanies application of the assumed century window.
- When a nondate is moved to a date-field, the sending field is also assumed to be compatible with the receiving date field in the number of year and nonyear characters. When, for example, you move a nondate to a windowed date field, the nondate field is assumed to hold a date with the same number of nonyear characters and a 2-digit year. Using the DATEVAL function will eliminate the warning-level message that accompanies this assumption.
- In all supported arithmetic operations involving date fields, nondates are treated as simple numeric values. The DATEVAL intrinsic function can be used to convert a nondate into an explicit date in supported arithmetic operations involving dates.

The UNDATE Intrinsic Function: The UNDATE intrinsic function is used to convert a date field into a nondate when you wish to treat a date field as a nondate. Use of this function is appropriate when you wish to:

- Circumvent certain subsystem restrictions on the use of windowed date fields.
- If you do not wish windowing to be applied when comparing the value of a date field to a literal in a comparison statement.
- When you wish to use a date as an argument to an intrinsic function.

The YEARWINDOW Intrinsic Function: When enabled, the YEARWINDOW intrinsic function returns the starting year of the century window specified by the YEARWINDOW compiler option. This value can be used to ensure that the century windows used by automatic date windowing, COBOL intrinsic functions and Language Environment Callable Services are consistent. See sample program TARDTE3X for an example.

5.6 An Outline of the MLE Implementation Process

This section presents a proposed outline of the steps to be followed when you have adopted windowing as your chosen approach and wish to take advantage of the date processing capabilities offered by the Millennium Language Extensions. The tasks outlined here are intended only as a suggested blueprint and may require modification in order to conform to the procedures in effect at your installation.

1. Decide on a century window for the program to be modified. You should take into consideration how far back into the past your historical data goes and how far into the future it will extend. If, for example, the program is processing a large number of date fields containing dates before the year 1950, it would be unwise to select a century window of 1960 - 2059 since any 2-digit year less than 60 will be interpreted as belonging to the Twenty-First Century. Remember as well that the choice of a century window must be made for each program.
2. Notify the compiler that you wish to enable the Millennium Language Extensions by using the DATEPROC compiler option. If you wish compiler diagnostic messages related to date-fields to be produced, use the DATEPROC(FLAG) option.
3. Inform the compiler about the type of century window you have decided to use by using the YEARWINDOW compiler option to set the century window. For a fixed window, specify a 4-digit year between 1900 and 1999. To define a sliding window, specify an integer from -1 to -99, bearing in mind that the first year of the window must lie within the Twentieth Century.

   These options can be specified in the PROCESS (or CBL) statement preceding the Identification Division header in your program. An alternative to actually changing the compiler options in the program is to specify those compiler options which might initially require frequent modification in the PARM parameter on the EXEC statement in your compilation JCL as Figure 64 on page 163 illustrates.

```
//ITSORS3C JOB ,
//             MSGLEVEL=(1,1),MSGCLASS=H,TIME=(,59),
//             NOTIFY=&SYSUID,USER=&SYSUID
//COBOL  EXEC PGM=IGYCRCTL,PARM='DATEPROC(FLAG),YEARWINDOW(1956)'
//             REGION=2048K
//STEPLIB  DD  DSNAME=IGYV2R10.SIGYCOMP,DISP=SHR
//         DD  DSNAME=IGY.V2R1MA.SIGYCOMP,DISP=SHR
```

Figure 64. Sample Specification of Compiler Options in the JCL EXEC Statement

4. Identify dates that can be windowed and those for which expansion is the only possible alternative.
5. Add an appropriate DATE FORMAT clause to the data description entries of those data items within the program that should be recognized by the compiler as windowed or expanded dates.
6. Expand windowed dates by using MOVE or COMPUTE statements to copy their contents to expanded date fields. MLE assists you in date expansion by automatically supplying the century portion of the expanded date by applying the century window. Don't forget to make any changes necessary to the data definitions of items into which dates are accepted from conceptual data items by means of an ACCEPT statement.
7. Convert between date and nondate fields as the context requires by using the DATEVAL and UNDATE intrinsic functions.
8. Compile the program with the DATEPROC(FLAG) option and review diagnostic messages to see if date processing has produced any unexpected results or if the modifications performed violate any language constraints.
9. Eliminate severe error, error- and warning-level compiler diagnostic messages. Pay particular attention to warning-level messages indicating that the assumed century window is being applied or that a nondate is used in conjunction with a date-field.
10. Recompile your program with DATEPROC(NOFLAG) when only information-level diagnostic messages are generated by the compiler.
11. Run regression tests using the default YEARWINDOW value of 1900, which should produce the same results as were produced before the date-related modifications were introduced.
12. Add the YEARWINDOW and DATEPROC options to the PROCESS or CBL statement which precedes the Identification Division in your program.
13. Run date simulator tests using the YEARWINDOW value specified in Step 3 above.

Chapter 6. Using CCCA to Process the Results of the Find Phase

This chapter describes how you can use the output which results from the use of the MA2000 product during the find step to obtain input for the corrective measures performed during the fix process.

Step 5 of the process described in 5.6, “An Outline of the MLE Implementation Process” on page 162 consists of the addition of a DATE FORMAT clause to the data description entries of all data items that are to be treated as windowed or expanded dates. You can perform this task automatically by using CCCA. This saves you a lot of time and effort and provides you with a preliminary fix of your programs using MLE enhancements.

The approaches toward a final fix as described in Chapter 7, “Using MLE Enhancements to Implement Sample Year 2000 Solutions” on page 177 represent the MLE ″fine tuning.″ How many changes you have to make in the Procedure Division of your programs depend on your program code. The MLE-related compiler messages provide you with the necessary information about what additional changes you have to make.

Besides the program source CCCA only needs a date identification file (DIF) as input in order to add the DATE FORMAT clauses. MA2000 for example can generate this date identification file (see 4.4, “Running the Year 2000 Analysis” on page 122 and 4.5.5, “Date Identification File” on page 150).

6.1 Overview of CCCA

COBOL and CICS Command Level Conversion Aid helps you convert old COBOL 68 Standard and COBOL 74 Standard language in source programs and copybooks to COBOL 85 Standard language. Because MLE requires the new Language Environment-based IBM COBOL compilers that only support the COBOL 85 Standard and higher, it might be necessary to convert your programs to this level before you can apply MLE. However, such a conversion is not directly related to the tasks performed in order to make a program Year 2000 ready. Therefore in this book we don′t explain in detail the standard conversion capabilities of CCCA.

6.1.1 Converting Language Standard

The input to CCCA are the program sources and copybooks in the source language level. The output of CCCA are the program sources and copybooks in the target language level. The possible source and target language levels are illustrated on the **CCCA Language Level panel** shown in Figure 65 on page 166:

```
---------------------------- CCCA Language Level -------------------------------
Command ===>

   Source language level ===> _   1. DOS/VS COBOL LANGLVL(1)
                                  2. DOS/VS COBOL LANGLVL(2)
                                  3. OS/VS COBOL LANGLVL(1)
                                  4. OS/VS COBOL LANGLVL(2)
                                  5. VS COBOL II Release 1.0  1.1  2.0, or
                                     any COBOL with the CMPR2 option
                                  6. VS COBOL II NOCMPR2 Release 3.0  3.1  3.2
                                  7. VS COBOL II NOCMPR2 Release 4.0
                                  8. COBOL/370 NOCMPR2
                                  9. COBOL/VSE NOCMPR2
                                 10. COBOL for MVS and VM NOCMPR2
                                 11. COBOL for OS/390 and VM NOCMPR2


   Target language level ===> _   1. VS COBOL II
                                  2. COBOL/VSE
                                  3. COBOL for MVS and VM
                                  4. COBOL for OS/390

PF1 Help   F3 Exit   F4 Return   ENTER Save options
```

Figure 65. CCCA Language Level Panel

CCCA identifies COBOL language elements and CICS commands in the input source programs that are not supported by the target language or that are supported in a different manner. It then does one of the following:

- Converts them to the equivalent in the target language
- Removes them
- Flags them.

In addition, CCCA

- Removes or converts the Base Locator for Linkage (BLL) section mechanism and references
- Eliminates conflicts between user-defined names and words reserved for new COBOL.

6.1.2 Adding MLE Syntax

The DATE FORMAT conversion option of CCCA adds a DATE FORMAT clause to selected data description entries. It is not a conversion from one COBOL standard level to another level. However, CCCA does perform the DATE FORMAT conversion in addition to any other conversion required for converting to a different level of COBOL. If you select the DATE FORMAT option, you have to specify an MLE supporting target-language level. MLE is supported by the target-language levels 2, 3, and 4 on the CCCA language level panel (see Figure 65).

If your program has been written using a level of COBOL that supports the DATE FORMAT clause but the program source does not include the DATE FORMAT clause, you can use CCCA to perform the DATE FORMAT conversion only. In this case, you specify the same level of COBOL for both the source and target languages.

CCCA does not, itself, identify which data items within a COBOL program are used to contain dates. Instead, CCCA requires the names and format of each of these data items to be supplied as additional input (date identification file).

6.1.3 How CCCA Works

CCCA is a combination of an interactive system using ISPF panels and a batch application. You use CCCA online ISPF panels to:

- Define the type of conversion you want
- Submit a batch job to convert your programs

Figure 66 on page 168 shows the three phases of a conversion job.

Phase 1: Analyze input source

At the start of a conversion job, Phase 1:

- Extracts copy members from the appropriate copy libraries and merges them with the source program.
- Translates the original source program and copy books into a set of character strings known as *tokenized source*. A token record is created for each:
  - COBOL word
  - literal
  - picture character-string
  - separator
  - line of comment paragraphs in ID Division
- For each language element in the tokenized source, identifies whether conversion is required and, if so, which Language Conversion Program (LCP) to use.
- Checks whether each COBOL word in the input source program is in the COBOL Reserved Word data set or not.

Phase 2: Create change requests

For each item that needs converting, Phase 2:

- Reads the tokenized source.
- Loads and runs the LCP that is indicated by the code, when a token record is encountered that has something other than change code 999.
- Generates detailed change requests for converting the input source program.

Phase 3: Apply changes and generate output

Finally, Phase 3:

- Applies the change requests from Phase 2, creating new source programs and, if required, new copy members.
- Generates the diagnostic listing.

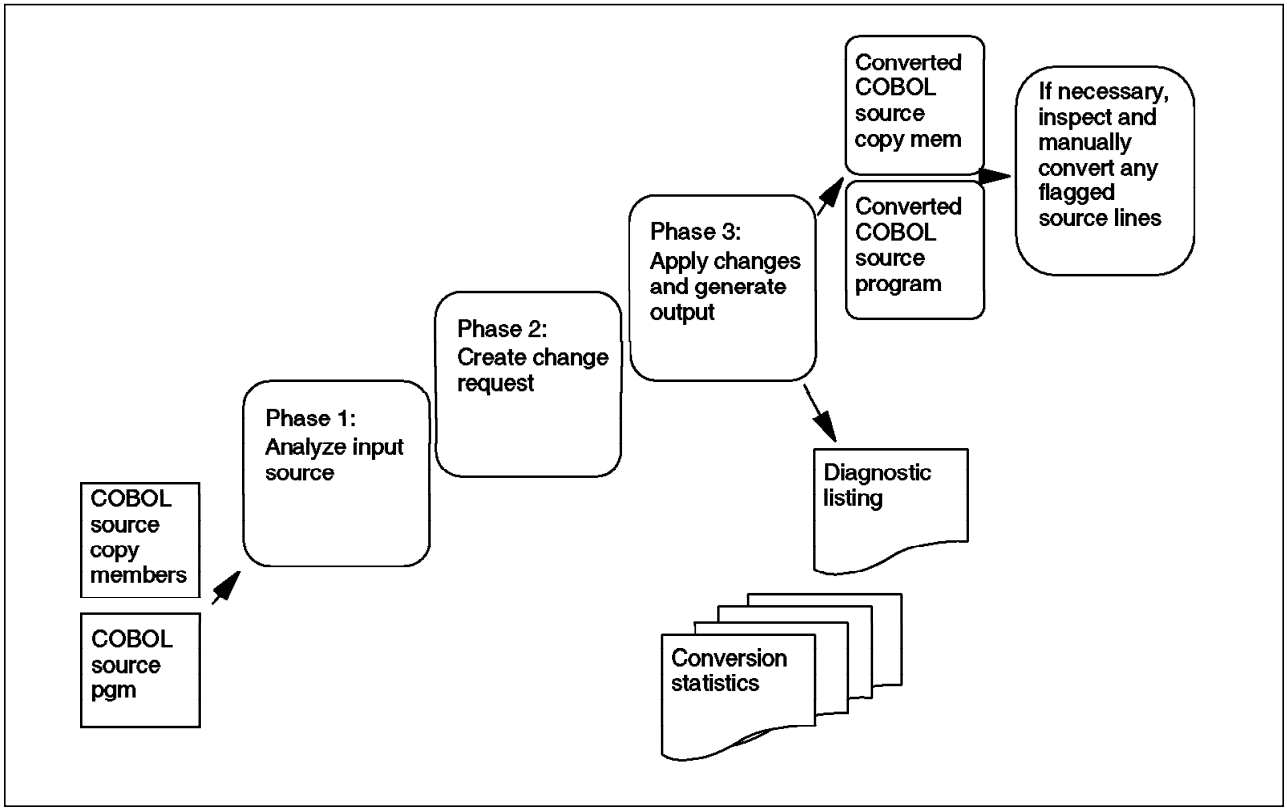


Figure 66. CCCA Conversion Phases

You can change the language conversion programs provided by CCCA if you need to — for example, if a particular language element must be converted differently for your environment. Or you can write your own LCP — for example, if you have additional language elements (even non-COBOL) that need to be converted, flagged, or removed. An LCP is a COBOL-like program. The source of the standard LCPs is shipped with CCCA as samples.

6.2 Specify the Conversion Options

When you invoke CCCA for the first time, you have to specify the environment options before you can do anything else. From the **CCCA Options Menu** as shown in Figure 67 on page 169, select **Option 1**.

```
------------------------------ CCCA Options Menu -------------------------------
Option ===>

   1  ENVIRONMENT    - Set environment options

   2  LANGUAGE       - Set language level

   3  CONVERSION     - Set conversion options 1

   4  CONVERSION     - Set conversion options 2









PF1 Help   PF3 Exit   PF4 Return
```

Figure 67. CCCA Options Menu

CCCA stores all the information about your conversions, including the data used to create the conversion statistics, in VSAM and other data sets. On the environment options panel as shown in Figure 68, you have to specify the high-level qualifiers you want CCCA to use when it creates these files, as well as a valid job card for the batch jobs.

```
-------------------------- CCCA Environment Options ---------------------------
COMMAND ===>

   High level qualifiers:
     Non-VSAM Shared Data Sets .  ===> IGY.CCCA
     NON-VSAM Private Data Sets   ===> ITSORS3
     VSAM Shared Data Sets .....  ===> IGY.CCCA
     VSAM Private Data Sets ....  ===> ITSORS3

   UNIT for Work Files .......... ===> SYSDA
   CLIST debugging .............. ===> N   Y/N

Job statement information:      (Verify before proceeding)
  ===> //ITSORS3C JOB ,
  ===> //             MSGLEVEL=(1,1),MSGCLASS=H,TIME=(,59),
  ===> //             NOTIFY=&SYSUID,USER=&SYSUID
  ===>


SYSOUT CLASS ===> *


PF1 Help   PF3 Exit   PF4 Return   ENTER Save options
```

Figure 68. CCCA Environment Options Panel

The only option we are interested in for the DATE FORMAT conversion is on the second conversion option panel (Option 4 of the CCCA Options Menu in Figure 67). Specify **Y** for **Option 8**, **Add DATE FORMAT clause to date fields** as shown in Figure 69 on page 170. If you don't perform a language standard conversion at the same time, leave the default values for all the other options.

```
------------------------ CCCA Conversion Options 2 --------------------------
Command ===>
Option
     1. Check procedure names . . . . . .          ===> Y   Y/N
     2. Flag Report Writer statements . . . . . . . ===> Y   Y/N
     3. Remove obsolete elements. . . . . . . . . . ===> Y   Y/N
     4. Negate implicit EXIT PROGRAM. . . . . . . . ===> Y   Y/N
     5. Generate END PROGRAM header . . . . . . . . ===> N   Y/N
     6. Compile after converting. . . . . . . . . . ===> Y   Y/N
     7. Flag manual changes in new source programs. . ===> Y   Y/N
     8. Add DATE FORMAT clause to date fields . . . . ===> Y   Y/N
     9. Remove VALUE clauses in File/Linkage Sections ===> Y   Y/N
    10. Flag IF FILE-STATUS (NOT) = "00". . . . . . . ===> Y   Y/N
    11. Flag BLL cell arithmetic. . . . . . . . . . ===> Y   Y/N
    12. BLL cell conversion method. . . . . . . . . ===> A   A/B
    13. Search source for literal delimiter . . . . . ===> Y   Y/N
    14. Literal delimiter (QUOTE or APOST). . . . . . ===> Q   Q/A
    15. . . . . . . . . . . . . . . . . . . . . . . ===> N   Y/N


  Note: Option numbers appear on the Program/File report

PF1 Help   PF3 Exit   F4 Return   ENTER Save options
```

Figure 69. CCCA Conversion Options 2 Panel

If you specify **Y** for Option 6, **Compile after converting**, CCCA compiles the converted source. Although, it does not save the object module, you get the compiler messages. This is especially useful when you perform a DATE FORMAT conversion, in which case you can immediately see all the MLE messages. If **Option 8** and **Option 6** are both set to **Y**, CCCA automatically uses DATEPROC(FLAG) YEARWINDOW(1900) as compiler options.

6.3 Perform the Conversion

To run the conversion, select **Option 1**, **Convert**, from the **CCCA Master Menu** shown in Figure 70 on page 171 and press Enter.

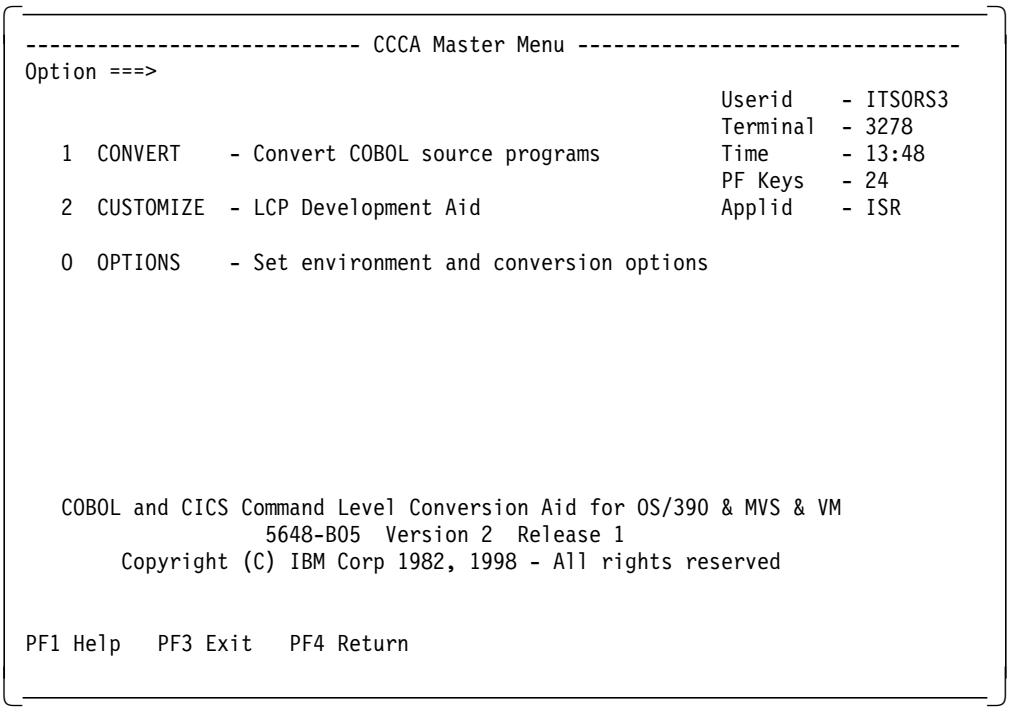


Figure 70. CCCA Master Menu

On the **CCCA Converter Menu** (Figure 71) select **Option 2**, **CONVERT PROGRAM - Convert COBOL source programs**, and press Enter.

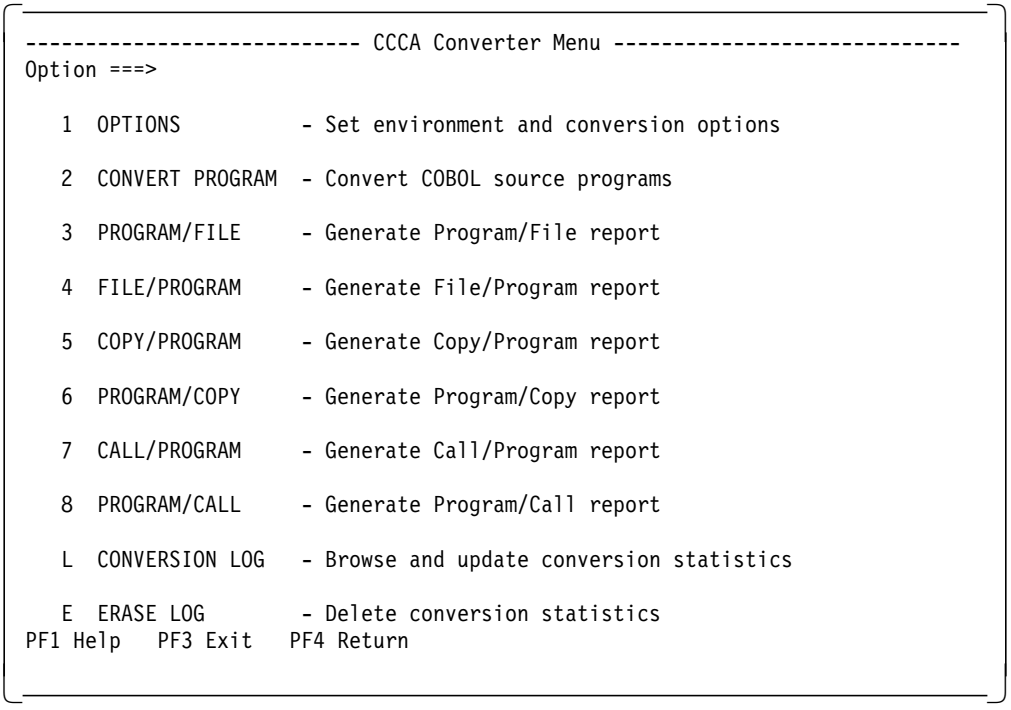


Figure 71. CCCA Converter Menu

The conversion job statement information panel is displayed as shown in Figure 72 on page 172. Specify a valid job card for your installation and press enter.

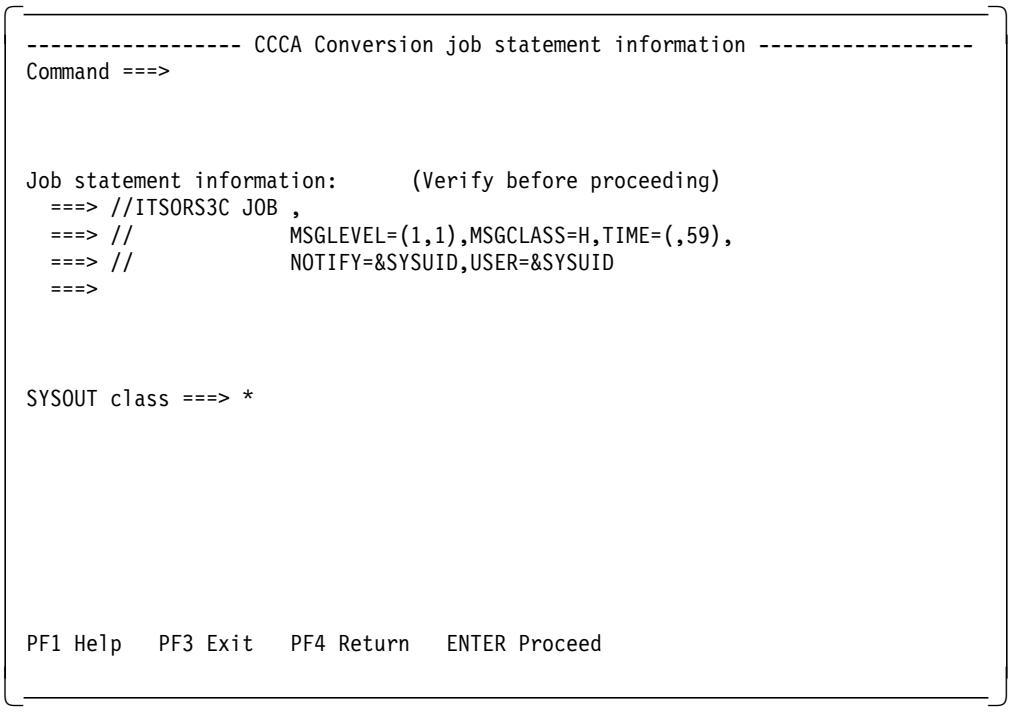


Figure 72. CCCA Conversion Job Statement Information Panel

On the conversion selection panel shown in Figure 73 on page 173, you have to specify:

- The name of the library that contains your program sources.

  You can specify a single input program, all programs of the input library (*), or you can get a list of the programs from which you can select the members you want to convert.
- The names of the libraries that contain your copybooks.
- The name of the library where you want CCCA to put the converted program sources. These data sets must be allocated beforehand.
- The name of the library where you want CCCA to put the converted copybooks. These data sets must be allocated beforehand.
- The name of your date identification file. This input field is only on the panel if you select the DATE FORMAT conversion option (see 6.2, “Specify the Conversion Options” on page 168). You can have one date identification file that contains the information for all the programs. If you specify a PDS without a member name, CCCA assumes that you have a separate date identification file member for every program. In this case the DIF member name has to be the same as the program name.

```
------------------------ CCCA Conversion selection ----------------------------
Command ===>
Program source:                            Options:
   Project . . . ===> ITSORS2                 Language level ===> *    (* 1-11)
   Library . . . ===> SAMPAPPL                CICS . . . . . ===> Y    (Y N)
   Type. . . . . ===> COBOL                   SQL. . . . . . ===> N    (Y N)
   Member. . . . ===> *        (Blank for member list, * for all members)

Other source file:
   Data set name  ===>
Copy libraries:
DDNAME ===> SYSLIB   LIBRARY ===> 'ITSORS2.SAMPAPPL.COPYBOOK'
       ===>                  ===>
       ===>                  ===>
       ===>                  ===>
       ===>                  ===>
       ===>                  ===>
Output source:
   Program library ===> 'ITSORS2.CCCA.COBOL'
   Copy library. . ===> 'ITSORS2.CCCA.COPYBOOK'
Date identification file:
   Data set name*  ===> 'ITSORS3.MA2000.DIF.YEAR2000'
*If PDS without member name, then program source member names used.
PF1 Help   PF3 Exit   PF4 Return   ENTER Build JCL
```

Figure 73. CCCA Conversion Selection Panel

When you press Enter, the conversion submission panel is displayed as shown in Figure 74. The important information on this panel is the number of members selected for the conversion.

```
------------------------ CCCA Conversion submission ---------------------------
Command ===>


Instructions:
   Press ENTER  to continue generating JCL.
   Press PF3    to submit job and exit
   Press PF4    to submit job and return
   Press PF12   to exit without submitting job
   Enter Cancel command to exit without submitting job.

         5 member(s) built for conversion.



Job statement information:
      //ITSORS3C JOB ,
      //             MSGLEVEL=(1,1),MSGCLASS=H,TIME=(,59),
      //             NOTIFY=&SYSUID,USER=&SYSUID


PF1 Help   PF3 Submit Job   PF4 Submit job   PF12 Cancel   ENTER Generate JCL
               and exit         and return                      for member
```

Figure 74. CCCA Conversion Submission Panel

If you press Enter, you return to the previous panel. You can then specify (for example) another source library and generate a new additional JCL for it. When

you press PF3 or PF4 on the CCCA conversion submission panel, all the jobs that you generated are submitted.

Check the SYSPRINT of the jobs. A condition code greater than zero most likely results from the compile step. See Chapter 7, “Using MLE Enhancements to Implement Sample Year 2000 Solutions” on page 177 for more information about how to resolve MLE related errors and warnings.

If, in the date identification file, you don’t have an entry for the program you want to convert, the first step of the conversion results in a return code of 16 and the following steps are not performed.

6.4 Date Identification File Format

The date identification file consists of 80-byte records containing data in a free-format style. Each record may contain one or more fields. Each field within a record is separated by one or more spaces. The details for each data item are grouped by program name, allowing the same date identification file to be used for more than one program.

The format of the date identification file is the program name followed by one or more data item details:

- **Program name** — The name of the program to which the data item information that follows applies. The program name can be a maximum of 30 characters and must be enclosed in ″<″ and ″>″ symbols. That’s how CCCA recognizes that the following details belong to a new program.
- **Data item details** — The data item detail consist of three elements:
  - **Number** — A numeric field that CCCA uses as a delimiter to recognize that a new data item follows. CCCA does not use the actual value of the field. However, it can be useful to use this field as a reference to the line number in the source COBOL program where the data item is defined. MA2000, for example, uses this line number when it generates the date identification file.
  - **Date Format** — The date format of the item. Any of the date formats supported by MLE is valid. Table 28 on page 159 illustrates which patterns are supported.
  - **Item name** — The qualified name of the data item. The syntax for qualifying names within normal COBOL source code allows either the word ″IN″ or the word ″OF″ to be used between the lower-level data name and the higher-level data name. However, only ″OF″ is acceptable in the case of qualified names in the date identification file.

While the date identification file is free-format, you will find it far more readable, and therefore much easier to reference, if a formatted style is used.

As an example Figure 75 on page 175 shows how MA2000 formats the date identification file.

```
<TARMU3>
 00044 YYXXX    EMP-DATE-JOINED OF EMPLOYEE-MASTER-RECORD
 00046 YYXXXX   EMP-DATE-TERMINATED OF EMPLOYEE-MASTER-RECORD
 00048 YYXXX    EMP-DATE-MAINTAINED OF EMPLOYEE-MASTER-RECORD
 00050 YYXXX    EMP-BIRTH-DATE OF EMPLOYEE-MASTER-RECORD
 00052 YYXXXX   EMP-SECURITY-EXP OF EMPLOYEE-MASTER-RECORD
 00103 YYYYXXX  WORK-EIB-DATE-CHAR OF WORK-VARS
 00106 YYXXX    WORK-EIB-YYDDD OF WORK-EIB-DATE-CHAR OF WORK-VARS
 00107 YYXXX    WORK-JOINED-YYDDD OF WORK-VARS
 00110 YY       WORK-JOINED-YY OF WORK-JOINED-MMDDYY OF WORK-VARS
 00112 YYXXX    WORK-TERMINATED-YYDDD OF WORK-VARS
 00122 YYXXX    TARDATE-DATE-YYDDD OF TARDATE-DATE-RED1 OF TARDATE-PARAMETERS
 00125 YYXXXX   TARDATE-DATE-YYMMDD OF TARDATE-DATE-RED2 OF TARDATE-PARAMETERS
<TARDTE3>
 00003 YYXXXX   WRK-DATE OF WORK-DATES
 00004 YY       WRK-DATE-YY OF WRK-DATE OF WORK-DATES
 00008 YYXXXX   WRK-DATE-RED OF WORK-DATES
 00009 YY       WRK-DTE-YY OF WRK-DATE-RED OF WORK-DATES
 00015 YYXXX    WRK-DATE-YYDDD OF WORK-DATES
 00016 YY       WRK-DATE-YY OF WRK-DATE-YYDDD OF WORK-DATES
 00023 YYYY     WRK-YEAR-YYYY OF WORK-DATES
 00025 YY       WRK-YEAR-YY OF WRK-YEAR-YYYY OF WORK-DATES
 00026 YYYY     WRK-YEAR-YYYY-NUM OF WORK-DATES
 00034 YY       WRK-DATE-YY OF WRK-DATE-MMXDDXYY OF WORK-DATES
 00037 YY       WRK-DATE-YY OF WRK-DATE-YYXDDXMM OF WORK-DATES
```

Figure 75. Sample Date Identification File

Chapter 7. Using MLE Enhancements to Implement Sample Year 2000 Solutions

This chapter uses our sample application programs as the basis for documenting possible approaches which can be adopted to solve the problems associated with the use of 2-digit date fields. In each case, we illustrate how the Millennium Language Extensions can be used within the context of each approach taken to aid you in the resolution of your date-related processing problems. The approaches we have chosen to describe are:

- Basic remediation
- Internal bridging
- File and database conversion
- Full field expansion

7.1 Sample JCL for COBOL Program Compilation and Linkage Editor Processing

Figure 76 provides a sample of the JCL we used to compile our sample application COBOL programs.

```
//IGYWIVPM JOB ,                                                        00010000
//             MSGLEVEL=(1,1),MSGCLASS=H,TIME=(,59),                    00020000
//             NOTIFY=&SYSUID,USER=&SYSUID                              00030000
//*                                                                     00040000
//IGYWCLG PROC SYSLBLK=3200,                                            00050000
//             LIBPRFX='SYS1',                                          00060000
//             HLQ='USERID',                                            00070000
//             GOPGM=GO                                                 00090000
//*                                                                     00100000
//*********************************************************************  00110000
//*                                                                   *  00120000
//* IBM COBOL FOR OS/390 & VM                                         *  00130000
//*              VERSION 2 RELEASE 1 MODIFICATION 0                   *  00140000
//*                                                                   *  00150000
//* LICENSED MATERIALS - PROPERTY OF IBM                              *  00160000
//*                                                                   *  00170000
//* 5648-A25 (C) COPYRIGHT IBM CORP. 1991, 1997                       *  00180000
//* ALL RIGHTS RESERVED                                               *  00190000
//*                                                                   *  00200000
//* US GOVERNMENT USERS RESTRICTED RIGHTS - USE, DUPLICATION OR       *  00210000
//* DISCLOSURE RESTRICTED BY GSA ADP SCHEDULE CONTRACT WITH IBM       *  00220000
//* CORP.                                                             *  00230000
//*                                                                   *  00240000
//*********************************************************************  00250000
//*                                                                       00260000
//*  COMPILE, AND LINK EDIT A COBOL PROGRAM                               00270000
//*                                                                       00280000
//*  PARAMETER  DEFAULT VALUE    USAGE                                    00290000
//*   SYSLBLK   3200             BLKSIZE FOR OBJECT DATA SET              00300000
//*   LIBPRFX   CEE              PREFIX FOR LIBRARY DATA SET NAMES        00310000
//*   HLQ       USERID           DATASET HIGH LEVEL QUALIFIER             00320000
//*   GOPGM     TARMU3           MEMBER NAME FOR LOAD MODULE              00340001
//*                                                                       00350000
//*  CALLER MUST SUPPLY //COBOL.SYSIN DD ...                              00351000
//*                                                                       00352000
//*********************************************************************  00360000
```

Figure 76 (Part 1 of 2). Sample JCL for COBOL Program Compilation and Linkage Editor

```
//TRN    EXEC PGM=DFHECP1$,                                          00370000
//            PARM=('COBOL2'),                                      00370100
//            REGION=2M                                             00370200
//STEPLIB  DD  DSN=CICSVS.CICS410.SDFHLOAD,DISP=SHR                  00370300
//SYSPRINT DD  SYSOUT=*                                              00370400
//SYSIN    DD  DSN=&HLQ..SAMPAPPL.COBOL(&GOPGM),DISP=SHR   <--  COBOL Source Program                00370501
//SYSPUNCH DD  DSN=&&SYSCIN,                                         00370600
//             DISP=(,PASS),UNIT=SYSDA,                              00370700
//             DCB=BLKSIZE=400,                                      00370800
//             SPACE=(400,(400,100))                                 00370900
//*                                                                  00371000
//COBOL  EXEC PGM=IGYCRCTL,PARM='DATEPROC(FLAG),YEARWINDOW(1950)',   00371100
//            REGION=2048K                                          00372000
//STEPLIB  DD  DSNAME=IGYV2R10.SIGYCOMP,DISP=SHR                     00380000
//         DD  DSNAME=IGY.V2R1MA.SIGYCOMP,DISP=SHR                   00390000
//SYSLIB   DD  DSNAME=CICSVS.CICS410.SDFHCOB,DISP=SHR                00392000
//         DD  DSNAME=CICSVS.CICS410.SDFHMAC,DISP=SHR                00393000
//         DD  DSNAME=CICSVS.CICS410.SDFHSAMP,DISP=SHR               00394000
//         DD  DSNAME=&HLQ..SAMPAPPL.COPYBOOK,DISP=SHR  <--  COBOL Copybook              00391001
//SYSPRINT DD  SYSOUT=*                                              00400000
//SYSIN    DD  DSN=&&SYSCIN,DISP=(OLD,DELETE)                        00401000
//SYSLIN   DD  DSNAME=&&LOADSET,UNIT=SYSDA,                          00410000
//             DISP=(MOD,PASS),SPACE=(TRK,(3,3)),                    00420000
//             DCB=(BLKSIZE=&SYSLBLK)                                00430000
//SYSUT1   DD  UNIT=SYSDA,SPACE=(CYL,(1,1))                          00450000
//SYSUT2   DD  UNIT=SYSDA,SPACE=(CYL,(1,1))                          00460000
//SYSUT3   DD  UNIT=SYSDA,SPACE=(CYL,(1,1))                          00470000
//SYSUT4   DD  UNIT=SYSDA,SPACE=(CYL,(1,1))                          00480000
//SYSUT5   DD  UNIT=SYSDA,SPACE=(CYL,(1,1))                          00490000
//SYSUT6   DD  UNIT=SYSDA,SPACE=(CYL,(1,1))                          00500000
//SYSUT7   DD  UNIT=SYSDA,SPACE=(CYL,(1,1))                          00510000
//*                                                                  00511000
//LKED   EXEC PGM=HEWL,COND=(8,LT,COBOL),REGION=1024K                00520000
//SYSLIB   DD  DSNAME=&LIBPRFX..SCEELKED,DISP=SHR                    00530000
//         DD  DSNAME=CICSVS.CICS410.SDFHLOAD,DISP=SHR               00541000
//         DD  DSNAME=&HLQ..SAMPAPPL.LOAD,DISP=SHR     <--  Input Load Module         00542001
//SYSLMOD  DD  DSNAME=&HLQ..SAMPAPPL.LOAD(&GOPGM),DISP=SHR <--  Output Load Module       00551001
//SYSUT1   DD  UNIT=SYSDA,SPACE=(TRK,(10,10))                        00552000
//SYSPRINT DD  SYSOUT=*                                              00553000
//OBJECT   DD  DSN=&&LOADSET,DISP=(OLD,DELETE)                       00554000
//CICSOB   DD  DSN=CICSVS.CICS410.SDFHCOB,DISP=SHR                   00555000
//SYSIN    DD  DUMMY                                                 00556000
//SYSLIN   DD  DUMMY                                                 00557001
//   PEND                                                            00600000
//RUNIVP EXEC IGYWCLG,PARM.COBOL=RENT,REGION=1400K,                  00610000
//            HLQ='ITSORS2',GOPGM=TARMU3                  <--  Overrides       00610101
//LKED.SYSLIN  DD  *                                                 00610300
   INCLUDE CICSOB(DFHEILIC)                                        00610400
   INCLUDE OBJECT                                                  00610500
   NAME    TARMU3(R)                                       <--  Module Name       00610600
//*                                                                00621000
//*               ========> END OF JOB IGYWIVP1 <========            00630000
```

Figure 76 (Part 2 of 2). Sample JCL for COBOL Program Compilation and Linkage Editor

7.2 Basic Remediation

This section examines the approach to the Year 2000 problem known as basic remediation, discusses under what circumstances it can be applied and the advantages as well as the disadvantages of adopting such an approach. COBOL programs from our sample application also serve as the basis for a detailed treatment of what steps are necessary when using MLE in the implementation of this solution.

7.2.1 Definition

Basic remediation is an approach intended to ensure that programs continue to function through the year 2000 by the introduction of century windowing logic. Because each 2-digit number is unique within any 100-year interval, programs can unambiguously infer the correct century depending upon where in the century window the 2-digit year lies.

The automated windowing capability provided by the Millennium Language Extensions requires the specification of a century window and the fields within the program that contain windowed dates. The compiler then automatically supplies the logic to interpret the 2-digit years in the date fields in accordance with the century window.

7.2.2 Applicability

This approach is suitable when the files and database accessed by a program have not been converted from a 2- to a 4-digit format. The limited number of changes that must be made to program logic when using the MLE enhancements make this a preferred solution when a large number of programs must be modified and time is of the essence. Use of the automated windowed capability of MLE is also ideal for programs with a limited life-expectancy but which must nevertheless function through the year 2000. The definitions required for the windowing feature can be used as a basis for later migration to 4-digit expanded year logic in your programs.

7.2.3 Benefits of This Approach

The use of the automated windowing capability of MLE offers the following advantages:

- No file or database changes are required.
- Implementation is fast because changes are largely limited to data description entries.
- The compiler diagnostic messages help to discover date variables that have been missed during the find phase.
- Minimal changes to program logic require less testing.
- Programs will function beyond the year 2000 until a permanent solution can be applied.

7.2.4 Limitations of This Approach

- Because a 2-digit year can only be unique in any given 100-year period, the windowing solution cannot be used over an extended period. As soon as your data requirements demand that your data window extend beyond the boundaries of a century, windowing is no longer an acceptable solution.
- The consistent use of century windows across program and application boundaries is difficult to manage.
- Windowing cannot be applied to all 2-digit dates in all programs.
- Windowed or expanded dates can have only zero, 2, 3 or 4 nonyear characters.

7.2.5 Sample Application Program Illustrating This Approach

TARMU6E is a COBOL program which reads a VSAM file containing employee records and checks to see if an active employee's security clearance expiration date has been reached. If the value of this date field is less than the current date, then the date is incremented by one year. A report file is also created by the program, listing which employee records have been updated. In addition to other information, this report contains the date the employee joined the firm as well as their newly extended security clearance expiration date, both of which are in the format of MM/DD/YY.

7.2.5.1 Initial MLE Revisions

Figure 77 illustrates the initial modifications made to the original version of TARMU6 in order to exploit the automatic windowing capability of the Millennium Language Extensions. The data description entries previously contained in copybook TARMU6C have been integrated into the program to simplify the example.

```
       CBL DATEPROC(FLAG) YEARWINDOW(1935) NODYNAM QUOTE SOURCE LIST   1
       IDENTIFICATION DIVISION.
       PROGRAM-ID.   TARMU6E
       AUTHOR.       MERRILL BANI/ERIC PRIVETTE
       DATE-WRITTEN. JUNE 1997/APRIL 1998
      * -------------------------------------------------------- */
      * LICENSED MATERIALS - PROPERTY OF IBM                     */
      *                                                          */
      * (C) COPYRIGHT IBM CORP. 1997                            */
      *                                                          */
      * US GOVERNMENT USERS RESTRICTED RIGHTS - USE,            */
      * DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP         */
      * SCHEDULE CONTRACT WITH IBM CORP.                        */
      * -------------------------------------------------------- */
      *REMARKS.
      *
      *    TARDIS MVS COBOL TEST APPLICATION
      *
      *    EMPLOYEE SECURITY RENEWAL
      *
      *
       EJECT
       ENVIRONMENT DIVISION.

       INPUT-OUTPUT SECTION.

       FILE-CONTROL.

           SELECT EMPLOYEE-MASTER-FILE
                  ASSIGN TO EMPMAST
                  ORGANIZATION IS INDEXED
                  ACCESS IS SEQUENTIAL
                  RECORD KEY IS EMPLOYEE-ID
                  FILE STATUS IS EMP-FILE-STATUS.


           SELECT PRINT-EDIT-REPORT
                  ASSIGN TO REPFILE
                  FILE STATUS IS PRT-FILE-STATUS.

       EJECT
```

Figure 77 (Part 1 of 6). Initial MLE Enhancements Made to TARMU6

```
       DATA DIVISION.

       FILE SECTION.

       FD  EMPLOYEE-MASTER-FILE
           RECORD CONTAINS 200 CHARACTERS.
      *
       01  EMPLOYEE-MASTER-RECORD.
      *        ** key field
           03  EMPLOYEE-ID                 PIC X(6).
           03  EMPLOYEE-DEPT-CODE          PIC X(4).
           03  EMPLOYEE-NAME               PIC X(30).
           03  EMPLOYEE-ADDR-1             PIC X(30).
           03  EMPLOYEE-ADDR-2             PIC X(30).
           03  EMPLOYEE-ADDR-3             PIC X(30).
           03  EMPLOYEE-ZIP-CODE           PIC X(5).
      *        ** format (yyddd)
           03  EMPLOYEE-DATE-JOINED        PIC 9(5)
                                           DATE FORMAT YYXXX.          2
      *        ** format (yymmdd)
           03  EMPLOYEE-DATE-TERMINATED    PIC 9(6)
                                           DATE FORMAT YYXXXX.         2
      *        ** format (yyddd)
           03  EMPLOYEE-DATE-MAINTAINED    PIC 9(5)
                                           DATE FORMAT YYXXX.          2
      *        ** format (yyddd)
           03  EMPLOYEE-BIRTH-DATE         PIC 9(5)
                                           DATE FORMAT YYXXX.          2
      *        ** format (yyddd)
           03  EMPLOYEE-SECURITY-EXP       PIC 9(5) COMP-3
                                           DATE FORMAT YYXXX.          2
           03  FILLER                      PIC X(41).
           SKIP3
       FD  PRINT-EDIT-REPORT
           RECORDING MODE IS F
           RECORD CONTAINS 132 CHARACTERS
           LABEL RECORDS ARE OMITTED.
       01  PRINT-RECORD                    PIC X(132).
           SKIP3
           EJECT
       WORKING-STORAGE SECTION.

       01  FILE-STATUS-FIELDS.
           03  EMP-FILE-STATUS             PIC XX.
           03  PRT-FILE-STATUS             PIC XX.

       01  SWITCHES.
           03  VSAM-ERROR-SWITCH           PIC X VALUE SPACE.
               88  VSAM-ERROR                    VALUE "Y".
           03  EOF-SWITCH                  PIC X VALUE SPACE.
               88  EOF                           VALUE "Y".
           03  ERROR-SWITCH                PIC X VALUE SPACE.
               88  ERRORS                        VALUE "Y".

       01  WORK-DATE-YYMMDD                PIC X(6)
                                           DATE FORMAT IS YYXXXX.      2

       01  WORK-DATE-YYDDD                 PIC 9(5)
                                           DATE FORMAT YYXXX.          2
       01  WORK-TIME-HHMMSS.
           03  WRK-TIME-HH                 PIC XX.
           03  WRK-TIME-MM                 PIC XX.
           03  WRK-TIME-SS                 PIC XX.
           03  WRK-TIME-MS                 PIC XX.
       01  WORK-TIME-DISPLAY.
           03  WRK-TIME-HH                 PIC XX.
           03  FILLER                      PIC X VALUE ":".
           03  WRK-TIME-MM                 PIC XX.
           03  FILLER                      PIC X VALUE ":".
           03  WRK-TIME-SS                 PIC XX.
```

Figure 77 (Part 2 of 6). Initial MLE Enhancements Made to TARMU6

```
*    ** report headings & detail line
 01  REPORT-HEAD1.
     03  FILLER                      PIC X(01)  VALUE "1".
     03  FILLER                      PIC X(48)  VALUE
         "REPORT ID  TARMU6E".
     03  FILLER                      PIC X(61)  VALUE            4
         "EMPLOYEE SECURITY RENEWAL REPORT".
     03  RPT-HD1-DATE.
         05  RPT-HD1-MM              PIC XX.
         05  FILLER                  PIC X      VALUE "/".
         05  RPT-HD1-DD              PIC XX.
         05  FILLER                  PIC X      VALUE "/".
         05  RPT-HD1-YY              PIC XXXX
                                     DATE FORMAT YYYY.          3
     03  FILLER                      PIC XX     VALUE SPACES.
     03  RPT-HD1-TIME                PIC X(8).
     03  FILLER                      PIC XX     VALUE SPACES.

 01  REPORT-HEAD2.
     03  FILLER                      PIC X(01)  VALUE "0".
     03  FILLER                      PIC X(44)  VALUE
         "   ID       NAME                              ".
     03  FILLER                      PIC X(88)  VALUE
         "DEPT  JOINED      SEC/EXPIRY".                        4

 01  REPORT-DETAIL.
     03  FILLER                      PIC X(01)  VALUE "0".
     03  FILLER                      PIC X(03)  VALUE SPACES.
     03  RPT-DET-ID                  PIC X(06).
     03  FILLER                      PIC X(03)  VALUE SPACES.
     03  RPT-DET-NAME-ADDR           PIC X(30).
     03  FILLER                      PIC X(02)  VALUE SPACES.
     03  RPT-DET-DEPT                PIC X(04).
     03  FILLER                      PIC X(02)  VALUE SPACES.
     03  RPT-DET-JOINED.
         05 RPT-DET-JOINED-MMDD PIC X(06).
         05 RPT-DET-JOINED-YY   PIC X(04).                       5
     03  FILLER                      PIC X(02)  VALUE SPACES.
     03  RPT-DET-SEC-EXP.
         05 RPT-DET-SEC-EXP-MMDD PIC X(06).
         05 RPT-DET-SEC-EXP-YY   PIC X(04)  DATE FORMAT YYYY.    6
    03  FILLER                  PIC X(60)  VALUE SPACES.        4

 01  REPORT-WORK-VARIABLES.
     03  REP-LINE-COUNT              PIC 9(2)   VALUE 99.


 01  VSAM-ERROR-VARS.
     03  VSAM-ERROR-ACTION           PIC X(8).
     03  VSAM-ERROR-FILE             PIC X(10).
     03  VSAM-ERROR-STATUS           PIC X(2).

 01  TARDATE-PARAMETERS.
     03  TARDATE-DATE                PIC X(8).                  7
     03  TARDATE-DATE-RED1 REDEFINES TARDATE-DATE.
         05  TARDATE-DATE-YYDDD      PIC 9(5)
                                     DATE FORMAT YYXXX.         2
         05  FILLER                  PIC X(3).
     03  TARDATE-DATE-RED2 REDEFINES TARDATE-DATE.
         05  TARDATE-DATE-YYMMDD     PIC 9(6)
                                     DATE FORMAT YYXXXX.        2
         05  FILLER                  PIC X(2).
     03  TARDATE-DATE-RED3 REDEFINES TARDATE-DATE.
         05  TARDATE-DATE-MMDD       PIC X(6).
         05  TARDATE-DATE-YY         PIC 9(2)
                                     DATE FORMAT YY.            2
     03  TARDATE-INPUT-FORMAT        PIC X(8).
     03  TARDATE-OUTPUT-FORMAT       PIC X(8).
     03  TARDATE-MESSAGE             PIC X(30).

 01  DISPLAY-DATE                    PIC 9(4) DATE FORMAT YYYY.

     EJECT
```

Figure 77 (Part 3 of 6). Initial MLE Enhancements Made to TARMU6

```
 PROCEDURE DIVISION.

     PERFORM 100-INITIALIZE THRU 100-EXIT.

     IF  VSAM-ERROR   THEN
         NEXT SENTENCE
     ELSE
         PERFORM 200-MAIN-PROCESS THRU 200-EXIT.

     STOP RUN.


* ============================================================ *
     SKIP3
 100-INITIALIZE.

*    **  retrieve todays date

     ACCEPT WORK-DATE-YYMMDD FROM DATE.
     ACCEPT WORK-DATE-YYDDD  FROM DAY.
     ACCEPT WORK-TIME-HHMMSS FROM TIME.

     MOVE WORK-DATE-YYMMDD(1:2) TO RPT-HD1-YY.
     MOVE WORK-DATE-YYMMDD(5:2) TO RPT-HD1-DD.
     MOVE WORK-DATE-YYMMDD(3:2) TO RPT-HD1-MM.
     MOVE CORRESPONDING WORK-TIME-HHMMSS TO WORK-TIME-DISPLAY.
     MOVE WORK-TIME-DISPLAY     TO RPT-HD1-TIME.

*    **  open files

     MOVE "OPEN FILE" TO VSAM-ERROR-ACTION.

     OPEN I-O EMPLOYEE-MASTER-FILE.
     IF  EMP-FILE-STATUS = "00" THEN
         NEXT SENTENCE
     ELSE
         MOVE "EMPMAST"       TO VSAM-ERROR-FILE
         MOVE EMP-FILE-STATUS TO VSAM-ERROR-STATUS
         PERFORM 999-VSAM-ERROR THRU 999-EXIT
     END-IF.


     OPEN OUTPUT PRINT-EDIT-REPORT.
     IF  PRT-FILE-STATUS = "00" THEN
         NEXT SENTENCE
     ELSE
         MOVE "REPFILE"      TO VSAM-ERROR-FILE
         MOVE PRT-FILE-STATUS TO VSAM-ERROR-STATUS
         PERFORM 999-VSAM-ERROR THRU 999-EXIT
     END-IF.

 100-EXIT.
     EXIT.
* ============================================================ *
     SKIP3
 200-MAIN-PROCESS.

     READ EMPLOYEE-MASTER-FILE
         AT END
         GO TO 200-EXIT
     END-READ.

     IF  EMPLOYEE-ID = "999999"  THEN
         GO TO 200-MAIN-PROCESS.
```

Figure 77 (Part 4 of 6). Initial MLE Enhancements Made to TARMU6

```
          IF  EMPLOYEE-DATE-TERMINATED = ZEROS THEN

              IF  EMPLOYEE-SECURITY-EXP <= WORK-DATE-YYDDD THEN

                  PERFORM 300-PRINT-REPORT THRU 300-EXIT

                  PERFORM 400-UPDATE-FILE  THRU 400-EXIT

              END-IF

          END-IF.

          GO TO 200-MAIN-PROCESS.


       200-EXIT.
           EXIT.
      * ============================================================ *

       300-PRINT-REPORT.

      *    **  print edit report

           IF  REP-LINE-COUNT > 55 THEN
               WRITE PRINT-RECORD FROM REPORT-HEAD1
                     AFTER ADVANCING PAGE
               WRITE PRINT-RECORD FROM REPORT-HEAD2
                     AFTER ADVANCING 1 LINE
               MOVE ZERO TO REP-LINE-COUNT
           END-IF.

           MOVE SPACES                   TO  REPORT-DETAIL.
           WRITE PRINT-RECORD FROM REPORT-DETAIL
                    AFTER ADVANCING 1 LINE.

           MOVE SPACES                   TO  REPORT-DETAIL.
           MOVE EMPLOYEE-ID              TO  RPT-DET-ID.
           MOVE EMPLOYEE-DEPT-CODE       TO  RPT-DET-DEPT.
           MOVE EMPLOYEE-NAME            TO  RPT-DET-NAME-ADDR.
      *
           MOVE EMPLOYEE-DATE-JOINED     TO  TARDATE-DATE-YYDDD.
           MOVE "YYDDD"                  TO  TARDATE-INPUT-FORMAT.
           MOVE "MM/DD/YY"               TO  TARDATE-OUTPUT-FORMAT.
           CALL "TARDTE3"             USING  TARDATE-DATE            7
                                             TARDATE-INPUT-FORMAT
                                             TARDATE-OUTPUT-FORMAT
                                             TARDATE-MESSAGE.
           MOVE TARDATE-DATE-MMDD TO RPT-DET-JOINED-MMDD.
           MOVE TARDATE-DATE-YY   TO RPT-DET-JOINED-YY.
      *
           MOVE EMPLOYEE-SECURITY-EXP    TO  TARDATE-DATE-YYDDD.
           MOVE "YYDDD"                  TO  TARDATE-INPUT-FORMAT.
           MOVE "MM/DD/YY"               TO  TARDATE-OUTPUT-FORMAT.
           CALL "TARDTE3"             USING  TARDATE-DATE            7
                                             TARDATE-INPUT-FORMAT
                                             TARDATE-OUTPUT-FORMAT
                                             TARDATE-MESSAGE.
```

Figure 77 (Part 5 of 6). Initial MLE Enhancements Made to TARMU6

```
           ADD 1 TO  TARDATE-DATE-YY
              ON SIZE ERROR GO TO 500-SIZE-ERROR.                    8

           IF  TARDATE-DATE-MMDD = "02/29/"  THEN
               MOVE "02/28/" TO TARDATE-DATE-MMDD

           END-IF.
           MOVE TARDATE-DATE-MMDD TO RPT-DET-SEC-EXP-MMDD.
           MOVE TARDATE-DATE-YY   TO RPT-DET-SEC-EXP-YY.             6

           WRITE PRINT-RECORD FROM REPORT-DETAIL
                   AFTER ADVANCING 1 LINE.

           ADD 2 TO REP-LINE-COUNT.


       300-EXIT.
           EXIT.
      * ============================================================ *
       400-UPDATE-FILE.
           MOVE "MM/DD/YY"               TO  TARDATE-INPUT-FORMAT.
           MOVE "YYDDD"                  TO  TARDATE-OUTPUT-FORMAT.
           CALL "TARDTE3"              USING  TARDATE-DATE           7
                                              TARDATE-INPUT-FORMAT
                                              TARDATE-OUTPUT-FORMAT
                                              TARDATE-MESSAGE.

           MOVE TARDATE-DATE-YYDDD  TO  EMPLOYEE-SECURITY-EXP.
           REWRITE EMPLOYEE-MASTER-RECORD.
           IF  EMP-FILE-STATUS  NOT = "00" THEN
               MOVE "REWRITE FILE"  TO VSAM-ERROR-ACTION
               MOVE "EMPMAST"       TO VSAM-ERROR-FILE
               MOVE EMP-FILE-STATUS TO VSAM-ERROR-STATUS
               PERFORM 999-VSAM-ERROR THRU 999-EXIT
           END-IF.

       400-EXIT.
           EXIT.

       500-SIZE-ERROR.                                                  8
           MOVE TARDATE-DATE-YY TO DISPLAY-DATE.
           DISPLAY "TARDATE-DATE-YY OUTSIDE CENTURY WINDOW IF DATE "
                   DISPLAY-DATE " INCREMENTED".
           GO TO 200-EXIT.

       500-EXIT.
           EXIT.
      * ============================================================ *
       999-VSAM-ERROR.

           SET VSAM-ERROR  TO  TRUE.

           DISPLAY VSAM-ERROR-ACTION
                   VSAM-ERROR-FILE
                   "STATUS ERROR "
                   VSAM-ERROR-STATUS.

       999-EXIT.
           EXIT.
```

Figure 77 (Part 6 of 6). Initial MLE Enhancements Made to TARMU6

1 The DATEPROC compiler option enables the language extensions of the COBOL compiler. With DATEPROC(FLAG), the compiler will produce a diagnostic message **wherever** a language element uses or is affected by the extensions. The YEARWINDOW compiler option is used to specify the starting year of a fixed 100-year interval ranging from 1935-2034. Given this century window, the year part (yy) of a numeric windowed date field is treated as if it were expanded as follows:

- If yy < 35, then add 2000 to yy.
- If yy >= 35, then add 1900 to yy.

Alphanumeric windowed date fields are treated similarly, but a prefix of ″19″ or ″20″ is added in accordance with the century window. Only the value yy is actually stored in the windowed date field, however.

2 The addition of the DATE FORMAT clause means that the compiler recognizes the date fields as windowed date fields and automatically applies the century window when they are referenced. A value of 08 in any of these variables will be recognized by subsequent statements in the program as 2008 in accordance with the century window.

3 The addition of the DATE FORMAT clause means that the compiler will recognize these items as expanded, or 4-digit, dates. When the contents of a 2-digit date are assigned to these date fields by a MOVE or COMPUTE statement, the compiler will automatically supply the century portion represented by the 2 leftmost positions by applying the century window.

4 Other data description entries can also be affected by the introduction of MLE language elements into the data division. In this case, the expansion of the date field RPT-HD1-YY means that the length of other report variables must be adjusted to produce the correct alignment in the output report.

5 Like the variables identified in **3** above, RPT-DET-JOINED-YY should have been identified as an expanded date by means of a DATE FORMAT clause. This was intentionally omitted to illustrate the type of diagnostic message produced by the compiler in such an instance.

6 A DATE FORMAT clause is used to identify expanded and windowed date fields to the compiler. This clause cannot be used, however, unless the year portion of the date is the first or only part of the date field. For this reason, the elementary items RPT-DET-JOINED and RPT-DET-SEC-EXP must be converted into group items in order to allow the year portion of the date to be isolated so that date windowing can be used.

7 If data-name TARDATE-DATE found in this USING phrase contained a DATE FORMAT clause, the compiler would generate a diagnostic message and the date field would be treated as if it were a nondate. This would mean that the DATE FORMAT clause would be ignored and the contents would not be windowed.

8 An ON SIZE ERROR phrase should always be used when the contents of windowed date fields are involved in arithmetic operations. In this case, the windowed year TARDATE-DATE-YY is incremented by one. If this windowed date contained a value of 09 and the century window were defined as 1910-2009, the arithmetic operation is performed, the value 10 would be stored in TARDATE-DATE-YY. Subsequent statements in the program, however, would recognize this value as 1900 in accordance with the century window, which is not the intended result. In such cases where the result of an arithmetic operation falls outside the century window, the result of the operation depends on whether or not the ON SIZE ERROR phrase is specified:

- If ON SIZE ERROR is specified, the value of the receiving field is not changed and the SIZE error imperative statement is executed. In this case, an error message is written to SYSOUT and program execution is halted.
- If ON SIZE ERROR is not specified, the result is stored in the receiving field with truncation performed on the leftmost 2 digits.

7.2.5.2 Eliminating Compiler Diagnostic Messages

Figure 78 illustrates the compiler diagnostic messages produced when the source code of TARMU6 was compiled using the DATEPROC(FLAG) option. The use of this option results in the production of a message for every statement which defines or references a date field. As is the case with other compiler diagnostic messages, messages related to date-sensitive statements can be classified according to their severity as follows:

- **Information-level messages** are intended to highlight each definition or use of a date field. Figure 78 contains the information-level messages produced by the compiler after the initial introduction of the MLE language elements:

```
LineID  Message code  Message text
    48  IGYDS1454-I   A group item contained one or more subordinate date fields.

                      Same message on line:    112    136    161

   163  IGYDS1462-I   Group item "TARDATE-DATE-RED1", which contained one or more date fields, redefined "TARDATE-DATE".

   167  IGYDS1462-I   Group item "TARDATE-DATE-RED2", which contained one or more date fields, redefined "TARDATE-DATE".

   171  IGYDS1462-I   Group item "TARDATE-DATE-RED3", which contained one or more date fields, redefined "TARDATE-DATE".

   200  IGYPA3290-I   The statement contained date logic.

                      Same message on line:    201    204    205    206    248    250    288    297    305    314    333    346
                                               347

   200  IGYPA3316-I   The "ACCEPT" statement will transfer a windowed date value into "WORK-DATE-YYMMDD (ALPHANUMERIC)".

   201  IGYPA3316-I   The "ACCEPT" statement will transfer a windowed date value into "WORK-DATE-YYDDD (NUMERIC INTEGER)".

        1
   248  IGYPA3310-I   A windowed date comparison will occur between "EMPLOYEE-DATE-TERMINATED (NUMERIC INTEGER)" and "ZEROS".

        1
   250  IGYPA3310-I   A windowed date comparison will occur between "EMPLOYEE-SECURITY-EXP (PACKED INTEGER)" and "WORK-DATE-YYDDD
                      (NUMERIC INTEGER)".

   305  IGYPA3299-I   The "SIZE ERROR" imperative statements will be executed if the year part of the date result is not valid for
                      windowed date receiving field "TARDATE-DATE-YY (NUMERIC INTEGER)".

   314  IGYPA3309-I   The windowed date value in "TARDATE-DATE-YY (NUMERIC INTEGER)" will be expanded and moved into
                      "RPT-DET-SEC-EXP-YY (ALPHANUMERIC)".

   346  IGYPA3309-I   The windowed date value in "TARDATE-DATE-YY (NUMERIC INTEGER)" will be expanded and moved into "DISPLAY-DATE
                      (NUMERIC INTEGER)".
```

Figure 78. Compiler Diagnostic Messages (I) for Remediation Program TARMU6

1 This message indicates that data of two different types is involved in a windowed comparison:

  - The numeric integer EMPLOYEE-DATE-TERMINATED and the figurative constant ZEROS
  - The packed integer EMPLOYEE-SECURITY-EXP and the numeric integer WORK-DATE-YYDDD.

- **Warning-level messages** indicate that the compiler has been forced to make an assumption about a date or a nondate field because of the absence of sufficient information in the program or because date logic exists that must be manually investigated to determine if it is free of error. Compilation continues with the noted assumptions in effect. Figure 79 on page 188 lists the warning-level messages produced during the compilation of TARMU6.

```
LineID  Message code  Message text
          1
   248  IGYPA3304-W   Non-date "ZEROS" was treated as if it had the same date format as date field "EMPLOYEE-DATE-TERMINATED
                      (NUMERIC INTEGER)", but with a base year of 1900.
```

Figure 79. Compiler Diagnostic Messages (W) For Remediation Program TARMU6

1 This message is issued whenever a nondate is involved in a comparison with a date field and should **never** be disregarded. When such a comparison takes place, the nondate is assumed to have the same number of year and nonyear digits as the date field with which it is being compared. Not only is an assumption made regarding the date format of the nondate, an additional assumption is made regarding the century window, which should be applied when resolving it. The assumed century window is defined by the 100-year interval ranging from 1900-1999, which is normally not the same as the century window in effect for the compile unit and specified by the YEARWINDOW compiler option. One of the objectives of the Millennium Language Extensions is not to change the existing semantics of programs by the addition of date fields. Existing programs assume that 2-digit year dates which are expressed as nondates, such as literals, are in the range 1900-1999 and the extensions maintain that assumption.

In our example, the value 000000 is regarded as having the same date format as the windowed date EMPLOYEE-DATE-TERMINATED. As a result, its 2-digit year is interpreted on the basis of a century window. Because ZEROS is not defined as a date field by means of a DATE FORMAT clause, however, the assumed window is applied to the 2-digit year portion of the date instead of the YEARWINDOW in effect for the compile unit. Consequently, the year portion of ZEROS in the program is interpreted as 1900 whereas the 00 contained in the year portion of the date field in the employee record is regarded as 2000 as specified in the YEARWINDOW compiler option for the compile unit. Unless the century window of the compile unit is changed to a value equivalent to the assumed window, this comparison will never equate as true.

The following examples illustrate how the assumptions which apply when nondates are involved in comparison operations with date fields affect comparison operations.

– Figure 80 illustrates a comparison between a 6-digit nondate literal and a 2-digit numeric windowed date field.

```
CBL DATEPROC(FLAG) YEARWINDOW(1950)
⋮
01 WINDOWED-DATE-FIELD  PIC 9(6) DATE FORMAT YYXXXX VALUE 980505.
77 NON-DATE-LITERAL     PIC 9(6) VALUE 500505.
⋮
IF WINDOWED-DATE-FIELD > NON-DATE-LITERAL
  ...
```

Figure 80. A 6-Character Nondate in a Comparison Operation

WINDOWED-DATE-FIELD is greater than NON-DATE-LITERAL because the assumed century window of 1900-1999 is applied to NON-DATE-LITERAL, resulting in a value of 19500505, which is less than the value of the windowed date 19980505.

– Figure 81 on page 189 illustrates a comparison between an 8-digit nondate literal and a 2-digit numeric windowed date field

```
CBL DATEPROC(FLAG) YEARWINDOW(1950)
⋮
01 WINDOWED-DATE-FIELD  PIC 9(6) DATE FORMAT YYXXXX VALUE 980505.
77 NON-DATE-LITERAL     PIC 9(8) VALUE 00980505.
⋮
IF WINDOWED-DATE-FIELD > NON-DATE-LITERAL
  ...
```

Figure 81. An 8-Character Nondate (Century not Specified) in a Comparison Operation

WINDOWED-DATE-FIELD is equal to NON-DATE-LITERAL. The nondate literal is assumed to have the same number of nonyear and year digits as WINDOWED-DATE-FIELD. The last 4 digits of the nondate literal are therefore regarded as representing the nonyear portion, leaving 0098 to represent the year. 1900 is then added to the remaining year portion, resulting in a value of 1998. The NON-DATE-LITERAL consequently has a value of 19980505, which is equivalent to that of WINDOWED-DATE-FIELD.

Figure 82 provides a similar example.

```
CBL DATEPROC(FLAG) YEARWINDOW(1950)
⋮
01 WINDOWED-DATE-FIELD  PIC 9(6) DATE FORMAT YYXXXX VALUE 980505.
77 NON-DATE-LITERAL     PIC 9(8) VALUE 19980505.
⋮
IF WINDOWED-DATE-FIELD > NON-DATE-LITERAL
  ...
```

Figure 82. An 8-Character Nondate (Including Century) in a Comparison Operation

WINDOWED-DATE-FIELD is less than NON-DATE-LITERAL. The nondate literal is assumed to have the same number of nonyear and year digits as WINDOWED-DATE-FIELD. The last 4 digits of the nondate literal are therefore regarded as representing the nonyear portion, leaving 1998 to represent the year. 1900 is then added to the remaining year portion, resulting in a value of 3898. NON-DATE-LITERAL consequently has a value of 38980505, which is greater than that of WINDOWED-DATE-FIELD.

– Figure 83 illustrates a comparison between an 8-digit nondate literal and a 2-digit alphanumeric date field.

```
CBL DATEPROC(FLAG) YEARWINDOW(1950)
⋮
01 WINDOWED-DATE-FIELD  PIC X(6) DATE FORMAT YYXXXX VALUE 980505.
77 NON-DATE-LITERAL     PIC X(8) VALUE "19980505".
⋮
IF WINDOWED-DATE-FIELD > NON-DATE-LITERAL
  ...
```

Figure 83. A Comparison Operation Involving an Alphanumeric Date Field

WINDOWED-DATE-FIELD is greater than NON-DATE-LITERAL. The nondate literal is assumed to have the same number of nonyear and year digits as WINDOWED-DATE-FIELD. The last 4 digits of the nondate literal are therefore regarded as representing the nonyear portion, and the first 2 digits "19" as representing the year. The

assumed century window value of ″19″ is then prefixed to the year portion, resulting in an interpretation of the year as ″1919″. The literal value ″1919980505″ is less than ″19980505″ which is contained in WINDOWED-DATE-FIELD.

Figure 85 on page 191 **1** shows one possible solution that will eliminate this compiler warning-level diagnostic message. The UNDATE function is used to tell the compiler to treat the date field EMPLOYEE-DATE-TERMINATED as a nondate in this particular instance. The UNDATE function can be used if you are moving a date field to a nondate or if, as in this case, you are comparing a nondate and a windowed date field and you don′t want a windowed comparison. Because the comparison involving a date field is eliminated by this conversion of a date field to a non-date, no windowing is applied.

- **Error-level messages** indicate that the date field is used incorrectly. Although compilation is not halted, the diagnostic message highlights the fact that the run-time results of such a use of the date field are unreliable. The statement generating the error is eliminated from the compilation. Figure 84 lists the error-level messages produced during the compilation of TARMU6.

```
LineID  Message code  Message text
          2
   204  IGYPA3314-E   Windowed date field ″WORK-DATE-YYMMDD (ALPHANUMERIC REFERENCE MODIFIED ITEM)″ was reference modified.
                      It was treated as a non-date in this context.

                      Same message on line:    205    206

          3
   299  IGYPA3307-E   Date field ″TARDATE-DATE-YY (NUMERIC INTEGER)″ was found as a sender, but the receiver was non-date
                      ″RPT-DET-JOINED-YY (ALPHANUMERIC)″.  ″TARDATE-DATE-YY (NUMERIC INTEGER)″ was treated as a non-date
                      in this context.
```

Figure 84. Compiler Diagnostic Messages (E) For Basic Remediation Program TARMU6

2 Reference modification of date fields will produce an error-level message. A message of this severity should be eliminated by the correction of the use of the date field in the impacted statement. Figure 85 on page 191 **2** shows how this message can be eliminated by defining additional elementary date items representing the year, month, and day portions of the group item. The 2-digit year portion of the date field is still eligible for automatic windowing because it is the first and only part of the newly created date field.

3 This message is the result of our intentional omission of the DATE FORMAT clause in the data description entry for RPT-DET-JOINED-YY in order to simulate what might happen if the definition of a date item is omitted. As this error was caused by our failure to provide a DATE FORMAT clause to identify this item as an expanded date field, the addition of the clause remedies the problem as illustrated in Figure 85 on page 191 **3** . It is important to note, however, that moving a date field to a nondate is not permitted. If this had actually been our intention, conversion of either the sending or the target field would be required. Use the DATEVAL function to covert a nondate to a date field or the UNDATE function to convert a date to a nondate field in such cases.

- **Severe-level messages** also indicate that the use of a date field is incorrect, but in this case the statement that generated the message is discarded from

the compilation entirely. The compilation of TARMU6 did not result in the generation of any severe-level messages.

Figure 85 shows the modifications made to the initial version of TARMU6 to avoid the generation of warning- and error-level messages during compilation. The original source code is provided as commentary for purposes of comparison.

```
      *01  WORK-DATE-YYMMDD               PIC X(6)                          2
      *                                   DATE FORMAT IS YYXXXX.

       01  WORK-DATE-YYMMDD         DATE FORMAT IS YYXXXX.
           03  WORK-DATE-YY  PIC XX DATE FORMAT IS YY.
           03  WORK-DATE-MM  PIC XX.
           03  WORK-DATE-DD  PIC XX.
⋮
      *    05 RPT-DET-JOINED-YY   PIC X(04).                               3
           05 RPT-DET-JOINED-YY   PIC X(04) DATE FORMAT YYYY.
⋮
      *    MOVE WORK-DATE-YYMMDD(1:2) TO RPT-HD1-YY.                       2
      *    MOVE WORK-DATE-YYMMDD(5:2) TO RPT-HD1-DD.
      *    MOVE WORK-DATE-YYMMDD(3:2) TO RPT-HD1-MM.
           MOVE WORK-DATE-YY TO RPT-HD1-YY.
           MOVE WORK-DATE-DD TO RPT-HD1-DD.
           MOVE WORK-DATE-MM TO RPT-HD1-MM.
⋮
      *    IF  EMPLOYEE-DATE-TERMINATED = ZEROS THEN                       1
           IF  FUNCTION UNDATE(EMPLOYEE-DATE-TERMINATED) = ZEROS THEN
```

Figure 85. Corrections Made to Eliminate Compiler Diagnostic Messages (E/W)

Figure 86 on page 192 illustrates the compiler diagnostic messages that are generated after the elimination of the warning- and error-level messages. For as long as the DATEPROC(FLAG) compiler option is in effect, information-level messages highlighting the definition and use of every date field will be produced. When compilation results in the generation of strictly informative messages such as those shown in Figure 86 on page 192, you can then recompile with the DATEPROC(NOFLAG) option in effect. This will suppress the generation of information-level diagnostics regarding date-sensitive statements.

```
LineID  Message code  Message text

    48  IGYDS1454-I   A group item contained one or more subordinate date fields.

                      Same message on line:    114    138    165

   167  IGYDS1462-I   Group item "TARDATE-DATE-RED1", which contained one or more date fields, redefined "TARDATE-DATE".

   171  IGYDS1462-I   Group item "TARDATE-DATE-RED2", which contained one or more date fields, redefined "TARDATE-DATE".

   175  IGYDS1462-I   Group item "TARDATE-DATE-RED3", which contained one or more date fields, redefined "TARDATE-DATE".

   204  IGYPA3290-I   The statement contained date logic.

                      Same message on line:    205    208    252    257    295    303    305    313    321    340    353    354

   204  IGYPA3316-I   The "ACCEPT" statement will transfer a windowed date value into "WORK-DATE-YYMMDD (GROUP)".

   205  IGYPA3316-I   The "ACCEPT" statement will transfer a windowed date value into "WORK-DATE-YYDDD (NUMERIC INTEGER)".

   208  IGYPA3309-I   The windowed date value in "WORK-DATE-YY (ALPHANUMERIC)" will be expanded and moved into "RPT-HD1-YY
                      (ALPHANUMERIC)".

   257  IGYPA3310-I   A windowed date comparison will occur between "EMPLOYEE-SECURITY-EXP (PACKED INTEGER)" and "WORK-DATE-YYDDD
                      (NUMERIC INTEGER)".

   303  IGYPA3309-I   The windowed date value in "TARDATE-DATE-YY (NUMERIC INTEGER)" will be expanded and moved into
                      "RPT-DET-JOINED-YY (ALPHANUMERIC)".

   313  IGYPA3299-I   The "SIZE ERROR" imperative statements will be executed if the year part of the date result is not valid for
                      windowed date receiving field "TARDATE-DATE-YY (NUMERIC INTEGER)".

   321  IGYPA3309-I   The windowed date value in "TARDATE-DATE-YY (NUMERIC INTEGER)" will be expanded and moved into
                      "RPT-DET-SEC-EXP-YY (ALPHANUMERIC)".

   353  IGYPA3309-I   The windowed date value in "TARDATE-DATE-YY (NUMERIC INTEGER)" will be expanded and moved into "DISPLAY-DATE
                      (NUMERIC INTEGER)".
```

Figure 86. Compiler Diagnostic Messages after Correction

7.2.5.3 Year 2000 Compliant Application Program with MLE Enhancements

Figure 87 on page 193 provides an example of the final version of program TARMU6, which has been renamed to TARMU6E to reflect the use of the Millennium Language Extensions. A total of 26 modifications were made to the original program in order to take advantage of the automatic windowing capability of the COBOL compiler:

- 1 change was made to the PROCESS (CBL) statement.
- 17 changes were made to the Data Division, including the addition of 13 DATE FORMAT clauses to existing variables and the introduction of one new variable with a DATE FORMAT clause.
- 8 changes were made to the Procedure Division.

```
       CBL DATEPROC(NOFLAG) YEARWINDOW(1935) NODYNAM QUOTE SOURCE LIST
       IDENTIFICATION DIVISION.
       PROGRAM-ID.   TARMU6E
       AUTHOR.       MERRILL BANI/ERIC PRIVETTE
       DATE-WRITTEN. JUNE 1997/APRIL 1998
      * -------------------------------------------------------- */
      * LICENSED MATERIALS - PROPERTY OF IBM                     */
      *                                                          */
      * (C) COPYRIGHT IBM CORP. 1997                             */
      *                                                          */
      * US GOVERNMENT USERS RESTRICTED RIGHTS - USE,             */
      * DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP          */
      * SCHEDULE CONTRACT WITH IBM CORP.                         */
      * -------------------------------------------------------- */
      *REMARKS.
      *
      *    TARDIS MVS COBOL TEST APPLICATION
      *
      *    EMPLOYEE SECURITY RENEWAL
      *
      *
       EJECT
       ENVIRONMENT DIVISION.

       INPUT-OUTPUT SECTION.

       FILE-CONTROL.

           SELECT EMPLOYEE-MASTER-FILE
                  ASSIGN TO EMPMAST
                  ORGANIZATION IS INDEXED
                  ACCESS IS SEQUENTIAL
                  RECORD KEY IS EMPLOYEE-ID
                  FILE STATUS IS EMP-FILE-STATUS.


           SELECT PRINT-EDIT-REPORT
                  ASSIGN TO REPFILE
                  FILE STATUS IS PRT-FILE-STATUS.

       EJECT
       DATA DIVISION.

       FILE SECTION.

       FD  EMPLOYEE-MASTER-FILE
           RECORD CONTAINS 200 CHARACTERS.
      *
```

Figure 87 (Part 1 of 6). Final MLE Version of TARMU6/TARMU6E

```
 01  EMPLOYEE-MASTER-RECORD.
*        ** key field
     03  EMPLOYEE-ID                 PIC X(6).
     03  EMPLOYEE-DEPT-CODE          PIC X(4).
     03  EMPLOYEE-NAME               PIC X(30).
     03  EMPLOYEE-ADDR-1             PIC X(30).
     03  EMPLOYEE-ADDR-2             PIC X(30).
     03  EMPLOYEE-ADDR-3             PIC X(30).
     03  EMPLOYEE-ZIP-CODE           PIC X(5).
*        ** format (yyddd)
     03  EMPLOYEE-DATE-JOINED        PIC 9(5)
                                     DATE FORMAT YYXXX.
*        ** format (yymmdd)
     03  EMPLOYEE-DATE-TERMINATED    PIC 9(6)
                                     DATE FORMAT YYXXXX.
*        ** format (yyddd)
     03  EMPLOYEE-DATE-MAINTAINED    PIC 9(5)
                                     DATE FORMAT YYXXX.
*        ** format (yyddd)
     03  EMPLOYEE-BIRTH-DATE         PIC 9(5)
                                     DATE FORMAT YYXXX.
*        ** format (yyddd)
     03  EMPLOYEE-SECURITY-EXP       PIC 9(5) COMP-3
                                     DATE FORMAT YYXXX.
     03  FILLER                      PIC X(41).
     SKIP3
 FD  PRINT-EDIT-REPORT
     RECORDING MODE IS F
     RECORD CONTAINS 132 CHARACTERS
     LABEL RECORDS ARE OMITTED.
 01  PRINT-RECORD                    PIC X(132).
     SKIP3
     EJECT
 WORKING-STORAGE SECTION.

 01  FILE-STATUS-FIELDS.
     03  EMP-FILE-STATUS             PIC XX.
     03  PRT-FILE-STATUS             PIC XX.

 01  SWITCHES.
     03  VSAM-ERROR-SWITCH           PIC X VALUE SPACE.
         88  VSAM-ERROR                    VALUE "Y".
     03  EOF-SWITCH                  PIC X VALUE SPACE.
         88  EOF                           VALUE "Y".
     03  ERROR-SWITCH                PIC X VALUE SPACE.
         88  ERRORS                        VALUE "Y".

 01  WORK-DATE-YYMMDD         DATE FORMAT IS YYXXXX.
     03  WORK-DATE-YY  PIC XX DATE FORMAT IS YY.
     03  WORK-DATE-MM  PIC XX.
     03  WORK-DATE-DD  PIC XX.

 01  WORK-DATE-YYDDD                 PIC 9(5) DATE FORMAT YYXXX.
 01  WORK-TIME-HHMMSS.
     03  WRK-TIME-HH                 PIC XX.
     03  WRK-TIME-MM                 PIC XX.
     03  WRK-TIME-SS                 PIC XX.
     03  WRK-TIME-MS                 PIC XX.
 01  WORK-TIME-DISPLAY.
     03  WRK-TIME-HH                 PIC XX.
     03  FILLER                      PIC X VALUE ":".
     03  WRK-TIME-MM                 PIC XX.
     03  FILLER                      PIC X VALUE ":".
     03  WRK-TIME-SS                 PIC XX.

*    ** report headings & detail line
```

Figure 87 (Part 2 of 6). Final MLE Version of TARMU6/TARMU6E

```
01  REPORT-HEAD1.
    03  FILLER                     PIC X(01)  VALUE "1".
    03  FILLER                     PIC X(48)  VALUE
        "REPORT ID  TARMU6E".
    03  FILLER                     PIC X(61)  VALUE
        "EMPLOYEE SECURITY RENEWAL REPORT".
    03  RPT-HD1-DATE.
        05  RPT-HD1-MM             PIC XX.
        05  FILLER                 PIC X      VALUE "/".
        05  RPT-HD1-DD             PIC XX.
        05  FILLER                 PIC X      VALUE "/".
        05  RPT-HD1-YY             PIC XXXX
                                   DATE FORMAT YYYY.
    03  FILLER                     PIC XX     VALUE SPACES.
    03  RPT-HD1-TIME               PIC X(8).
    03  FILLER                     PIC XX     VALUE SPACES.

01  REPORT-HEAD2.
    03  FILLER                     PIC X(01)  VALUE "0".
    03  FILLER                     PIC X(44)  VALUE
        "   ID       NAME                    ".
    03  FILLER                     PIC X(88)  VALUE
        "DEPT  JOINED      SEC/EXPIRY".

01  REPORT-DETAIL.
    03  FILLER                     PIC X(01)  VALUE "0".
    03  FILLER                     PIC X(03)  VALUE SPACES.
    03  RPT-DET-ID                 PIC X(06).
    03  FILLER                     PIC X(03)  VALUE SPACES.
    03  RPT-DET-NAME-ADDR          PIC X(30).
    03  FILLER                     PIC X(02)  VALUE SPACES.
    03  RPT-DET-DEPT               PIC X(04).
    03  FILLER                     PIC X(02)  VALUE SPACES.
    03  RPT-DET-JOINED.
        05 RPT-DET-JOINED-MMDD PIC X(06).
        05 RPT-DET-JOINED-YY   PIC X(04) DATE FORMAT YYYY.
    03  FILLER                  PIC X(02)  VALUE SPACES.
    03  RPT-DET-SEC-EXP.
        05 RPT-DET-SEC-EXP-MMDD PIC X(06).
        05 RPT-DET-SEC-EXP-YY   PIC X(04) DATE FORMAT YYYY.
    03  FILLER                   PIC X(60)  VALUE SPACES.

01  REPORT-WORK-VARIABLES.
    03  REP-LINE-COUNT             PIC 9(2)   VALUE 99.

01  VSAM-ERROR-VARS.
    03  VSAM-ERROR-ACTION          PIC X(8).
    03  VSAM-ERROR-FILE            PIC X(10).
    03  VSAM-ERROR-STATUS          PIC X(2).

01  TARDATE-PARAMETERS.
    03  TARDATE-DATE               PIC X(8).
    03  TARDATE-DATE-RED1 REDEFINES TARDATE-DATE.
        05  TARDATE-DATE-YYDDD     PIC 9(5)
                                   DATE FORMAT YYXXX.
        05  FILLER                 PIC X(3).
    03  TARDATE-DATE-RED2 REDEFINES TARDATE-DATE.
        05  TARDATE-DATE-YYMMDD    PIC 9(6)
                                   DATE FORMAT YYXXXX.
        05  FILLER                 PIC X(2).
    03  TARDATE-DATE-RED3 REDEFINES TARDATE-DATE.
        05  TARDATE-DATE-MMDD      PIC X(6).
        05  TARDATE-DATE-YY        PIC 9(2)
                                   DATE FORMAT YY.
    03  TARDATE-INPUT-FORMAT       PIC X(8).
    03  TARDATE-OUTPUT-FORMAT      PIC X(8).
    03  TARDATE-MESSAGE            PIC X(30).

01  DISPLAY-DATE                   PIC 9(4) DATE FORMAT YYYY.

    EJECT
```

Figure 87 (Part 3 of 6). Final MLE Version of TARMU6/TARMU6E

```
 PROCEDURE DIVISION.

     PERFORM 100-INITIALIZE THRU 100-EXIT.

     IF  VSAM-ERROR   THEN
         NEXT SENTENCE
     ELSE
         PERFORM 200-MAIN-PROCESS THRU 200-EXIT.

     STOP RUN.


* ============================================================ *
     SKIP3
 100-INITIALIZE.

*    **  retrieve todays date

     ACCEPT WORK-DATE-YYMMDD FROM DATE.
     ACCEPT WORK-DATE-YYDDD  FROM DAY.
     ACCEPT WORK-TIME-HHMMSS FROM TIME.

     MOVE WORK-DATE-YY TO RPT-HD1-YY.
     MOVE WORK-DATE-DD TO RPT-HD1-DD.
     MOVE WORK-DATE-MM TO RPT-HD1-MM.
     MOVE CORRESPONDING WORK-TIME-HHMMSS TO WORK-TIME-DISPLAY.
     MOVE WORK-TIME-DISPLAY     TO RPT-HD1-TIME.

*    **  open files

     MOVE "OPEN FILE" TO VSAM-ERROR-ACTION.

     OPEN I-O EMPLOYEE-MASTER-FILE.
     IF  EMP-FILE-STATUS = "00" THEN
         NEXT SENTENCE
     ELSE
         MOVE "EMPMAST"       TO VSAM-ERROR-FILE
         MOVE EMP-FILE-STATUS TO VSAM-ERROR-STATUS
         PERFORM 999-VSAM-ERROR THRU 999-EXIT
     END-IF.


     OPEN OUTPUT PRINT-EDIT-REPORT.
     IF  PRT-FILE-STATUS = "00" THEN
         NEXT SENTENCE
     ELSE
         MOVE "REPFILE"      TO VSAM-ERROR-FILE
         MOVE PRT-FILE-STATUS TO VSAM-ERROR-STATUS
         PERFORM 999-VSAM-ERROR THRU 999-EXIT
     END-IF.

 100-EXIT.
     EXIT.
* ============================================================ *
     SKIP3
 200-MAIN-PROCESS.

     READ EMPLOYEE-MASTER-FILE
         AT END
         GO TO 200-EXIT
     END-READ.

     IF  EMPLOYEE-ID = "999999"  THEN
         GO TO 200-MAIN-PROCESS.
```

Figure 87 (Part 4 of 6). Final MLE Version of TARMU6/TARMU6E

```
           IF  FUNCTION UNDATE(EMPLOYEE-DATE-TERMINATED) = ZEROS THEN
      *    FIGURATIVE CONSTANT ZEROS NOT ACCEPTED BY FUNCTION DATEVAL.
      *    IF  EMPLOYEE-DATE-TERMINATED =
      *        FUNCTION DATEVAL(ZEROS, 'YYXXXX')  THEN

               IF  EMPLOYEE-SECURITY-EXP <= WORK-DATE-YYDDD THEN

                   PERFORM 300-PRINT-REPORT THRU 300-EXIT

                   PERFORM 400-UPDATE-FILE  THRU 400-EXIT

               END-IF

           END-IF.

           GO TO 200-MAIN-PROCESS.


       200-EXIT.
           EXIT.
      * ========================================================== *

       300-PRINT-REPORT.

      *    **  print edit report

           IF  REP-LINE-COUNT > 55 THEN
               WRITE PRINT-RECORD FROM REPORT-HEAD1
                    AFTER ADVANCING PAGE
               WRITE PRINT-RECORD FROM REPORT-HEAD2
                    AFTER ADVANCING 1 LINE
               MOVE ZERO TO REP-LINE-COUNT
           END-IF.

           MOVE SPACES                   TO  REPORT-DETAIL.
           WRITE PRINT-RECORD FROM REPORT-DETAIL
                    AFTER ADVANCING 1 LINE.

           MOVE SPACES                   TO  REPORT-DETAIL.
           MOVE EMPLOYEE-ID              TO  RPT-DET-ID.
           MOVE EMPLOYEE-DEPT-CODE       TO  RPT-DET-DEPT.
           MOVE EMPLOYEE-NAME            TO  RPT-DET-NAME-ADDR.
      *
           MOVE EMPLOYEE-DATE-JOINED     TO  TARDATE-DATE-YYDDD.
           MOVE "YYDDD"                  TO  TARDATE-INPUT-FORMAT.
           MOVE "MM/DD/YY"               TO  TARDATE-OUTPUT-FORMAT.
           CALL "TARDTE3"             USING  TARDATE-DATE
                                             TARDATE-INPUT-FORMAT
                                             TARDATE-OUTPUT-FORMAT
                                             TARDATE-MESSAGE.
           MOVE TARDATE-DATE-MMDD TO RPT-DET-JOINED-MMDD.
           MOVE TARDATE-DATE-YY   TO RPT-DET-JOINED-YY.
      *
           MOVE EMPLOYEE-SECURITY-EXP    TO  TARDATE-DATE-YYDDD.
           MOVE "YYDDD"                  TO  TARDATE-INPUT-FORMAT.
           MOVE "MM/DD/YY"               TO  TARDATE-OUTPUT-FORMAT.
           CALL "TARDTE3"             USING  TARDATE-DATE
                                             TARDATE-INPUT-FORMAT
                                             TARDATE-OUTPUT-FORMAT
                                             TARDATE-MESSAGE.

           ADD 1 TO  TARDATE-DATE-YY
              ON SIZE ERROR GO TO 500-SIZE-ERROR.

           IF  TARDATE-DATE-MMDD = "02/29/"  THEN
               MOVE "02/28/" TO TARDATE-DATE-MMDD

           END-IF.
```

Figure 87 (Part 5 of 6). Final MLE Version of TARMU6/TARMU6E

```
           MOVE TARDATE-DATE-MMDD TO RPT-DET-SEC-EXP-MMDD.
           MOVE TARDATE-DATE-YY   TO RPT-DET-SEC-EXP-YY.

           WRITE PRINT-RECORD FROM REPORT-DETAIL
                   AFTER ADVANCING 1 LINE.

           ADD 2 TO REP-LINE-COUNT.


       300-EXIT.
           EXIT.
      * ============================================================ *
       400-UPDATE-FILE.
           MOVE "MM/DD/YY"               TO  TARDATE-INPUT-FORMAT.
           MOVE "YYDDD"                  TO  TARDATE-OUTPUT-FORMAT.
           CALL "TARDTE3"             USING  TARDATE-DATE
                                             TARDATE-INPUT-FORMAT
                                             TARDATE-OUTPUT-FORMAT
                                             TARDATE-MESSAGE.

           MOVE TARDATE-DATE-YYDDD  TO  EMPLOYEE-SECURITY-EXP.
           REWRITE EMPLOYEE-MASTER-RECORD.
           IF  EMP-FILE-STATUS  NOT = "00" THEN
               MOVE "REWRITE FILE"  TO VSAM-ERROR-ACTION
               MOVE "EMPMAST"       TO VSAM-ERROR-FILE
               MOVE EMP-FILE-STATUS TO VSAM-ERROR-STATUS
               PERFORM 999-VSAM-ERROR THRU 999-EXIT
           END-IF.

       400-EXIT.
           EXIT.

       500-SIZE-ERROR.
           MOVE TARDATE-DATE-YY TO DISPLAY-DATE.
           DISPLAY "TARDATE-DATE-YY OUTSIDE CENTURY WINDOW IF DATE "
                   DISPLAY-DATE " INCREMENTED".
           GO TO 200-EXIT.

       500-EXIT.
           EXIT.
      * ============================================================ *
       999-VSAM-ERROR.

           SET VSAM-ERROR  TO  TRUE.

           DISPLAY VSAM-ERROR-ACTION
                   VSAM-ERROR-FILE
                   "STATUS ERROR "
                   VSAM-ERROR-STATUS.

       999-EXIT.
           EXIT.
```

Figure 87 (Part 6 of 6). Final MLE Version of TARMU6/TARMU6E

7.2.6 Coordinating the Implementation and Use of Automatic Windowing

Managing the use of the century window solution across application boundaries can be a difficult task requiring careful analysis, planning, and coordination. As our simple sample application shows, however, even the use of century windowing within a single application program requires orchestration if consistent results are to be produced. This section examines two aspects of this synchronization effort in connection with our sample application programs:

- Implementing changes in multiple object programs within a run unit.
- Using century windows consistently within the same run unit.

7.2.6.1 Using the MLE Enhancements in Called Subprograms

Although program TARMU6 has been subjected to the process of basic remediation in order to ensure that it continues to function correctly through the year 2000, it still contains a call to a subprogram that expects a field containing a 2-digit year as an argument. This subprogram, moreover, is called by other programs within our sample application. This use of shared resources requires that whatever changes might be introduced to the subprogram in connection with the introduction of century windowing in TARMU6 cannot be performed in isolation. This section examines the effect of possible changes to the calling and the target program.

Note

The transfer of control within the TARMU6 run unit continues to function even after MLE language elements have been added to the calling program because the data description entry of the level-01 argument passed to the subprogram does not contain a DATE FORMAT clause.

It is indisputable that some changes will have to be introduced to the subprogram TARDTE3 in its present form to enable it to continue functioning correctly through the year 2000. Whether or not this will require any additional modifications to the calling program depends upon the approach adopted to correct the target module:

- If basic remediation using century windowing is used to ensure that the subprogram continues to function correctly, then no further changes to the calling program would be necessary. The same century window should be used in both programs, however, to ensure that the date fields in the two programs are interpreted consistently.
- The current or subsequent use of internal bridging in the subprogram would also require no further immediate changes in the calling module. For a discussion of internal bridging, see 7.3, “Internal Bridging” on page 204.
- If the full field expansion solution is adopted for the subprogram, then the arguments passed to it would require expansion as well.
  - This could, theoretically, be accomplished by using the Millennium Language Extensions to expand the 2-digit date fields prior to the call and to contract them from 4 to 2 digits after control has been returned to the calling program. For an illustration of such a technique, see 7.3.5.3, “Year 2000 Compliant Application Program with MLE Enhancements” on page 221.
  - Because even the use of MLE for the automated expansion and contraction of the date fields passed as arguments to and from the called subprogram would require the introduction of new expanded date fields into the calling program, the same result could be achieved by expanding the original date fields passed between the compile units, as Figure 88 on page 200 illustrates.

```
01  TARDATE-PARAMETERS.
    03  TARDATE-DATE               PIC X(10).
    03  TARDATE-DATE-RED1 REDEFINES TARDATE-DATE.
        05  TARDATE-DATE-YYDDD     PIC 9(7)
                                   DATE FORMAT YYYYXXX.
        05  FILLER                 PIC X(3).
    03  TARDATE-DATE-RED2 REDEFINES TARDATE-DATE.
        05  TARDATE-DATE-YYMMDD    PIC 9(8)
                                   DATE FORMAT YYYYXXXX.
        05  FILLER                 PIC X(2).
    03  TARDATE-DATE-RED3 REDEFINES TARDATE-DATE.
        05  TARDATE-DATE-MMDD      PIC X(6).
        05  TARDATE-DATE-YY        PIC 9(4)
                                   DATE FORMAT YYYY.
    03  TARDATE-INPUT-FORMAT       PIC X(10).
    03  TARDATE-OUTPUT-FORMAT      PIC X(10).
    03  TARDATE-MESSAGE            PIC X(30).
```

Figure 88. Full Field Expansion Applied to the Arguments to TARDTE3

- Another alternative would be to modify the target program in such a way that it could return arguments containing date fields in the requested format with either 2- or 4-digit years. This is the solution adopted in the new program TARDTE3X illustrated in Figure 89 on page 201. This program utilizes the Language Environment callable date and time services to format date fields containing either 2- or 4-digit years as requested by converting them to and from a Lilian date. A Lilian date represents the number of days since October 15, 1582, the date the Gregorian calendar was adopted.

  Calling the new target program illustrated in Figure 89 on page 201 from TARMU6 would also require the expansion of the calling parameters from 8 to 10 characters as described above in connection with the full field expansion alternative.

7.2.6.2 Coordinating the Use of Century Windows Within a Program

There are other types of century windows available to COBOL programmers in addition to that specified by the YEARWINDOW compiler option introduced with MLE. A century window can also be specified by:

- An argument to the windowing COBOL intrinsic functions DATE-TO-YYYYMMDD, DAY-TO-YYYYDDD and YEAR-TO-YYYY
- The CEESCEN callable service in Language Environment

This section uses programs from our sample application to illustrate how the use of century windows by different functions within the same program can be coordinated.

Program TARDTE3X in Figure 89 on page 201 also illustrates a technique for ensuring that all the century windows used in all functions within a program are equal. By default, the century window applied by the Language Environment callable services used in TARDTE3X starts 80 years prior to the current year. This value can only be modified by invoking the CEESCEN callable service. Program TARDTE3X shows how the COBOL century window can be interrogated by the use of the YEARWINDOW intrinsic function introduced in MLE and the result used to set the Language Environment century window.

```
       CBL TEST(ALL,SYM) NODYNAM APOST RENT
       CBL DATEPROC(FLAG) YEARWINDOW(1935)
       IDENTIFICATION DIVISION.
       PROGRAM-ID.   TARDTE3X
       AUTHOR.       MERRILL BANI
       DATE-WRITTEN. JUNE 1997.
      * -------------------------------------------------------- */
      * LICENSED MATERIALS - PROPERTY OF IBM                     */
      *                                                          */
      * (C) COPYRIGHT IBM CORP. 1997                             */
      *                                                          */
      * US GOVERNMENT USERS RESTRICTED RIGHTS - USE,             */
      * DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP          */
      * SCHEDULE CONTRACT WITH IBM CORP.                         */
      * -------------------------------------------------------- */
      ************************************************************/
      *                                                          */
      *    VA2000 MVS COBOL TEST APPLICATION (Y2K VERSION)       */
      *                                                          */
      *    UTILITY PROGRAM TO VERIFY INPUT DATE, REFORMAT &      */
      *            RETURN USING LE FUNCTIONS                     */
      *            MODIFIED TO OPTIONALLY RETURN FULL YEAR       */
      ************************************************************/
       EJECT
       ENVIRONMENT DIVISION.

       INPUT-OUTPUT SECTION.

       FILE-CONTROL.


       EJECT
       DATA DIVISION.

       WORKING-STORAGE SECTION.


       01 LE-CALL-FC.
         02 CONDITION-TOKEN.
           88  CEE000            VALUE X"0000000000000000".
           88  CEE2EB            VALUE X"000309CB59C3C5C5".
           88  CEE2EC            VALUE X"000309CC59C3C5C5".
           88  CEE2ED            VALUE X"000309CD59C3C5C5".
           88  CEE2EH            VALUE X"000309D159C3C5C5".
           88  CEE2EL            VALUE X"000309D559C3C5C5".
           88  CEE2EM            VALUE X"000309D659C3C5C5".
           88  CEE2EO            VALUE X"000309D859C3C5C5".
           88  CEE2EP            VALUE X"000309D959C3C5C5".

          03 CASE-1.
            04 SEVERITY           PIC S9(4)      BINARY.
            04 MSG-NO             PIC S9(4)      BINARY.
          03 CASE-2          REDEFINES CASE-1.
            04 CLASS-CODE         PIC S9(4)      BINARY.
            04 CLAUSE-CODE        PIC S9(4)      BINARY.
          03 CASE-SEVERITY        PIC X.
          03 FACILITY-ID          PIC XXX.
         02 I-S-INFO              PIC S9(9)  BINARY.
```

Figure 89 (Part 1 of 3). Program TARDTE3X Using LE Callable Services

```
01  CEESCEN               PIC X(8) VALUE 'CEESCEN'.
01  CEEDAYS               PIC X(8) VALUE 'CEEDAYS'.
01  CEEDATE               PIC X(8) VALUE 'CEEDATE'.
01  LILIAN                PIC S9(9) BINARY.
01  CHRDATE               PIC X(80).
01  CURRENT-YEAR          PIC 9(4)  DATE FORMAT YYYY.
01  COBOL-START-YEAR      PIC 9(4)  DATE FORMAT YYYY.
77  START-LE-CW           PIC S9(9) BINARY.
01  IN-DATE.
    02  IN-DATE-LGTH      PIC S9(4)  BINARY.
    02  IN-DATE-TEXT.
        03  IN-DATE-CHAR  PIC X
                          OCCURS 0 TO 256 TIMES
                          DEPENDING ON IN-DATE-LGTH.
01  PICSTR.
    02  PICSTR-LGTH       PIC S9(4)  BINARY.
    02  PICSTR-TEXT.
        03  PICSTR-CHAR   PIC X
                          OCCURS 0 TO 256 TIMES
                          DEPENDING ON PICSTR-LGTH.

01  CHECK-LENGTH.
    02  CHECK-LENGTH-CHAR  PIC X OCCURS 15.

LINKAGE SECTION.

01  INPUT-DATE                    PIC X(10).
01  INPUT-FORMAT                  PIC X(10).
01  OUTPUT-FORMAT                 PIC X(10).
01  MSG                           PIC X(30).

PROCEDURE DIVISION USING INPUT-DATE
                         INPUT-FORMAT
                         OUTPUT-FORMAT
                         MSG.

    MOVE SPACES TO MSG.

*   Interrogate COBOL century window to modify Language
*   Environment century window.

    MOVE FUNCTION CURRENT-DATE(1:4) TO CURRENT-YEAR.
    COMPUTE COBOL-START-YEAR = FUNCTION YEARWINDOW.
    COMPUTE START-LE-CW = CURRENT-YEAR - COBOL-START-YEAR.
    CALL CEESCEN USING START-LE-CW LE-CALL-FC.
    IF  NOT CEE000   THEN
        MOVE "E- CEESCEN FAILED" TO MSG
        GO TO TARDATE-END
    END-IF.

*   **  call ceedays (convert input-date to lilian days)

    MOVE INPUT-FORMAT  TO  CHECK-LENGTH.
    MOVE 1             TO  IN-DATE-LGTH
    PERFORM UNTIL CHECK-LENGTH-CHAR(IN-DATE-LGTH) = SPACE
         ADD 1          TO IN-DATE-LGTH
    END-PERFORM.
    SUBTRACT 1          FROM IN-DATE-LGTH.
    MOVE IN-DATE-LGTH   TO PICSTR-LGTH.
    MOVE INPUT-DATE     TO IN-DATE-TEXT.
    MOVE INPUT-FORMAT   TO PICSTR-TEXT.
    CALL CEEDAYS USING IN-DATE, PICSTR, LILIAN, LE-CALL-FC.
    IF  NOT CEE000   THEN
        MOVE "E- CEEDAYS FAILED" TO MSG
        GO TO TARDATE-END
    END-IF.
```

Figure 89 (Part 2 of 3). Program TARDTE3X Using LE Callable Services

```
      *    **  call ceedate (convert lilian date to output format)

           MOVE OUTPUT-FORMAT  TO  CHECK-LENGTH.
           MOVE 1              TO  IN-DATE-LGTH
           PERFORM UNTIL CHECK-LENGTH-CHAR(IN-DATE-LGTH) = SPACE
                ADD 1          TO  IN-DATE-LGTH
           END-PERFORM.
           SUBTRACT 1          FROM IN-DATE-LGTH.
           MOVE IN-DATE-LGTH   TO  PICSTR-LGTH.
           MOVE OUTPUT-FORMAT  TO  PICSTR-TEXT.
           CALL CEEDATE USING LILIAN, PICSTR, CHRDATE, LE-CALL-FC.
           IF  NOT CEE000   THEN
               MOVE "E- CEEDATE FAILED" TO MSG
               GO TO TARDATE-END
           END-IF.
           MOVE CHRDATE     TO  INPUT-DATE.

       TARDATE-END.

           GOBACK.
```

Figure 89 (Part 3 of 3). Program TARDTE3X Using LE Callable Services

Figure 90 shows the program statements necessary to coordinate century windows between the YEARWINDOW compiler option and the COBOL intrinsic functions used to perform date expansion services. The YEARWINDOW intrinsic function allows you to define a field that can be used in calls to the COBOL intrinsic functions to define their century window. The second argument to these functions supplies the century window to be applied to the date provided as the first argument. This century window is specified as a number which is then added to the year at the time of execution to define the **end** of the 100-year interval used.

```
       CBL DATEPROC(FLAG) YEARWINDOW(1935) NODYNAM QUOTE SOURCE LIST
       IDENTIFICATION DIVISION.
       PROGRAM-ID.   TARMU6F
⋮
       DATA DIVISION.

       FILE SECTION.

       FD  EMPLOYEE-MASTER-FILE
           RECORD CONTAINS 200 CHARACTERS.
       01  EMPLOYEE-MASTER-RECORD.
⋮
      *        ** format (yyddd)
      *        assume that this cannot be windowed for some reason
      *        and that julian date is required output format
      *        function is integer, so argument must be numeric.

           03  EMPLOYEE-DATE-MAINTAINED    PIC 9(5).
⋮
       01  REPORT-DETAIL.
           03  RPT-DET-MAINTAINED          PIC 9(07).
⋮
       WORKING-STORAGE SECTION.
⋮
       01  COBOL-START-YEAR                PIC 9(4) DATE FORMAT YYYY.

       01  COBOL-INTRINSIC-ARG             PIC 9(4).

       01  CURRENT-YEAR                    PIC 9(4) DATE FORMAT YYYY.
⋮
```

Figure 90 (Part 1 of 2). Coordinating Windowing with COBOL Intrinsic Functions

```
       PROCEDURE DIVISION.
⋮
           MOVE FUNCTION CURRENT-DATE(1:4) TO CURRENT-YEAR.

           COMPUTE COBOL-START-YEAR = FUNCTION YEARWINDOW.

           COMPUTE
           COBOL-INTRINSIC-ARG = COBOL-START-YEAR + 99 - CURRENT-YEAR.
⋮
      *    RETURNED VALUE IS EXPANDED DATE FIELD WITH IMPLICIT
      *    FORMAT OF YYYYXXX.

           COMPUTE RPT-DET-MAINTAINED =
              FUNCTION DAY-TO-YYYYDDD
              (EMPLOYEE-DATE-MAINTAINED, COBOL-INTRINSIC-ARG)
```

Figure 90 (Part 2 of 2). Coordinating Windowing with COBOL Intrinsic Functions

7.3 Internal Bridging

This section illustrates in detail the internal bridging solution to the Year 2000 problem using one of the COBOL programs in our sample application. We will cover, step by step, the activities involved in using Millennium Language Extensions to implement this solution, reviewing the compiler diagnostic, and eliminating those messages to get the final program that is Year 2000 ready with MLE enhancements.

7.3.1 Definition

Internal bridging is one of the possible solution to the Year 2000 problem. It is a technique of converting 2-digit year dates to 4-digit year dates and converting the 4-digit year dates back to the 2-digit year dates in a program. Using the internal bridging technique, your program will be structured like this:

1. Declare the 2-digit year dates in the record as windowed date fields.
2. Declare new corresponding 4-digit year dates in the Working Storage as expanded date fields.
3. Read the input files with 2-digit year windowed dates.
4. Move the 2-digit year windowed dates into the 4-digit year expanded dates so that the compiler automatically expands them to the 4-digit year expanded dates.
5. In the main body of the program, use the 4-digit year expanded dates for all date processing. This means that you need to change all date field names from the 2-digit year date field names to the 4-digit year field names.
6. Window the 4-digit year expanded dates back to the 2-digit year windowed dates.
7. Write the 2-digit year windowed dates to the output files.

Note

If you do not want to change the date field names in the main body of the program as stated in step 5 above, you can do the following:

- Rename the 2-digit year windowed dates as something else, for example, you can prefix them with **OLD-**
- Use the original 2-digit year date field names as the field names of the new 4-digit year dates.

Using this method, you do not need to change the date field names in the main body of the program, since the existing date field names in the program already reflect the new 4-digit expanded date fields.

Using the internal bridging technique, your changes to the program logic are minimal. You simply add statements to expand and contract the dates, and change those statements that refer to dates to use the 4-digit year date fields instead of the 4-digit year fields in the records.

7.3.2 Applicability

This technique is applicable when your files and databases are still using 2-digit year dates, but you want to use the 4-digit expanded dates in the source program. There are several reason why you might want to do internal bridging instead of the basic remediation:

- This technique has a performance advantage over using the windowed dates. The action of viewing a windowed date field for a COBOL IF or MOVE statement still imposes some processor overhead.
- This process provides a convenient migration path to a full expanded-date solution.

7.3.3 Benefits of This Approach

The benefits of the internal bridging solution include:

- Straightforward changes to the program logic make testing easy.
- This solution will allow your programs to function into and beyond the Year 2000.
- This is a good incremental step toward a solution using full field expansion.
- Performance is good.
- No changes to the data are necessary.

7.3.4 Limitations of This Approach

Internal bridging involves windowed dates. Therefore, the same limitations apply as with the basic remediation.

7.3.5 Sample Application Program Illustrating This Approach

To illustrate the internal bridging technique, we use program TARMU3B from our sample application. We do the following:

- Illustrate, step by step, the activities performed to get the initial MLE revisions in order to implement the internal bridging solution. In these initial MLE revisions, we are focusing only on applying the internal bridging technique steps. As a result, we do not apply MLE revisions to all other date fields in the program. Also, we let the compiler give us warning or error messages.
- Review the Compiler Diagnostic Message that we get from the initial MLE revisions and, based on the messages, show you how to eliminate them.
- Review the Compiler Diagnostic Message that we get from the second MLE revisions and, based on the messages, show you how to eliminate them.
- Present the final Year 2000-ready application program with MLE enhancement from the initial MLE revisions.

7.3.5.1 Initial MLE Revisions

The program in Figure 91 is an example of an internal bridging program that manipulates date fields with 2-digit years. The program does the following:

- It reads a batch input file that contains information about employees — either existing employees or new hires.
- It verifies that the input details are valid.
- It updates (add or change) the employee VSAM file.

The Employee VSAM file contains five different types of 2-digit year dates. In this program, we use the internal bridging technique to process the 2-digit year dates.

```
       CBL TEST(ALL,SYM) NODYNAM APOST RENT
       CBL DATEPROC(FLAG) YEARWINDOW(1950)                              1
       IDENTIFICATION DIVISION.
       PROGRAM-ID.   TARMU3B.
       AUTHOR.       MERRILL BANI / RINA SUMIANTORO.
       DATE-WRITTEN. JUNE 1997 / APRIL 1998.
      * ------------------------------------------------------- */
      * LICENSED MATERIALS - PROPERTY OF IBM                    */
      *                                                         */
      * (C) COPYRIGHT IBM CORP. 1997                            */
      *                                                         */
      * US GOVERNMENT USERS RESTRICTED RIGHTS - USE,            */
      * DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP         */
      * SCHEDULE CONTRACT WITH IBM CORP.                        */
```

Figure 91 (Part 1 of 10). Using Internal Bridging to Process 2-Digit Year — Initial

```
      * ------------------------------------------------------- */
      * REMARKS.                                                */
      *                                                         */
      *    VA2000 MVS COBOL TEST APPLICATION                    */
      *                                                         */
      *    ONLINE UPDATE OF EMPLOYEE DATABASE                   */
      *                                                         */
      *    1. Read batch input                                  */
      *                                                         */
      *    2. Verify details                                    */
      *                                                         */
      *    3. If valid, update employee database                */
      *                                                         */
      * ------------------------------------------------------- */
       EJECT
       ENVIRONMENT DIVISION.

       INPUT-OUTPUT SECTION.

       FILE-CONTROL.

           SELECT EMPLOYEE-MASTER-FILE
                  ASSIGN TO EMPMAST
                  ORGANIZATION IS INDEXED
                  ACCESS IS RANDOM
                  RECORD KEY IS EMP-ID
                  FILE STATUS IS EMP-FILE-STATUS.

           SELECT DEPT-MASTER-FILE
                  ASSIGN TO DEPTMAST
                  ORGANIZATION IS INDEXED
                  ACCESS IS RANDOM
                  RECORD KEY IS DEPT-CODE
                  FILE STATUS IS DEPT-FILE-STATUS.

           SELECT INPUT-FILE
                  ASSIGN TO INPFILE
                  ORGANIZATION IS SEQUENTIAL
                  ACCESS IS SEQUENTIAL
                  FILE STATUS IS INP-FILE-STATUS.

       EJECT

       DATA DIVISION.

       FILE SECTION.

       FD  EMPLOYEE-MASTER-FILE
           RECORD CONTAINS 200 CHARACTERS.
      *
       01  EMPLOYEE-MASTER-RECORD.
      *        ** key field
           03  EMP-ID                      PIC X(6).
           03  EMP-DEPT-CODE               PIC X(4).
           03  EMP-NAME                    PIC X(30).
           03  EMP-ADDR-1                  PIC X(30).
           03  EMP-ADDR-2                  PIC X(30).
           03  EMP-ADDR-3                  PIC X(30).
           03  EMP-ZIP-CODE                PIC X(5).
      *        ** format (yyddd)
           03  OLD-EMP-DATE-JOINED         PIC 9(5) DATE FORMAT YYXXX.        2
      *        ** format (yymmdd)
           03  OLD-EMP-DATE-TERMINATED     PIC 9(6) DATE FORMAT YYXXXX.
      *        ** format (yyddd)
           03  OLD-EMP-DATE-MAINTAINED     PIC 9(5) DATE FORMAT YYXXX.
      *        ** format (yyddd)
           03  OLD-EMP-BIRTH-DATE          PIC 9(5) DATE FORMAT YYXXX.
      *        ** format (yyddd)
           03  OLD-EMP-SECURITY-EXP        PIC 9(5) COMP-3
                                                    DATE FORMAT YYXXX.
           03  FILLER                      PIC X(41).
```

Figure 91 (Part 2 of 10). Using Internal Bridging to Process 2-Digit Year — Initial

```
 FD  DEPT-MASTER-FILE
     RECORD CONTAINS 80 CHARACTERS.
*
 01  DEPT-MASTER-RECORD.
*        ** key field
     03  DEPT-CODE                   PIC X(4).
     03  DEPT-DESCRIPTION            PIC X(30).
*        ** format (yydddd) packed
     03  DEPT-DATE-MAINTAINED        PIC 9(5) COMP-3.
     03  FILLER                      PIC X(43).

 FD  INPUT-FILE
     RECORD CONTAINS 167 CHARACTERS.
*
 01  INPUT-RECORD.
*        ** key field
     03  INP-EMP-ID                  PIC X(6).
     03  INP-EMP-DEPT-CODE           PIC X(4).
     03  INP-EMP-NAME                PIC X(30).
     03  INP-EMP-ADDR-1              PIC X(30).
     03  INP-EMP-ADDR-2              PIC X(30).
     03  INP-EMP-ADDR-3              PIC X(30).
     03  INP-EMP-ZIP-CODE            PIC X(5).
*        ** format (yy/mm/dd)
     03  INP-EMP-DATE-JOINED         PIC 9(8).
*        ** format (yy/mm/dd)
     03  INP-EMP-DATE-TERMINATED     PIC 9(8).
*        ** format (yy/mm/dd)
     03  INP-EMP-BIRTH-DATE          PIC 9(8).
*        ** format (yy/mm/dd)
     03  INP-EMP-SECURITY-EXP        PIC 9(8).

 WORKING-STORAGE SECTION.

 01  OUTPUT-RECORD.
*        ** key field
     03  OUT-EMP-ID                  PIC X(6).
     03  OUT-EMP-DEPT-CODE           PIC X(4).
     03  OUT-EMP-NAME                PIC X(30).
     03  OUT-EMP-ADDR-1              PIC X(30).
     03  OUT-EMP-ADDR-2              PIC X(30).
     03  OUT-EMP-ADDR-3              PIC X(30).
     03  OUT-EMP-ZIP-CODE            PIC X(5).
*        ** format (yy/mm/dd)
     03  OUT-EMP-DATE-JOINED         PIC X(8).
*        ** format (yy/mm/dd)
     03  OUT-EMP-DATE-TERMINATED     PIC X(8).
*        ** format (yy/mm/dd)
     03  OUT-EMP-DATE-MAINTAINED     PIC X(8).
*        ** format (yy/mm/dd)
     03  OUT-EMP-BIRTH-DATE          PIC X(8).
*        ** format (yy/mm/dd)
     03  OUT-EMP-SECURITY-EXP        PIC X(8).
     03  FILLER                      PIC X(41).

 01  SWITCHES.
     03  ERROR-SWITCH                PIC X VALUE SPACE.
         88  ERRORS                        VALUE "Y".
 01  ADD-RECORD                      PIC X VALUE SPACES.
 01  CHANGE-RECORD                   PIC X VALUE SPACES.

 01  SWITCH-OFF                      PIC X VALUE "N".

*    ** message table
```

Figure 91 (Part 3 of 10). Using Internal Bridging to Process 2-Digit Year — Initial

```
 01  MESSAGE-TABLE.
     03  FILLER  PIC X(30) VALUE "I-RECORD ADDED             ".
     03  FILLER  PIC X(30) VALUE "I-RECORD CHANGED           ".
     03  FILLER  PIC X(30) VALUE "E-DEPARTMENT CODE INVALID  ".
     03  FILLER  PIC X(30) VALUE "E-ZIP CODE NOT NUMERIC     ".
     03  FILLER  PIC X(30) VALUE "E-INVALID DATE             ".
     03  FILLER  PIC X(30) VALUE "I-ENTER EMPLOYEE NUMBER    ".
     03  FILLER  PIC X(30) VALUE "I-ENTER EMPLOYEE DETAILS   ".
     03  FILLER  PIC X(30) VALUE "I-ENTER CHANGE DETAILS     ".
     03  FILLER  PIC X(30) VALUE "I-SCR VALID PF1O TO UPDATE ".
     03  FILLER  PIC X(30) VALUE "E-NAME MISSING             ".
     03  FILLER  PIC X(30) VALUE "E-ADDRESS LINE 1 MISSING   ".
     03  FILLER  PIC X(30) VALUE "E-JOINED > TERMINATED DATE ".
     03  FILLER  PIC X(30) VALUE "E-DATE MISSING             ".
     03  FILLER  PIC X(30) VALUE "E-EMPLOYEE NO NOT NUMERIC  ".

 01  MSG-TABLE-RED REDEFINES MESSAGE-TABLE.
     03  MSG OCCURS 14 TIMES.
         05  FILLER                  PIC X(30).

*    ** work variables

 01  WORK-VARS.
     03  WORK-TODAYS-MMDDYY          PIC 9(8).
     03  WORK-MSG-CODE               PIC 99.
     03  WORK-EIB-DATE               PIC 9(7).
     03  WORK-EIB-DATE-RED REDEFINES WORK-EIB-DATE.
         05  FILLER                  PIC X(1).
         05  WORK-EIB-CENTURY        PIC X(1).
         05  WORK-EIB-YYDDD          PIC X(5).
     03  WORK-JOINED-YYDDD           PIC 9(5).
     03  WORK-JOINED-MMDDYY.
         05  WORK-JOINED-MMDD        PIC X(6).
         05  WORK-JOINED-YY          PIC 99.
     03  WORK-SEC-EXP                PIC 9(5).
     03  WORK-TERMINATED-YYDDD       PIC 9(5).

 01  FILE-STATUS-FIELDS.
     03  EMP-FILE-STATUS             PIC XX.
     03  DEPT-FILE-STATUS            PIC XX.
     03  INP-FILE-STATUS             PIC XX.

 01  EMP-LENGTH                      PIC S9(4) COMP VALUE +200.
 01  DEP-LENGTH                      PIC S9(4) COMP VALUE +80.
 01  RESPONSE                        PIC S9(8) COMP VALUE +0.
 01  LAST-EMP-ID                     PIC X(6)  VALUE SPACES.

 01  TARDATE-PARAMETERS.
     03  TARDATE-DATE                PIC X(8).
     03  TARDATE-DATE-RED1 REDEFINES TARDATE-DATE.
         05  TARDATE-DATE-YYDDD      PIC 9(5).
         05  FILLER                  PIC X(3).
     03  TARDATE-DATE-RED2 REDEFINES TARDATE-DATE.
         05  TARDATE-DATE-YYMMDD     PIC 9(6).
         05  FILLER                  PIC X(2).
     03  TARDATE-INPUT-FORMAT        PIC X(8).
     03  TARDATE-OUTPUT-FORMAT       PIC X(8).
     03  TARDATE-MESSAGE             PIC X(30).

*    ** expanded date variables

*    ** format (yyyyddd)  expanded
 01  EMP-DATE-JOINED             PIC 9(7) DATE FORMAT YYYYXXX.          3
*    ** format (yyyymmdd) expanded
 01  EMP-DATE-TERMINATED         PIC 9(8) DATE FORMAT YYYYXXXX.
*    ** format (yyyyddd)  expanded
 01  EMP-DATE-MAINTAINED         PIC 9(7) DATE FORMAT YYYYXXX.
*    ** format (yyyyddd)  expanded
 01  EMP-BIRTH-DATE              PIC 9(7) DATE FORMAT YYYYXXX.
*    ** format (yyyyddd)  expanded
 01  EMP-SECURITY-EXP            PIC 9(7) COMP-3
                                      DATE FORMAT YYYYXXX.
     EJECT
```

Figure 91 (Part 4 of 10). Using Internal Bridging to Process 2-Digit Year — Initial

```
 PROCEDURE DIVISION.

     OPEN I-O   EMPLOYEE-MASTER-FILE
     OPEN INPUT DEPT-MASTER-FILE
     OPEN INPUT INPUT-FILE

*    **  retrieve todays date

     ACCEPT WORK-EIB-YYDDD FROM  DAY.
     MOVE   WORK-EIB-YYDDD TO    TARDATE-DATE.
     MOVE   "YYDDD"        TO    TARDATE-INPUT-FORMAT.
     MOVE   "MM/DD/YY"     TO    TARDATE-OUTPUT-FORMAT.
     CALL   "TARDTE3"      USING TARDATE-DATE
                                 TARDATE-INPUT-FORMAT
                                 TARDATE-OUTPUT-FORMAT
                                 TARDATE-MESSAGE.
     MOVE TARDATE-DATE     TO    WORK-TODAYS-MMDDYY.

     PERFORM 200-READ-INPUT-FILE THRU 200-EXIT.

     STOP RUN.

*================================================================

 200-READ-INPUT-FILE.

     READ INPUT-FILE
         AT END
         GO TO 200-EXIT
     END-READ.

*    **  validate employee number

     IF INP-EMP-ID = ZERO THEN
         MOVE 6              TO  WORK-MSG-CODE
         DISPLAY MSG(WORK-MSG-CODE)
         GO TO 200-READ-NEXT-INPUT-FILE
     ELSE
        IF INP-EMP-ID IS NOT NUMERIC THEN
           MOVE 14             TO  WORK-MSG-CODE
           DISPLAY MSG(WORK-MSG-CODE)
           GO TO 200-READ-NEXT-INPUT-FILE
        END-IF
     END-IF.

     PERFORM 400-READ-EMPLOYEE-MASTER-FILE THRU 400-EXIT
     PERFORM 500-VALIDATE-INPUT-DATA THRU 500-EXIT.
     IF NOT ERRORS
        PERFORM 600-UPDATE-FILE THRU 600-EXIT
     END-IF.

 200-READ-NEXT-INPUT-FILE.

     DISPLAY " "
     GO TO 200-READ-INPUT-FILE.

 200-EXIT.
     EXIT.

*================================================================

 400-READ-EMPLOYEE-MASTER-FILE.

     MOVE SPACES TO CHANGE-RECORD ADD-RECORD.
     MOVE SPACES TO EMPLOYEE-MASTER-RECORD.

     MOVE INP-EMP-ID TO EMP-ID
     READ EMPLOYEE-MASTER-FILE                                    4
          KEY IS EMP-ID.
```

Figure 91 (Part 5 of 10). Using Internal Bridging to Process 2-Digit Year — Initial

```
*    **  MOVE THE OLD NON-EXPANDED DATE VARIABLES TO THE NEW
*    **  EXPANDED DATE VARIABLES AFTER READING THE FILE.

     MOVE OLD-EMP-DATE-JOINED      TO EMP-DATE-JOINED.                  5
     MOVE OLD-EMP-DATE-TERMINATED  TO EMP-DATE-TERMINATED.
     MOVE OLD-EMP-DATE-MAINTAINED  TO EMP-DATE-MAINTAINED.
     MOVE OLD-EMP-BIRTH-DATE       TO EMP-BIRTH-DATE.
     MOVE OLD-EMP-SECURITY-EXP     TO EMP-SECURITY-EXP.

     MOVE LOW-VALUES TO OUTPUT-RECORD.
     MOVE EMP-ID     TO OUT-EMP-ID.

     IF EMP-FILE-STATUS = "23" THEN
         MOVE "Y"                  TO ADD-RECORD
         MOVE SPACES               TO OUT-EMP-DEPT-CODE
                                      OUT-EMP-NAME
                                      OUT-EMP-ADDR-1
                                      OUT-EMP-ADDR-2
                                      OUT-EMP-ADDR-3
                                      OUT-EMP-ZIP-CODE
                                      OUT-EMP-DATE-JOINED
                                      OUT-EMP-BIRTH-DATE
                                      OUT-EMP-DATE-TERMINATED
                                      OUT-EMP-SECURITY-EXP
     ELSE
         MOVE "Y"                  TO CHANGE-RECORD
         MOVE EMP-DEPT-CODE        TO OUT-EMP-DEPT-CODE
         MOVE EMP-NAME             TO OUT-EMP-NAME
         MOVE EMP-ADDR-1           TO OUT-EMP-ADDR-1
         MOVE EMP-ADDR-2           TO OUT-EMP-ADDR-2
         MOVE EMP-ADDR-3           TO OUT-EMP-ADDR-3
         MOVE EMP-ZIP-CODE         TO OUT-EMP-ZIP-CODE
*
         MOVE EMP-DATE-JOINED      TO TARDATE-DATE
         MOVE "YYDDD"              TO TARDATE-INPUT-FORMAT
         MOVE "MM/DD/YY"           TO TARDATE-OUTPUT-FORMAT
         CALL "TARDTE3" USING TARDATE-DATE
                              TARDATE-INPUT-FORMAT
                              TARDATE-OUTPUT-FORMAT
                              TARDATE-MESSAGE
         MOVE TARDATE-DATE         TO OUT-EMP-DATE-JOINED
*
         MOVE EMP-BIRTH-DATE       TO TARDATE-DATE
         MOVE "YYDDD"              TO TARDATE-INPUT-FORMAT
         MOVE "MM/DD/YY"           TO TARDATE-OUTPUT-FORMAT
         CALL "TARDTE3" USING TARDATE-DATE
                              TARDATE-INPUT-FORMAT
                              TARDATE-OUTPUT-FORMAT
                              TARDATE-MESSAGE
         MOVE TARDATE-DATE         TO OUT-EMP-BIRTH-DATE
*
         IF  EMP-DATE-TERMINATED > ZEROS THEN
             MOVE EMP-DATE-TERMINATED  TO TARDATE-DATE
             MOVE "YYMMDD"             TO TARDATE-INPUT-FORMAT
             MOVE "MM/DD/YY"           TO TARDATE-OUTPUT-FORMAT
             CALL "TARDTE3" USING TARDATE-DATE
                                  TARDATE-INPUT-FORMAT
                                  TARDATE-OUTPUT-FORMAT
                                  TARDATE-MESSAGE
             MOVE TARDATE-DATE         TO OUT-EMP-DATE-TERMINATED
         ELSE
             MOVE SPACES               TO OUT-EMP-DATE-TERMINATED
         END-IF
```

Figure 91 (Part 6 of 10). Using Internal Bridging to Process 2-Digit Year — Initial

```
*
         MOVE EMP-SECURITY-EXP     TO TARDATE-DATE
         MOVE "YYDDD"              TO TARDATE-INPUT-FORMAT
         MOVE "MM/DD/YY"           TO TARDATE-OUTPUT-FORMAT
         CALL "TARDTE3" USING TARDATE-DATE
                              TARDATE-INPUT-FORMAT
                              TARDATE-OUTPUT-FORMAT
                              TARDATE-MESSAGE
         MOVE TARDATE-DATE         TO OUT-EMP-SECURITY-EXP
*
     END-IF.

 400-EXIT.
     EXIT.

*================================================================

 500-VALIDATE-INPUT-DATA.

     MOVE SWITCH-OFF  TO  ERROR-SWITCH.

*    **    verify dept code

     MOVE INP-EMP-DEPT-CODE TO DEPT-CODE
     READ DEPT-MASTER-FILE
          KEY IS DEPT-CODE
          IF DEPT-FILE-STATUS = "23" THEN
             MOVE 3 TO WORK-MSG-CODE
             DISPLAY MSG(WORK-MSG-CODE)
             SET ERRORS TO TRUE
          END-IF

     IF  INP-EMP-NAME IS NOT > SPACES   THEN
         MOVE 10        TO WORK-MSG-CODE
         DISPLAY MSG(WORK-MSG-CODE)
         SET ERRORS TO TRUE
     END-IF

     IF  INP-EMP-ADDR-1 IS NOT > SPACES   THEN
         MOVE 11        TO WORK-MSG-CODE
         DISPLAY MSG(WORK-MSG-CODE)
         SET ERRORS TO TRUE
     END-IF

     IF  INP-EMP-ZIP-CODE IS NOT NUMERIC   THEN
         MOVE 4         TO WORK-MSG-CODE
         DISPLAY MSG(WORK-MSG-CODE)
         SET ERRORS TO TRUE
     END-IF

     IF  INP-EMP-DATE-JOINED > SPACES THEN
         MOVE INP-EMP-DATE-JOINED   TO TARDATE-DATE
         MOVE "MM/DD/YY"      TO TARDATE-INPUT-FORMAT
         MOVE "YYDDD"         TO TARDATE-OUTPUT-FORMAT
         CALL "TARDTE3" USING TARDATE-DATE
                              TARDATE-INPUT-FORMAT
                              TARDATE-OUTPUT-FORMAT
                              TARDATE-MESSAGE
         IF  TARDATE-MESSAGE NOT = SPACES THEN
             MOVE 5     TO WORK-MSG-CODE
             DISPLAY MSG(WORK-MSG-CODE)
             SET ERRORS TO TRUE
         ELSE
             MOVE TARDATE-DATE-YYDDD TO WORK-JOINED-YYDDD
             MOVE INP-EMP-DATE-JOINED TO WORK-JOINED-MMDDYY
         END-IF
     ELSE
         MOVE WORK-TODAYS-MMDDYY TO OUT-EMP-DATE-JOINED
                                    WORK-JOINED-MMDDYY
         MOVE WORK-EIB-YYDDD     TO WORK-JOINED-YYDDD
     END-IF
```

Figure 91 (Part 7 of 10). Using Internal Bridging to Process 2-Digit Year — Initial

```
           IF  INP-EMP-BIRTH-DATE > SPACES     THEN
               MOVE INP-EMP-BIRTH-DATE   TO TARDATE-DATE
               MOVE "MM/DD/YY"      TO TARDATE-INPUT-FORMAT
               MOVE "YYDDD"         TO TARDATE-OUTPUT-FORMAT
               CALL "TARDTE3" USING TARDATE-DATE
                                    TARDATE-INPUT-FORMAT
                                    TARDATE-OUTPUT-FORMAT
                                    TARDATE-MESSAGE
               IF  TARDATE-MESSAGE NOT = SPACES THEN
                   MOVE 5     TO WORK-MSG-CODE
                   DISPLAY MSG(WORK-MSG-CODE)
                   SET ERRORS TO TRUE
               END-IF
           ELSE
               MOVE 13    TO WORK-MSG-CODE
               DISPLAY MSG(WORK-MSG-CODE)
               SET ERRORS TO TRUE
           END-IF

           IF  INP-EMP-DATE-TERMINATED > SPACES THEN
               MOVE INP-EMP-DATE-TERMINATED TO TARDATE-DATE
               MOVE "MM/DD/YY"      TO TARDATE-INPUT-FORMAT
               MOVE "YYDDD"         TO TARDATE-OUTPUT-FORMAT
               CALL "TARDTE3" USING TARDATE-DATE
                                    TARDATE-INPUT-FORMAT
                                    TARDATE-OUTPUT-FORMAT
                                    TARDATE-MESSAGE
               IF  TARDATE-MESSAGE NOT = SPACES THEN
                   MOVE 5     TO WORK-MSG-CODE
                   DISPLAY MSG(WORK-MSG-CODE)
                   SET ERRORS TO TRUE
               ELSE
                   MOVE OUT-EMP-DATE-TERMINATED TO OUT-EMP-SECURITY-EXP
                   MOVE TARDATE-DATE-YYDDD TO WORK-TERMINATED-YYDDD
                   IF  WORK-TERMINATED-YYDDD < WORK-JOINED-YYDDD
                       MOVE 12    TO WORK-MSG-CODE
                       DISPLAY MSG(WORK-MSG-CODE)
                       SET ERRORS TO TRUE
                   END-IF
               END-IF
           END-IF.

           IF  INP-EMP-SECURITY-EXP > SPACES     THEN
               MOVE INP-EMP-SECURITY-EXP TO TARDATE-DATE
               MOVE "MM/DD/YY"      TO TARDATE-INPUT-FORMAT
               MOVE "YYDDD"         TO TARDATE-OUTPUT-FORMAT
               CALL "TARDTE3" USING TARDATE-DATE
                                    TARDATE-INPUT-FORMAT
                                    TARDATE-OUTPUT-FORMAT
                                    TARDATE-MESSAGE
               IF  TARDATE-MESSAGE NOT = SPACES THEN
                   MOVE 5     TO WORK-MSG-CODE
                   DISPLAY MSG(WORK-MSG-CODE)
                   SET ERRORS TO TRUE
               END-IF
           ELSE
               COMPUTE WORK-JOINED-YY = WORK-JOINED-YY + 1
               IF  WORK-JOINED-MMDD = "02/29/" THEN
                   MOVE "02/28/"  TO WORK-JOINED-MMDD
               END-IF
               MOVE WORK-JOINED-MMDDYY TO OUT-EMP-SECURITY-EXP
           END-IF.

       500-EXIT.
           EXIT.
```

Figure 91 (Part 8 of 10). Using Internal Bridging to Process 2-Digit Year — Initial

```
*=============================================================

 600-UPDATE-FILE.

     MOVE SPACES             TO  EMPLOYEE-MASTER-RECORD.
     MOVE INP-EMP-ID         TO  EMP-ID.
     MOVE INP-EMP-DEPT-CODE  TO  EMP-DEPT-CODE.
     MOVE INP-EMP-NAME       TO  EMP-NAME.
     MOVE INP-EMP-ADDR-1     TO  EMP-ADDR-1.
     MOVE INP-EMP-ADDR-2     TO  EMP-ADDR-2.
     MOVE INP-EMP-ADDR-3     TO  EMP-ADDR-3.
     MOVE INP-EMP-ZIP-CODE   TO  EMP-ZIP-CODE.
     MOVE INP-EMP-DATE-JOINED TO  TARDATE-DATE
     MOVE "MM/DD/YY"         TO  TARDATE-INPUT-FORMAT
     MOVE "YYDDD"            TO  TARDATE-OUTPUT-FORMAT
     CALL "TARDTE3" USING        TARDATE-DATE
                                 TARDATE-INPUT-FORMAT
                                 TARDATE-OUTPUT-FORMAT
                                 TARDATE-MESSAGE.
     MOVE TARDATE-DATE-YYDDD TO  EMP-DATE-JOINED.
     MOVE INP-EMP-BIRTH-DATE TO  TARDATE-DATE
     MOVE "MM/DD/YY"         TO  TARDATE-INPUT-FORMAT
     MOVE "YYDDD"            TO  TARDATE-OUTPUT-FORMAT
     CALL "TARDTE3" USING        TARDATE-DATE
                                 TARDATE-INPUT-FORMAT
                                 TARDATE-OUTPUT-FORMAT
                                 TARDATE-MESSAGE.
     MOVE TARDATE-DATE-YYDDD TO  EMP-BIRTH-DATE.

     IF  INP-EMP-DATE-TERMINATED > SPACES THEN
         MOVE INP-EMP-DATE-TERMINATED TO TARDATE-DATE
         MOVE "MM/DD/YY"      TO TARDATE-INPUT-FORMAT
         MOVE "YYMMDD"        TO TARDATE-OUTPUT-FORMAT
         CALL "TARDTE3" USING    TARDATE-DATE
                                 TARDATE-INPUT-FORMAT
                                 TARDATE-OUTPUT-FORMAT
                                 TARDATE-MESSAGE
         MOVE TARDATE-DATE-YYMMDD TO  EMP-DATE-TERMINATED
     ELSE
         MOVE ZEROS           TO  EMP-DATE-TERMINATED
     END-IF.
     MOVE INP-EMP-SECURITY-EXP TO  TARDATE-DATE
     MOVE "MM/DD/YY"         TO  TARDATE-INPUT-FORMAT
     MOVE "YYDDD"            TO  TARDATE-OUTPUT-FORMAT
     CALL "TARDTE3" USING        TARDATE-DATE
                                 TARDATE-INPUT-FORMAT
                                 TARDATE-OUTPUT-FORMAT
                                 TARDATE-MESSAGE.
     MOVE TARDATE-DATE-YYDDD TO  EMP-SECURITY-EXP.

     MOVE WORK-EIB-YYDDD      TO  EMP-DATE-MAINTAINED.

*    **  MOVE THE NEW EXPANDED DATE VARIABLES BACK TO THE OLD          6
*    **  NON-EXPANDED DATE VARIABLES BEFORE UPDATING THE FILE.

     COMPUTE OLD-EMP-DATE-JOINED      = EMP-DATE-JOINED
       ON SIZE ERROR GO TO 600-OUT-OF-WINDOW-ERROR.

     IF FUNCTION UNDATE(EMP-DATE-TERMINATED) = 00000000
       MOVE EMP-DATE-TERMINATED TO OLD-EMP-DATE-TERMINATED
     ELSE
       COMPUTE OLD-EMP-DATE-TERMINATED  = EMP-DATE-TERMINATED
         ON SIZE ERROR GO TO 600-OUT-OF-WINDOW-ERROR
     END-IF.

     COMPUTE OLD-EMP-DATE-MAINTAINED  = EMP-DATE-MAINTAINED
       ON SIZE ERROR GO TO 600-OUT-OF-WINDOW-ERROR.

     COMPUTE OLD-EMP-BIRTH-DATE       = EMP-BIRTH-DATE
       ON SIZE ERROR GO TO 600-OUT-OF-WINDOW-ERROR.

     COMPUTE OLD-EMP-SECURITY-EXP     = EMP-SECURITY-EXP
       ON SIZE ERROR GO TO 600-OUT-OF-WINDOW-ERROR.
```

Figure 91 (Part 9 of 10). Using Internal Bridging to Process 2-Digit Year — Initial

```
           IF  ADD-RECORD = "Y"  THEN
               WRITE EMPLOYEE-MASTER-RECORD                          7
               MOVE 1    TO  WORK-MSG-CODE
               DISPLAY MSG(WORK-MSG-CODE)
               MOVE SPACES TO ADD-RECORD
               GO TO 600-EXIT
           ELSE
               REWRITE EMPLOYEE-MASTER-RECORD                        7
               MOVE 2    TO  WORK-MSG-CODE
               DISPLAY MSG(WORK-MSG-CODE)
               MOVE SPACES TO CHANGE-RECORD
               GO TO 600-EXIT
           END-IF.

       600-OUT-OF-WINDOW-ERROR.                                      6
           DISPLAY 'OUT OF WINDOW CONDITION HAS OCCURRED'.

       600-EXIT.
           EXIT.
```

Figure 91 (Part 10 of 10). Using Internal Bridging to Process 2-Digit Year — Initial

The following notes apply to Figure 91 on page 206:

1 The DATEPROC compiler option enables the Millennium Language Extensions. The YEARWINDOW compiler option specifies the century window to be used for this compile unit.

2 The 2-digit year date fields in Employee file have been renamed by adding **OLD-** in front of the original field names

The DATE FORMAT clauses have been added to the 2-digit year date fields in the Employee file so that COBOL will recognize them as windowed date fields.

3 Declare new corresponding 4-digit year dates in the Working Storage section and add the DATE FORMAT clause so that COBOL will recognize them as expanded date fields. Use the original 2-digit year date field names as their names correspondingly.

4 The input files is read with 2-digit year windowed dates.

5 Move the 2-digit year windowed dates of the employee VSAM files into the 4-digit expanded dates so that the compiler automatically expand them to the 4-digit year expanded dates.

Note

By renaming the 2-digit year date fields and using the 2-digit year date field names as the 4-digit year date field names (as explained in Notes **2** and **3**) you **do not** need to change the date field names in the main body of the program, since the existing date field names in the program already reflect the new 4-digit expanded date fields.

6 Window the 4-digit year expanded dates back to the 2-digit year windowed dates using the COMPUTE statements and MOVE statements.

When converting the 4-digit year dates back to 2-digit year dates we have to be aware of the possibility that the year could be outside the century window. For example, the century window is set to be 1905 - 2004, but during the processing of the year in the program using the expanded date, the year is getting a value of 2006, which is outside the century window. Simply storing the expanded date as a 2-digit year would be incorrect. To protect against this, you can use the COMPUTE statement to store the date, with the ON SIZE ERROR phrase to detect whether or not the date is within

the century window as shown on the program. The result of the COMPUTE statement depends on whether the ON SIZE ERROR is specified on the COMPUTE statement, as follows:

- If SIZE ERROR is specified, the receiving field **is not changed**, and the SIZE ERROR imperative statement is executed.
- If SIZE ERROR is not specified, the receiving field **is stored** in the receiving field with the two left-hand digits truncated.

In the body of the program, the EMP-DATE-TERMINATED is set to zeros (00000000) if the input is spaces (no input) and these zeros will be stored to the file later on. Thus, the value of zeros in the date field EMP-DATE-TERMINATED does not represent a date; it just shows that the employee is still active, not terminated yet. Therefore, before we convert the 4-digit year EMP-DATE-TERMINATED field back to the 2-digit year date, we should check whether the value is zeros or not:

- If it is zeros (not a date), then simply use the MOVE statement.
- If it is not zeros (a date), then use the COMPUTE / ON SIZE ERROR statement.

To check this, you can use the UNDATE intrinsic function.

7 Write the 2-digit year windowed dates to the output files.

7.3.5.2 Eliminating Compiler Diagnostic Messages — Initial MLE Revision

After applying the initial MLE revision to the program, we compiled the program using the JCL shown in Figure 76 on page 177. We got several compiler diagnostic messages as a result. Figure 92 on page 217 shows the information-level compiler diagnostic messages produced by the compiler. These messages are for information only. They mainly show that there is a date field usage in a statement, or a windowing has occurred. We do not need to change anything in the program.

```
LineID  Message code  Message text

    63  IGYDS1454-I   A group item contained one or more subordinate date fields.

    96  IGYGR1216-I   A "RECORDING MODE" of "F" was assumed for file "INPUT-FILE".

   235  IGYPA3290-I   The statement contained date logic.

                      Same message on line:    300    301    302    303    304    330    339    348    349    361    517    525
                                               535    537    546    548    553    556    557    559    563    566    569

     0  IGYPA3309-I   The windowed date value in "OLD-EMP-DATE-JOINED (NUMERIC INTEGER)" will be expanded and moved into
                      "EMP-DATE-JOINED (NUMERIC INTEGER)".

   301  IGYPA3309-I   The windowed date value in "OLD-EMP-DATE-TERMINATED (NUMERIC INTEGER)" will be expanded and moved into
                      "EMP-DATE-TERMINATED (NUMERIC INTEGER)".

   302  IGYPA3309-I   The windowed date value in "OLD-EMP-DATE-MAINTAINED (NUMERIC INTEGER)" will be expanded and moved into
                      "EMP-DATE-MAINTAINED (NUMERIC INTEGER)".

   303  IGYPA3309-I   The windowed date value in "OLD-EMP-BIRTH-DATE (NUMERIC INTEGER)" will be expanded and moved into
                      "EMP-BIRTH-DATE (NUMERIC INTEGER)".

   304  IGYPA3309-I   The windowed date value in "OLD-EMP-SECURITY-EXP (PACKED INTEGER)" will be expanded and moved into
                      "EMP-SECURITY-EXP (PACKED INTEGER)".

   553  IGYPA3299-I   The "SIZE ERROR" imperative statements will be executed if the year part of the date result is not valid for
                      windowed date receiving field "OLD-EMP-DATE-JOINED (NUMERIC INTEGER)".

   559  IGYPA3299-I   The "SIZE ERROR" imperative statements will be executed if the year part of the date result is not valid for
                      windowed date receiving field "OLD-EMP-DATE-TERMINATED (NUMERIC INTEGER)".

   563  IGYPA3299-I   The "SIZE ERROR" imperative statements will be executed if the year part of the date result is not valid for
                      windowed date receiving field "OLD-EMP-DATE-MAINTAINED (NUMERIC INTEGER)".
```

Figure 92. Compiler Diagnostic Messages (I) for Internal Bridging Program TARMU3B (1st Pass)

Figure 93 shows the error-level compiler diagnostic messages produced by the compiler. We do need to eliminate these messages by changing the program.

```
LineID  Message code  Message text

  [1]
   235  IGYPA3296-E   Non-date "WORK-EIB-YYDDD (ALPHANUMERIC)" was found as a receiver in an "ACCEPT DATE" or "DAY" statement.
                      Execution results are unpredictable.

  [2]
   330  IGYPA3307-E   Date field "EMP-DATE-JOINED (NUMERIC INTEGER)" was found as a sender, but the receiver was non-date
                      "TARDATE-DATE (ALPHANUMERIC)".  "EMP-DATE-JOINED (NUMERIC INTEGER)" was treated as a non-date in this
                      context.

  [2]
   339  IGYPA3307-E   Date field "EMP-BIRTH-DATE (NUMERIC INTEGER)" was found as a sender, but the receiver was non-date
                      "TARDATE-DATE (ALPHANUMERIC)".  "EMP-BIRTH-DATE (NUMERIC INTEGER)" was treated as a non-date in this
                      context.

  [2]
   349  IGYPA3307-E   Date field "EMP-DATE-TERMINATED (NUMERIC INTEGER)" was found as a sender, but the receiver was non-date
                      "TARDATE-DATE (ALPHANUMERIC)".  "EMP-DATE-TERMINATED (NUMERIC INTEGER)" was treated as a non-date in this
                      context.

  [2]
   361  IGYPA3307-E   Date field "EMP-SECURITY-EXP (PACKED INTEGER)" was found as a sender, but the receiver was non-date
                      "TARDATE-DATE (ALPHANUMERIC)".  "EMP-SECURITY-EXP (PACKED INTEGER)" was treated as a non-date in this
                      context.
```

Figure 93. Compiler Diagnostic Messages (E) for Internal Bridging Program TARMU3B (1st Pass)

We change the program to eliminate these messages. The changed portion of the program is shown on Figure 94 on page 218. The highlighted reference number in the Compiler Diagnostic Messages (E) shown on Figure 93 corresponds to the highlighted reference number in the updated program shown on Figure 94 on page 218. The following notes apply:

1 This error message refers to the statement

```
ACCEPT WORK-EIB-YYDDD FROM  DAY
```

The problem with this is that WORK-EIB-YYDDD is a nondate variable that accepts the date function. To resolve this, we need to add the DATE FORMAT CLAUSE on WORK-EIB-YYDDD (see Figure 93 on page 217).

1a Add date format to other date fields.

2 These error messages refer to moving the windowed dates to TARDATE-DATE which is a nondate variable. The solution to this problem has two parts:

- In the data division section, add the DATE FORMAT CLAUSE on TARDATE-DATE-YYDDD which is a part of the TARDATE-DATE group item.
- In the procedure division section, before calling TARDTE3, move the date fields to TARDATE-DATE-YYDDD which is a date field instead of TARDATE-DATE.

```
       CBL TEST(ALL,SYM) NODYNAM APOST RENT
       CBL DATEPROC(FLAG) YEARWINDOW(1950)
       IDENTIFICATION DIVISION.
       PROGRAM-ID.   TARMU3B.

⋮

       WORKING-STORAGE SECTION.

⋮

       01  WORK-VARS.
           03  WORK-TODAYS-MMDDYY          PIC 9(8).
           03  WORK-MSG-CODE               PIC 99.
           03  WORK-EIB-DATE               PIC 9(7).
           03  WORK-EIB-DATE-RED REDEFINES WORK-EIB-DATE.
               05  FILLER                  PIC X(1).
               05  WORK-EIB-CENTURY        PIC X(1).
      *        05  WORK-EIB-YYDDD          PIC X(5).
               05  WORK-EIB-YYDDD          PIC X(5) DATE FORMAT YYXXX.      1
      *    03  WORK-JOINED-YYDDD           PIC 9(5).
           03  WORK-JOINED-YYDDD           PIC 9(5) DATE FORMAT YYXXX.      1a
           03  WORK-JOINED-MMDDYY.
               05  WORK-JOINED-MMDD        PIC X(6).
      *        05  WORK-JOINED-YY          PIC 99.
               05  WORK-JOINED-YY          PIC 99   DATE FORMAT YY.         1a
           03  WORK-SEC-EXP                PIC 9(5).
      *    03  WORK-TERMINATED-YYDDD       PIC 9(5).
           03  WORK-TERMINATED-YYDDD       PIC 9(5) DATE FORMAT YYXXX.      1a

⋮

       01  TARDATE-PARAMETERS.
           03  TARDATE-DATE                PIC X(8).
           03  TARDATE-DATE-RED1 REDEFINES TARDATE-DATE.
      *        05  TARDATE-DATE-YYDDD      PIC 9(5).
               05  TARDATE-DATE-YYDDD      PIC 9(5) DATE FORMAT YYXXX.      2
               05  FILLER                  PIC X(3).
           03  TARDATE-DATE-RED2 REDEFINES TARDATE-DATE.
      *        05  TARDATE-DATE-YYMMDD     PIC 9(6).
               05  TARDATE-DATE-YYMMDD     PIC 9(6) DATE FORMAT YYXXXX.     2
               05  FILLER                  PIC X(2).
           03  TARDATE-INPUT-FORMAT        PIC X(8).
           03  TARDATE-OUTPUT-FORMAT       PIC X(8).
           03  TARDATE-MESSAGE             PIC X(30).
```

Figure 94 (Part 1 of 2). Using Internal Bridging to Process 2-Digit Year — Updated Version

```
⋮
       PROCEDURE DIVISION.
⋮
           ACCEPT WORK-EIB-YYDDD FROM  DAY.                                  1
       *   MOVE   WORK-EIB-YYDDD TO    TARDATE-DATE.
           MOVE   WORK-EIB-YYDDD TO    TARDATE-DATE-YYDDD.                   2
           MOVE   "YYDDD"        TO    TARDATE-INPUT-FORMAT.
           MOVE   "MM/DD/YY"     TO    TARDATE-OUTPUT-FORMAT.
           CALL   "TARDTE3"      USING TARDATE-DATE
                                       TARDATE-INPUT-FORMAT
                                       TARDATE-OUTPUT-FORMAT
                                       TARDATE-MESSAGE.
           MOVE TARDATE-DATE     TO    WORK-TODAYS-MMDDYY.
⋮
       *       MOVE EMP-DATE-JOINED      TO TARDATE-DATE
               MOVE EMP-DATE-JOINED      TO TARDATE-DATE-YYDDD               2
               MOVE "YYDDD"              TO TARDATE-INPUT-FORMAT
               MOVE "MM/DD/YY"           TO TARDATE-OUTPUT-FORMAT
               CALL "TARDTE3" USING TARDATE-DATE
                                    TARDATE-INPUT-FORMAT
                                    TARDATE-OUTPUT-FORMAT
                                    TARDATE-MESSAGE
               MOVE TARDATE-DATE         TO OUT-EMP-DATE-JOINED
       *
       *       MOVE EMP-BIRTH-DATE       TO TARDATE-DATE
               MOVE EMP-BIRTH-DATE       TO TARDATE-DATE-YYDDD               2
               MOVE "YYDDD"              TO TARDATE-INPUT-FORMAT
               MOVE "MM/DD/YY"           TO TARDATE-OUTPUT-FORMAT
               CALL "TARDTE3" USING TARDATE-DATE
                                    TARDATE-INPUT-FORMAT
                                    TARDATE-OUTPUT-FORMAT
                                    TARDATE-MESSAGE
               MOVE TARDATE-DATE         TO OUT-EMP-BIRTH-DATE
       *
               IF  EMP-DATE-TERMINATED > ZEROS THEN
       *           MOVE EMP-DATE-TERMINATED  TO TARDATE-DATE
                   MOVE EMP-DATE-TERMINATED  TO TARDATE-DATE-YYMMDD            2
                   MOVE "YYMMDD"             TO TARDATE-INPUT-FORMAT
                   MOVE "MM/DD/YY"           TO TARDATE-OUTPUT-FORMAT
                   CALL "TARDTE3" USING TARDATE-DATE
                                        TARDATE-INPUT-FORMAT
                                        TARDATE-OUTPUT-FORMAT
                                        TARDATE-MESSAGE
                   MOVE TARDATE-DATE         TO OUT-EMP-DATE-TERMINATED
               ELSE
                   MOVE SPACES               TO OUT-EMP-DATE-TERMINATED
               END-IF
       *
       *       MOVE EMP-SECURITY-EXP     TO TARDATE-DATE
               MOVE EMP-SECURITY-EXP     TO TARDATE-DATE-YYDDD               2
               MOVE "YYDDD"              TO TARDATE-INPUT-FORMAT
               MOVE "MM/DD/YY"           TO TARDATE-OUTPUT-FORMAT
               CALL "TARDTE3" USING TARDATE-DATE
                                    TARDATE-INPUT-FORMAT
                                    TARDATE-OUTPUT-FORMAT
                                    TARDATE-MESSAGE
               MOVE TARDATE-DATE         TO OUT-EMP-SECURITY-EXP
⋮
```

Figure 94 (Part 2 of 2). Using Internal Bridging to Process 2-Digit Year — Updated Version

After compiling the program for the second time, we are left with only the information-level diagnostic messages, as shown in Figure 95 on page 220

```
LineID  Message code  Message text

    63  IGYDS1454-I   A group item contained one or more subordinate date fields.

                      Same message on line:    175    200

    96  IGYGR1216-I   A "RECORDING MODE" of "F" was assumed for file "INPUT-FILE".

   179  IGYDS1462-I   Group item "WORK-EIB-DATE-RED", which contained one or more date fields, redefined "WORK-EIB-DATE".

   202  IGYDS1462-I   Group item "TARDATE-DATE-RED1", which contained one or more date fields, redefined "TARDATE-DATE".

   205  IGYDS1462-I   Group item "TARDATE-DATE-RED2", which contained one or more date fields, redefined "TARDATE-DATE".

   235  IGYPA3290-I   The statement contained date logic.

                      Same message on line:    236    300    301    302    303    304    330    339    348    349    361    423
                                               429    465    466    488    517    525    535    537    546    548    553    556
                                               557    559    563    566    569

   235  IGYPA3316-I   The "ACCEPT" statement will transfer a windowed date value into "WORK-EIB-YYDDD (ALPHANUMERIC)".

   300  IGYPA3309-I   The windowed date value in "OLD-EMP-DATE-JOINED (NUMERIC INTEGER)" will be expanded and moved into
                      "EMP-DATE-JOINED (NUMERIC INTEGER)".

   301  IGYPA3309-I   The windowed date value in "OLD-EMP-DATE-TERMINATED (NUMERIC INTEGER)" will be expanded and moved into
                      "EMP-DATE-TERMINATED (NUMERIC INTEGER)".

   302  IGYPA3309-I   The windowed date value in "OLD-EMP-DATE-MAINTAINED (NUMERIC INTEGER)" will be expanded and moved into
                      "EMP-DATE-MAINTAINED (NUMERIC INTEGER)".

   303  IGYPA3309-I   The windowed date value in "OLD-EMP-BIRTH-DATE (NUMERIC INTEGER)" will be expanded and moved into
                      "EMP-BIRTH-DATE (NUMERIC INTEGER)".

   304  IGYPA3309-I   The windowed date value in "OLD-EMP-SECURITY-EXP (PACKED INTEGER)" will be expanded and moved into
                      "EMP-SECURITY-EXP (PACKED INTEGER)".

   466  IGYPA3310-I   A windowed date comparison will occur between "WORK-TERMINATED-YYDDD (NUMERIC INTEGER)" and
                      "WORK-JOINED-YYDDD (NUMERIC INTEGER)".

   517  IGYPA3309-I   The windowed date value in "TARDATE-DATE-YYDDD (NUMERIC INTEGER)" will be expanded and moved into
                      "EMP-DATE-JOINED (NUMERIC INTEGER)".

   525  IGYPA3309-I   The windowed date value in "TARDATE-DATE-YYDDD (NUMERIC INTEGER)" will be expanded and moved into
                      "EMP-BIRTH-DATE (NUMERIC INTEGER)".

   535  IGYPA3309-I   The windowed date value in "TARDATE-DATE-YYMMDD (NUMERIC INTEGER)" will be expanded and moved into
                      "EMP-DATE-TERMINATED (NUMERIC INTEGER)".

   546  IGYPA3309-I   The windowed date value in "TARDATE-DATE-YYDDD (NUMERIC INTEGER)" will be expanded and moved into
                      "EMP-SECURITY-EXP (PACKED INTEGER)".

   548  IGYPA3309-I   The windowed date value in "WORK-EIB-YYDDD (ALPHANUMERIC)" will be expanded and moved into
                      "EMP-DATE-MAINTAINED (NUMERIC INTEGER)".

   553  IGYPA3299-I   The "SIZE ERROR" imperative statements will be executed if the year part of the date result is not valid for
                      windowed date receiving field "OLD-EMP-DATE-JOINED (NUMERIC INTEGER)".

   559  IGYPA3299-I   The "SIZE ERROR" imperative statements will be executed if the year part of the date result is not valid for
                      windowed date receiving field "OLD-EMP-DATE-TERMINATED (NUMERIC INTEGER)".

   563  IGYPA3299-I   The "SIZE ERROR" imperative statements will be executed if the year part of the date result is not valid for
                      windowed date receiving field "OLD-EMP-DATE-MAINTAINED (NUMERIC INTEGER)".
```

Figure 95. Compiler Diagnostic Messages (I) for Internal Bridging Program TARMU3B (2nd Pass)

If you are sure that the messages left are solely the information-level messages, then you could change the compiler option to DATEPROC(NOFLAG) for clean management. Using this option, you do not get any messages from the compiler. Again, however, make sure that you have only the information-level messages left.

7.3.5.3 Year 2000 Compliant Application Program with MLE Enhancements

Figure 96 provides an example of the final version of program TARMU3B to reflect the use of the Millennium Language Extensions.

```
       CBL TEST(ALL,SYM) NODYNAM APOST RENT
       CBL DATEPROC(FLAG) YEARWINDOW(1950)
       IDENTIFICATION DIVISION.
       PROGRAM-ID.   TARMU3B.
       AUTHOR.       MERRILL BANI / RINA SUMIANTORO.
       DATE-WRITTEN. JUNE 1997 / APRIL 1998.
      * -------------------------------------------------------- */
      * LICENSED MATERIALS - PROPERTY OF IBM                     */
      *                                                          */
      * (C) COPYRIGHT IBM CORP. 1997                             */
      *                                                          */
      * US GOVERNMENT USERS RESTRICTED RIGHTS - USE,             */
      * DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP          */
      * SCHEDULE CONTRACT WITH IBM CORP.                         */
      * -------------------------------------------------------- */
      *REMARKS.                                                  */
      *                                                          */
      *    VA2000 MVS COBOL TEST APPLICATION                     */
      *                                                          */
      *    ONLINE UPDATE OF EMPLOYEE DATABASE                    */
      *                                                          */
      *    1. Read batch input                                   */
      *                                                          */
      *    2. Verify details                                     */
      *                                                          */
      *    3. If valid, update employee database                 */
      *                                                          */
      * -------------------------------------------------------- */
       EJECT
       ENVIRONMENT DIVISION.

       INPUT-OUTPUT SECTION.

       FILE-CONTROL.

           SELECT EMPLOYEE-MASTER-FILE
                  ASSIGN TO EMPMAST
                  ORGANIZATION IS INDEXED
                  ACCESS IS RANDOM
                  RECORD KEY IS EMP-ID
                  FILE STATUS IS EMP-FILE-STATUS.

           SELECT DEPT-MASTER-FILE
                  ASSIGN TO DEPTMAST
                  ORGANIZATION IS INDEXED
                  ACCESS IS RANDOM
                  RECORD KEY IS DEPT-CODE
                  FILE STATUS IS DEPT-FILE-STATUS.

           SELECT INPUT-FILE
                  ASSIGN TO INPFILE
                  ORGANIZATION IS SEQUENTIAL
                  ACCESS IS SEQUENTIAL
                  FILE STATUS IS INP-FILE-STATUS.

       EJECT

       DATA DIVISION.

       FILE SECTION.

       FD  EMPLOYEE-MASTER-FILE
           RECORD CONTAINS 200 CHARACTERS.
      *
```

Figure 96 (Part 1 of 9). Using Internal Bridging to Process 2-Digit Year — Final Version

```
 01  EMPLOYEE-MASTER-RECORD.
*        ** key field
     03  EMP-ID                      PIC X(6).
     03  EMP-DEPT-CODE               PIC X(4).
     03  EMP-NAME                    PIC X(30).
     03  EMP-ADDR-1                  PIC X(30).
     03  EMP-ADDR-2                  PIC X(30).
     03  EMP-ADDR-3                  PIC X(30).
     03  EMP-ZIP-CODE                PIC X(5).
*        ** format (yyddd)
     03  OLD-EMP-DATE-JOINED         PIC 9(5) DATE FORMAT YYXXX.
*        ** format (yymmdd)
     03  OLD-EMP-DATE-TERMINATED     PIC 9(6) DATE FORMAT YYXXXX.
*        ** format (yyddd)
     03  OLD-EMP-DATE-MAINTAINED     PIC 9(5) DATE FORMAT YYXXX.
*        ** format (yyddd)
     03  OLD-EMP-BIRTH-DATE          PIC 9(5) DATE FORMAT YYXXX.
*        ** format (yyddd)
     03  OLD-EMP-SECURITY-EXP        PIC 9(5) COMP-3
                                             DATE FORMAT YYXXX.
     03  FILLER                      PIC X(41).

 FD  DEPT-MASTER-FILE
     RECORD CONTAINS 80 CHARACTERS.
*
 01  DEPT-MASTER-RECORD.
*        ** key field
     03  DEPT-CODE                   PIC X(4).
     03  DEPT-DESCRIPTION            PIC X(30).
*        ** format (yyddd) packed
     03  DEPT-DATE-MAINTAINED        PIC 9(5) COMP-3.
     03  FILLER                      PIC X(43).

 FD  INPUT-FILE
     RECORD CONTAINS 167 CHARACTERS.
*
 01  INPUT-RECORD.
*        ** key field
     03  INP-EMP-ID                  PIC X(6).
     03  INP-EMP-DEPT-CODE           PIC X(4).
     03  INP-EMP-NAME                PIC X(30).
     03  INP-EMP-ADDR-1              PIC X(30).
     03  INP-EMP-ADDR-2              PIC X(30).
     03  INP-EMP-ADDR-3              PIC X(30).
     03  INP-EMP-ZIP-CODE            PIC X(5).
*        ** format (mm/dd/yy)
     03  INP-EMP-DATE-JOINED         PIC 9(8).
*        ** format (mm/dd/yy)
     03  INP-EMP-DATE-TERMINATED     PIC 9(8).
*        ** format (mm/dd/yy)
     03  INP-EMP-BIRTH-DATE          PIC 9(8).
*        ** format (mm/dd/yy)
     03  INP-EMP-SECURITY-EXP        PIC 9(8).
```

Figure 96 (Part 2 of 9). Using Internal Bridging to Process 2-Digit Year — Final Version

```
       WORKING-STORAGE SECTION.

       01  OUTPUT-RECORD.
      *        ** key field
           03  OUT-EMP-ID                  PIC X(6).
           03  OUT-EMP-DEPT-CODE           PIC X(4).
           03  OUT-EMP-NAME                PIC X(30).
           03  OUT-EMP-ADDR-1              PIC X(30).
           03  OUT-EMP-ADDR-2              PIC X(30).
           03  OUT-EMP-ADDR-3              PIC X(30).
           03  OUT-EMP-ZIP-CODE            PIC X(5).
      *        ** format (mm/dd/yy)
           03  OUT-EMP-DATE-JOINED         PIC X(8).
      *        ** format (mm/dd/yy)
           03  OUT-EMP-DATE-TERMINATED     PIC X(8).
      *        ** format (mm/dd/yy)
           03  OUT-EMP-DATE-MAINTAINED     PIC X(8).
      *        ** format (mm/dd/yy)
           03  OUT-EMP-BIRTH-DATE          PIC X(8).
      *        ** format (mm/dd/yy)
           03  OUT-EMP-SECURITY-EXP        PIC X(8).
           03  FILLER                      PIC X(41).

       01  SWITCHES.
           03  ERROR-SWITCH                PIC X VALUE SPACE.
               88  ERRORS                        VALUE "Y".
       01  ADD-RECORD                      PIC X VALUE SPACES.
       01  CHANGE-RECORD                   PIC X VALUE SPACES.

       01  SWITCH-OFF                      PIC X VALUE "N".

      *    ** message table

       01  MESSAGE-TABLE.
           03  FILLER  PIC X(30) VALUE "I-RECORD ADDED              ".
           03  FILLER  PIC X(30) VALUE "I-RECORD CHANGED            ".
           03  FILLER  PIC X(30) VALUE "E-DEPARTMENT CODE INVALID   ".
           03  FILLER  PIC X(30) VALUE "E-ZIP CODE NOT NUMERIC      ".
           03  FILLER  PIC X(30) VALUE "E-INVALID DATE              ".
           03  FILLER  PIC X(30) VALUE "I-ENTER EMPLOYEE NUMBER     ".
           03  FILLER  PIC X(30) VALUE "I-ENTER EMPLOYEE DETAILS    ".
           03  FILLER  PIC X(30) VALUE "I-ENTER CHANGE DETAILS      ".
           03  FILLER  PIC X(30) VALUE "I-SCR VALID PF10 TO UPDATE  ".
           03  FILLER  PIC X(30) VALUE "E-NAME MISSING              ".
           03  FILLER  PIC X(30) VALUE "E-ADDRESS LINE 1 MISSING    ".
           03  FILLER  PIC X(30) VALUE "E-JOINED > TERMINATED DATE  ".
           03  FILLER  PIC X(30) VALUE "E-DATE MISSING              ".
           03  FILLER  PIC X(30) VALUE "E-EMPLOYEE NO NOT NUMERIC   ".

       01  MSG-TABLE-RED REDEFINES MESSAGE-TABLE.
           03  MSG OCCURS 14 TIMES.
               05  FILLER                  PIC X(30).

      *    ** work variables

       01  WORK-VARS.
           03  WORK-TODAYS-MMDDYY          PIC 9(8).
           03  WORK-MSG-CODE               PIC 99.
           03  WORK-EIB-DATE               PIC 9(7).
           03  WORK-EIB-DATE-RED REDEFINES WORK-EIB-DATE.
               05  FILLER                  PIC X(1).
               05  WORK-EIB-CENTURY        PIC X(1).
               05  WORK-EIB-YYDDD          PIC X(5) DATE FORMAT YYXXX.
           03  WORK-JOINED-YYDDD           PIC 9(5) DATE FORMAT YYXXX.
           03  WORK-JOINED-MMDDYY.
               05  WORK-JOINED-MMDD        PIC X(6).
               05  WORK-JOINED-YY          PIC 99   DATE FORMAT YY.
           03  WORK-SEC-EXP                PIC 9(5).
           03  WORK-TERMINATED-YYDDD       PIC 9(5) DATE FORMAT YYXXX.
```

Figure 96 (Part 3 of 9). Using Internal Bridging to Process 2-Digit Year — Final Version

```
       01  FILE-STATUS-FIELDS.
           03  EMP-FILE-STATUS             PIC XX.
           03  DEPT-FILE-STATUS            PIC XX.
           03  INP-FILE-STATUS             PIC XX.

       01  EMP-LENGTH                      PIC S9(4) COMP VALUE +200.
       01  DEP-LENGTH                      PIC S9(4) COMP VALUE +80.
       01  RESPONSE                        PIC S9(8) COMP VALUE +0.
       01  LAST-EMP-ID                     PIC X(6)  VALUE SPACES.

       01  TARDATE-PARAMETERS.
           03  TARDATE-DATE                PIC X(8).
           03  TARDATE-DATE-RED1 REDEFINES TARDATE-DATE.
               05  TARDATE-DATE-YYDDD      PIC 9(5) DATE FORMAT YYXXX.
               05  FILLER                  PIC X(3).
           03  TARDATE-DATE-RED2 REDEFINES TARDATE-DATE.
               05  TARDATE-DATE-YYMMDD     PIC 9(6) DATE FORMAT YYXXXX.
               05  FILLER                  PIC X(2).
           03  TARDATE-INPUT-FORMAT        PIC X(8).
           03  TARDATE-OUTPUT-FORMAT       PIC X(8).
           03  TARDATE-MESSAGE             PIC X(30).

      *    ** expanded date variables

      *    ** format (yyyyddd)  expanded
        01  EMP-DATE-JOINED           PIC 9(7) DATE FORMAT YYYYXXX.
      *    ** format (yyyymmdd) expanded
        01  EMP-DATE-TERMINATED       PIC 9(8) DATE FORMAT YYYYXXXX.
      *    ** format (yyyyddd)  expanded
        01  EMP-DATE-MAINTAINED       PIC 9(7) DATE FORMAT YYYYXXX.
      *    ** format (yyyyddd)  expanded
        01  EMP-BIRTH-DATE            PIC 9(7) DATE FORMAT YYYYXXX.
      *    ** format (yyyyddd)  expanded
        01  EMP-SECURITY-EXP          PIC 9(7) COMP-3
                                              DATE FORMAT YYYYXXX.
           EJECT
        PROCEDURE DIVISION.

           OPEN I-O   EMPLOYEE-MASTER-FILE
           OPEN INPUT DEPT-MASTER-FILE
           OPEN INPUT INPUT-FILE

      *    **  retrieve todays date

           ACCEPT WORK-EIB-YYDDD FROM  DAY.
           MOVE   WORK-EIB-YYDDD TO    TARDATE-DATE-YYDDD.
           MOVE   "YYDDD"        TO    TARDATE-INPUT-FORMAT.
           MOVE   "MM/DD/YY"     TO    TARDATE-OUTPUT-FORMAT.
           CALL   "TARDTE3"      USING TARDATE-DATE
                                       TARDATE-INPUT-FORMAT
                                       TARDATE-OUTPUT-FORMAT
                                       TARDATE-MESSAGE.
           MOVE TARDATE-DATE     TO    WORK-TODAYS-MMDDYY.

           PERFORM 200-READ-INPUT-FILE THRU 200-EXIT.

           STOP RUN.

      *================================================================

        200-READ-INPUT-FILE.

           READ INPUT-FILE
               AT END
               GO TO 200-EXIT
           END-READ.

      *    **  validate employee number
```

Figure 96 (Part 4 of 9). Using Internal Bridging to Process 2-Digit Year — Final Version

```
           IF INP-EMP-ID = ZERO THEN
               MOVE 6              TO  WORK-MSG-CODE
               DISPLAY MSG(WORK-MSG-CODE)
               GO TO 200-READ-NEXT-INPUT-FILE
           ELSE
              IF INP-EMP-ID IS NOT NUMERIC THEN
                 MOVE 14             TO  WORK-MSG-CODE
                 DISPLAY MSG(WORK-MSG-CODE)
                 GO TO 200-READ-NEXT-INPUT-FILE
              END-IF
           END-IF.

           PERFORM 400-READ-EMPLOYEE-MASTER-FILE THRU 400-EXIT
           PERFORM 500-VALIDATE-INPUT-DATA THRU 500-EXIT.
           IF NOT ERRORS
              PERFORM 600-UPDATE-FILE THRU 600-EXIT
           END-IF.

       200-READ-NEXT-INPUT-FILE.

           DISPLAY " "
           GO TO 200-READ-INPUT-FILE.

       200-EXIT.
           EXIT.

      *================================================================

       400-READ-EMPLOYEE-MASTER-FILE.

           MOVE SPACES TO CHANGE-RECORD ADD-RECORD.
           MOVE SPACES TO EMPLOYEE-MASTER-RECORD.

           MOVE INP-EMP-ID TO EMP-ID
           READ EMPLOYEE-MASTER-FILE
                KEY IS EMP-ID.

      *    **  MOVE THE OLD NON-EXPANDED DATE VARIABLES TO THE NEW
      *    **  EXPANDED DATE VARIABLES AFTER READING THE FILE.

           MOVE OLD-EMP-DATE-JOINED      TO EMP-DATE-JOINED.
           MOVE OLD-EMP-DATE-TERMINATED  TO EMP-DATE-TERMINATED.
           MOVE OLD-EMP-DATE-MAINTAINED  TO EMP-DATE-MAINTAINED.
           MOVE OLD-EMP-BIRTH-DATE       TO EMP-BIRTH-DATE.
           MOVE OLD-EMP-SECURITY-EXP     TO EMP-SECURITY-EXP.

           MOVE LOW-VALUES TO OUTPUT-RECORD.
           MOVE EMP-ID     TO OUT-EMP-ID.

           IF EMP-FILE-STATUS = "23" THEN
               MOVE "Y"                  TO ADD-RECORD
               MOVE SPACES               TO OUT-EMP-DEPT-CODE
                                            OUT-EMP-NAME
                                            OUT-EMP-ADDR-1
                                            OUT-EMP-ADDR-2
                                            OUT-EMP-ADDR-3
                                            OUT-EMP-ZIP-CODE
                                            OUT-EMP-DATE-JOINED
                                            OUT-EMP-BIRTH-DATE
                                            OUT-EMP-DATE-TERMINATED
                                            OUT-EMP-SECURITY-EXP
           ELSE
               MOVE "Y"                  TO CHANGE-RECORD
               MOVE EMP-DEPT-CODE        TO OUT-EMP-DEPT-CODE
               MOVE EMP-NAME             TO OUT-EMP-NAME
               MOVE EMP-ADDR-1           TO OUT-EMP-ADDR-1
               MOVE EMP-ADDR-2           TO OUT-EMP-ADDR-2
               MOVE EMP-ADDR-3           TO OUT-EMP-ADDR-3
               MOVE EMP-ZIP-CODE         TO OUT-EMP-ZIP-CODE
      *
               MOVE EMP-DATE-JOINED      TO TARDATE-DATE-YYDDD
               MOVE "YYDDD"              TO TARDATE-INPUT-FORMAT
               MOVE "MM/DD/YY"           TO TARDATE-OUTPUT-FORMAT
```

Figure 96 (Part 5 of 9). Using Internal Bridging to Process 2-Digit Year — Final Version

```
           CALL "TARDTE3" USING TARDATE-DATE
                                TARDATE-INPUT-FORMAT
                                TARDATE-OUTPUT-FORMAT
                                TARDATE-MESSAGE
           MOVE TARDATE-DATE         TO OUT-EMP-DATE-JOINED
      *
           MOVE EMP-BIRTH-DATE       TO TARDATE-DATE-YYDDD
           MOVE "YYDDD"              TO TARDATE-INPUT-FORMAT
           MOVE "MM/DD/YY"           TO TARDATE-OUTPUT-FORMAT
           CALL "TARDTE3" USING TARDATE-DATE
                                TARDATE-INPUT-FORMAT
                                TARDATE-OUTPUT-FORMAT
                                TARDATE-MESSAGE
           MOVE TARDATE-DATE         TO OUT-EMP-BIRTH-DATE
      *
           IF  EMP-DATE-TERMINATED > ZEROS THEN
               MOVE EMP-DATE-TERMINATED  TO TARDATE-DATE-YYMMDD
               MOVE "YYMMDD"             TO TARDATE-INPUT-FORMAT
               MOVE "MM/DD/YY"           TO TARDATE-OUTPUT-FORMAT
               CALL "TARDTE3" USING TARDATE-DATE
                                    TARDATE-INPUT-FORMAT
                                    TARDATE-OUTPUT-FORMAT
                                    TARDATE-MESSAGE
               MOVE TARDATE-DATE         TO OUT-EMP-DATE-TERMINATED
           ELSE
               MOVE SPACES               TO OUT-EMP-DATE-TERMINATED
           END-IF
      *
           MOVE EMP-SECURITY-EXP     TO TARDATE-DATE-YYDDD
           MOVE "YYDDD"              TO TARDATE-INPUT-FORMAT
           MOVE "MM/DD/YY"           TO TARDATE-OUTPUT-FORMAT
           CALL "TARDTE3" USING TARDATE-DATE
                                TARDATE-INPUT-FORMAT
                                TARDATE-OUTPUT-FORMAT
                                TARDATE-MESSAGE
           MOVE TARDATE-DATE         TO OUT-EMP-SECURITY-EXP
      *
       END-IF.

       400-EXIT.
           EXIT.

      *================================================================

       500-VALIDATE-INPUT-DATA.

           MOVE SWITCH-OFF  TO  ERROR-SWITCH.

      *    **    verify dept code

           MOVE INP-EMP-DEPT-CODE TO DEPT-CODE
           READ DEPT-MASTER-FILE
                KEY IS DEPT-CODE
                IF DEPT-FILE-STATUS = "23" THEN
                   MOVE 3 TO WORK-MSG-CODE
                   DISPLAY MSG(WORK-MSG-CODE)
                   SET ERRORS TO TRUE
                END-IF

           IF  INP-EMP-NAME IS NOT > SPACES   THEN
               MOVE 10        TO WORK-MSG-CODE
               DISPLAY MSG(WORK-MSG-CODE)
               SET ERRORS TO TRUE
           END-IF

           IF  INP-EMP-ADDR-1 IS NOT > SPACES   THEN
               MOVE 11        TO WORK-MSG-CODE
               DISPLAY MSG(WORK-MSG-CODE)
               SET ERRORS TO TRUE
           END-IF
```

Figure 96 (Part 6 of 9). Using Internal Bridging to Process 2-Digit Year — Final Version

```
IF  INP-EMP-ZIP-CODE IS NOT NUMERIC   THEN
    MOVE 4         TO WORK-MSG-CODE
    DISPLAY MSG(WORK-MSG-CODE)
    SET ERRORS TO TRUE
END-IF

IF  INP-EMP-DATE-JOINED > SPACES THEN
    MOVE INP-EMP-DATE-JOINED   TO TARDATE-DATE
    MOVE "MM/DD/YY"      TO TARDATE-INPUT-FORMAT
    MOVE "YYDDD"         TO TARDATE-OUTPUT-FORMAT
    CALL "TARDTE3" USING TARDATE-DATE
                         TARDATE-INPUT-FORMAT
                         TARDATE-OUTPUT-FORMAT
                         TARDATE-MESSAGE
    IF  TARDATE-MESSAGE NOT = SPACES THEN
        MOVE 5     TO WORK-MSG-CODE
        DISPLAY MSG(WORK-MSG-CODE)
        SET ERRORS TO TRUE
    ELSE
        MOVE TARDATE-DATE-YYDDD TO WORK-JOINED-YYDDD
        MOVE INP-EMP-DATE-JOINED TO WORK-JOINED-MMDDYY
    END-IF
ELSE
    MOVE WORK-TODAYS-MMDDYY TO OUT-EMP-DATE-JOINED
                               WORK-JOINED-MMDDYY
    MOVE WORK-EIB-YYDDD     TO WORK-JOINED-YYDDD
END-IF

IF  INP-EMP-BIRTH-DATE > SPACES     THEN
    MOVE INP-EMP-BIRTH-DATE   TO TARDATE-DATE
    MOVE "MM/DD/YY"      TO TARDATE-INPUT-FORMAT
    MOVE "YYDDD"         TO TARDATE-OUTPUT-FORMAT
    CALL "TARDTE3" USING TARDATE-DATE
                         TARDATE-INPUT-FORMAT
                         TARDATE-OUTPUT-FORMAT
                         TARDATE-MESSAGE
    IF  TARDATE-MESSAGE NOT = SPACES THEN
        MOVE 5     TO WORK-MSG-CODE
        DISPLAY MSG(WORK-MSG-CODE)
        SET ERRORS TO TRUE
    END-IF
ELSE
    MOVE 13    TO WORK-MSG-CODE
    DISPLAY MSG(WORK-MSG-CODE)
    SET ERRORS TO TRUE
END-IF

IF  INP-EMP-DATE-TERMINATED > SPACES THEN
    MOVE INP-EMP-DATE-TERMINATED TO TARDATE-DATE
    MOVE "MM/DD/YY"      TO TARDATE-INPUT-FORMAT
    MOVE "YYDDD"         TO TARDATE-OUTPUT-FORMAT
    CALL "TARDTE3" USING TARDATE-DATE
                         TARDATE-INPUT-FORMAT
                         TARDATE-OUTPUT-FORMAT
                         TARDATE-MESSAGE
    IF  TARDATE-MESSAGE NOT = SPACES THEN
        MOVE 5     TO WORK-MSG-CODE
        DISPLAY MSG(WORK-MSG-CODE)
        SET ERRORS TO TRUE
    ELSE
        MOVE OUT-EMP-DATE-TERMINATED TO OUT-EMP-SECURITY-EXP
        MOVE TARDATE-DATE-YYDDD TO WORK-TERMINATED-YYDDD
        IF  WORK-TERMINATED-YYDDD < WORK-JOINED-YYDDD
            MOVE 12    TO WORK-MSG-CODE
            DISPLAY MSG(WORK-MSG-CODE)
            SET ERRORS TO TRUE
        END-IF
    END-IF
END-IF.
```

Figure 96 (Part 7 of 9). Using Internal Bridging to Process 2-Digit Year — Final Version

```
           IF  INP-EMP-SECURITY-EXP > SPACES     THEN
               MOVE INP-EMP-SECURITY-EXP TO TARDATE-DATE
               MOVE "MM/DD/YY"      TO TARDATE-INPUT-FORMAT
               MOVE "YYDDD"         TO TARDATE-OUTPUT-FORMAT
               CALL "TARDTE3" USING TARDATE-DATE
                                    TARDATE-INPUT-FORMAT
                                    TARDATE-OUTPUT-FORMAT
                                    TARDATE-MESSAGE
               IF  TARDATE-MESSAGE NOT = SPACES THEN
                   MOVE 5     TO WORK-MSG-CODE
                   DISPLAY MSG(WORK-MSG-CODE)
                   SET ERRORS TO TRUE
               END-IF
           ELSE
               COMPUTE WORK-JOINED-YY = WORK-JOINED-YY + 1
               IF  WORK-JOINED-MMDD = "02/29/" THEN
                   MOVE "02/28/"  TO WORK-JOINED-MMDD
               END-IF
               MOVE WORK-JOINED-MMDDYY TO OUT-EMP-SECURITY-EXP
           END-IF.

       500-EXIT.
           EXIT.

      *================================================================

       600-UPDATE-FILE.

           MOVE SPACES             TO  EMPLOYEE-MASTER-RECORD.
           MOVE INP-EMP-ID         TO  EMP-ID.
           MOVE INP-EMP-DEPT-CODE  TO  EMP-DEPT-CODE.
           MOVE INP-EMP-NAME       TO  EMP-NAME.
           MOVE INP-EMP-ADDR-1     TO  EMP-ADDR-1.
           MOVE INP-EMP-ADDR-2     TO  EMP-ADDR-2.
           MOVE INP-EMP-ADDR-3     TO  EMP-ADDR-3.
           MOVE INP-EMP-ZIP-CODE   TO  EMP-ZIP-CODE.
           MOVE INP-EMP-DATE-JOINED TO  TARDATE-DATE
           MOVE "MM/DD/YY"         TO  TARDATE-INPUT-FORMAT
           MOVE "YYDDD"            TO  TARDATE-OUTPUT-FORMAT
           CALL "TARDTE3" USING        TARDATE-DATE
                                       TARDATE-INPUT-FORMAT
                                       TARDATE-OUTPUT-FORMAT
                                       TARDATE-MESSAGE.
           MOVE TARDATE-DATE-YYDDD TO  EMP-DATE-JOINED.
           MOVE INP-EMP-BIRTH-DATE TO  TARDATE-DATE
           MOVE "MM/DD/YY"         TO  TARDATE-INPUT-FORMAT
           MOVE "YYDDD"            TO  TARDATE-OUTPUT-FORMAT
           CALL "TARDTE3" USING        TARDATE-DATE
                                       TARDATE-INPUT-FORMAT
                                       TARDATE-OUTPUT-FORMAT
                                       TARDATE-MESSAGE.
           MOVE TARDATE-DATE-YYDDD TO  EMP-BIRTH-DATE.

           IF  INP-EMP-DATE-TERMINATED > SPACES THEN
               MOVE INP-EMP-DATE-TERMINATED TO TARDATE-DATE
               MOVE "MM/DD/YY"      TO TARDATE-INPUT-FORMAT
               MOVE "YYMMDD"        TO TARDATE-OUTPUT-FORMAT
               CALL "TARDTE3" USING    TARDATE-DATE
                                       TARDATE-INPUT-FORMAT
                                       TARDATE-OUTPUT-FORMAT
                                       TARDATE-MESSAGE
               MOVE TARDATE-DATE-YYMMDD TO  EMP-DATE-TERMINATED
           ELSE
               MOVE ZEROS           TO  EMP-DATE-TERMINATED
           END-IF.
```

Figure 96 (Part 8 of 9). Using Internal Bridging to Process 2-Digit Year — Final Version

```
           MOVE INP-EMP-SECURITY-EXP TO  TARDATE-DATE
           MOVE "MM/DD/YY"         TO  TARDATE-INPUT-FORMAT
           MOVE "YYDDD"            TO  TARDATE-OUTPUT-FORMAT
           CALL "TARDTE3" USING        TARDATE-DATE
                                       TARDATE-INPUT-FORMAT
                                       TARDATE-OUTPUT-FORMAT
                                       TARDATE-MESSAGE.
           MOVE TARDATE-DATE-YYDDD TO  EMP-SECURITY-EXP.

           MOVE WORK-EIB-YYDDD      TO  EMP-DATE-MAINTAINED.

       *    **  MOVE THE NEW EXPANDED DATE VARIABLES BACK TO THE OLD
       *    **  NON-EXPANDED DATE VARIABLES BEFORE UPDATING THE FILE.

           COMPUTE OLD-EMP-DATE-JOINED      = EMP-DATE-JOINED
             ON SIZE ERROR GO TO 600-OUT-OF-WINDOW-ERROR.

           IF FUNCTION UNDATE(EMP-DATE-TERMINATED) = 00000000
             MOVE EMP-DATE-TERMINATED TO OLD-EMP-DATE-TERMINATED
           ELSE
             COMPUTE OLD-EMP-DATE-TERMINATED  = EMP-DATE-TERMINATED
               ON SIZE ERROR GO TO 600-OUT-OF-WINDOW-ERROR
           END-IF.

           COMPUTE OLD-EMP-DATE-MAINTAINED  = EMP-DATE-MAINTAINED
             ON SIZE ERROR GO TO 600-OUT-OF-WINDOW-ERROR.

           COMPUTE OLD-EMP-BIRTH-DATE       = EMP-BIRTH-DATE
             ON SIZE ERROR GO TO 600-OUT-OF-WINDOW-ERROR.

           COMPUTE OLD-EMP-SECURITY-EXP     = EMP-SECURITY-EXP
             ON SIZE ERROR GO TO 600-OUT-OF-WINDOW-ERROR.

           IF  ADD-RECORD = "Y"  THEN
               WRITE EMPLOYEE-MASTER-RECORD
               MOVE 1    TO  WORK-MSG-CODE
               DISPLAY MSG(WORK-MSG-CODE)
               MOVE SPACES TO ADD-RECORD
               GO TO 600-EXIT
           ELSE
               REWRITE EMPLOYEE-MASTER-RECORD
               MOVE 2    TO  WORK-MSG-CODE
               DISPLAY MSG(WORK-MSG-CODE)
               MOVE SPACES TO CHANGE-RECORD
               GO TO 600-EXIT
           END-IF.

       600-OUT-OF-WINDOW-ERROR.
           DISPLAY 'OUT OF WINDOW CONDITION HAS OCCURRED'.

       600-EXIT.
           EXIT.
```

Figure 96 (Part 9 of 9). Using Internal Bridging to Process 2-Digit Year — Final Version

7.4 File and Database Conversion

No matter what short- or long-term approach you may take to making your application programs ready for the Year 2000, the expansion of 2-digit dates to 4-digit dates at some point of time is inevitable. This expansion, moreover, involves not just 2-digit date fields used in programs but the contents of your files and databases as well. This section discusses how you can use the Millennium Language Extensions to the COBOL compiler to assist you in writing one-time conversion programs to expand your files and databases so that all date fields have a 4-digit year.

Any such utility program written for the purpose of file or database conversion will have the same tasks to perform:

- Read in the old data.

- Expand 2-digit dates to 4-digit dates.
- Write the data back to an expanded version of the original file or database.

To ensure that the expanded output dates resulting from the conversion process are correct, each program will have to make consistent use of a century window as a means of interpreting 2-digit dates. Although other options such as the use of COBOL intrinsic date expansion functions, Language Environment callable services or even the manual coding of windowing logic could conceivably be used to perform date expansion in these utility programs, the MLE enhancements can be used to automatically furnish the century portion of the date in the expanded target field based upon the specified century window.

Figure 97 provides an example of such a special-purpose program, which copies data from one file to another while expanding the 2-digit date fields to 4-digit date fields. It reads in the VSAM files from our sample application containing employee and departmental data and writes the data to newly created copies of the original file, which have been expanded to hold 4-digit year data. These files will be used subsequently as input to versions of our sample application programs which have been modified to act on the new 4-digit year dates. This is known as the field expansion approach to the Year 2000 problem and is described in 7.5, “Full Field Expansion” on page 236.

```
       CBL TEST(ALL,SYM) NODYNAM APOST RENT
       CBL DATEPROC(FLAG) YEARWINDOW(1900)                              1
       CBL LIB
       IDENTIFICATION DIVISION.
       PROGRAM-ID.   TARMUCON.
       AUTHOR.       ERIC PRIVETTE
       DATE-WRITTEN. APRIL 1998.
      * -------------------------------------------------------- */
      * LICENSED MATERIALS - PROPERTY OF IBM                     */
      *                                                          */
      * (C) COPYRIGHT IBM CORP. 1997                             */
      *                                                          */
      * US GOVERNMENT USERS RESTRICTED RIGHTS - USE,             */
      * DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP          */
      * SCHEDULE CONTRACT WITH IBM CORP.                         */
      * -------------------------------------------------------- */
      *REMARKS.                                                  */
      *                                                          */
      *    FIND AND FIX IN OS/390 FILE CONVERSION APPLICATION    */
      *    ILLUSTRATING ASSORTMENT OF COBOL FEATURES             */
      *    IN FULL FIELD DATE EXPANSION WHEN                     */
      *    CONVERTING FILES:                                     */
      *                                                          */
      *    - MLE WINDOWED AND EXPANDED DATE FORMATS              */
      *                                                          */
      *    1. READ INPUT FILE REQUIRING CONVERSION               */
      *                                                          */
      *    2. EXPAND DATES                                       */
      *                                                          */
      *    3. WRITE TO NEW FILE                                  */
      *                                                          */
      * -------------------------------------------------------- */
       EJECT
```

Figure 97 (Part 1 of 5). Sample File and Database Conversion Program

```
       ENVIRONMENT DIVISION.

       INPUT-OUTPUT SECTION.

       FILE-CONTROL.

           SELECT EMPLOYEE-MASTER-FILE-OLD
                  ASSIGN TO EMPMSTO
                  ORGANIZATION IS INDEXED
                  ACCESS IS SEQUENTIAL
                  RECORD KEY IS EMP-ID OF EMPLOYEE-MASTER-RECORD-OLD
                  FILE STATUS IS EMP-FILE-STATUS-OLD.

           SELECT DEPT-MASTER-FILE-OLD
                  ASSIGN TO DEPMSTO
                  ORGANIZATION IS INDEXED
                  ACCESS IS SEQUENTIAL
                  RECORD KEY IS DEPT-CODE OF DEPT-MASTER-RECORD-OLD
                  FILE STATUS IS DEPT-FILE-STATUS-OLD.

           SELECT EMPLOYEE-MASTER-FILE-NEW
                  ASSIGN TO EMPMSTN
                  ORGANIZATION IS INDEXED
                  ACCESS IS RANDOM
                  RECORD KEY IS EMP-ID OF EMPLOYEE-MASTER-RECORD-NEW
                  FILE STATUS IS EMP-FILE-STATUS-NEW.

           SELECT DEPT-MASTER-FILE-NEW
                  ASSIGN TO DEPMSTN
                  ORGANIZATION IS INDEXED
                  ACCESS IS RANDOM
                  RECORD KEY IS DEPT-CODE OF DEPT-MASTER-RECORD-NEW
                  FILE STATUS IS DEPT-FILE-STATUS-NEW.


       EJECT

       DATA DIVISION.

       FILE SECTION.

       FD  EMPLOYEE-MASTER-FILE-OLD
           RECORD CONTAINS 200 CHARACTERS.
      *
       01  EMPLOYEE-MASTER-RECORD-OLD.
           03  EMP-ID                      PIC X(6).
           03  EMP-DEPT-CODE               PIC X(4).
           03  EMP-NAME                    PIC X(30).
           03  EMP-ADDR-1                  PIC X(30).
           03  EMP-ADDR-2                  PIC X(30).
           03  EMP-ADDR-3                  PIC X(30).
           03  EMP-ZIP-CODE                PIC X(5).
      *        ** FORMAT (YYDDD)
           03  EMP-DATE-JOINED             PIC 9(5) DATE FORMAT YYXXX.  2
      *        ** FORMAT (YYMMDD)
           03  EMP-DATE-TERMINATED         PIC 9(6) DATE FORMAT YYXXXX. 2
      *        ** FORMAT (YYDDD)
           03  EMP-DATE-MAINTAINED         PIC 9(5) DATE FORMAT YYXXX.  2
      *        ** FORMAT (YYDDD)
           03  EMP-BIRTH-DATE              PIC 9(5) DATE FORMAT YYXXX.  2
      *        ** FORMAT (YYDDD)
           03  EMP-SECURITY-EXP            PIC 9(5) COMP-3
                                                    DATE FORMAT YYXXX.  2
           03  REST                        PIC X(41).
```

Figure 97 (Part 2 of 5). Sample File and Database Conversion Program

```
 FD  EMPLOYEE-MASTER-FILE-NEW
     RECORD CONTAINS 200 CHARACTERS.
*
 01  EMPLOYEE-MASTER-RECORD-NEW.
     03  EMP-ID                      PIC X(6).
     03  EMP-DEPT-CODE               PIC X(4).
     03  EMP-NAME                    PIC X(30).
     03  EMP-ADDR-1                  PIC X(30).
     03  EMP-ADDR-2                  PIC X(30).
     03  EMP-ADDR-3                  PIC X(30).
     03  EMP-ZIP-CODE                PIC X(5).
*        ** FORMAT (YYYYDDD)
     03  EMP-DATE-JOINED             PIC 9(7)
                                     DATE FORMAT YYYYXXX.          3
*        ** FORMAT (YYYYMMDD)
     03  EMP-DATE-TERMINATED         PIC 9(8)
                                     DATE FORMAT YYYYXXXX.         3
*        ** FORMAT (YYYYDDD)
     03  EMP-DATE-MAINTAINED         PIC 9(7)
                                     DATE FORMAT YYYYXXX.          3
*        ** FORMAT (YYYYDDD)
     03  EMP-BIRTH-DATE              PIC 9(7)
                                     DATE FORMAT YYYYXXX.          3
*        ** FORMAT (YYYYDDD)
     03  EMP-SECURITY-EXP            PIC 9(7) COMP-3
                                     DATE FORMAT YYYYXXX.          3
     03  REST                        PIC X(32).                    4

 FD  DEPT-MASTER-FILE-OLD
     RECORD CONTAINS 80 CHARACTERS.
*
 01  DEPT-MASTER-RECORD-OLD.
     03  DEPT-CODE                   PIC X(4).
     03  DEPT-DESCRIPTION            PIC X(30).
*        ** FORMAT (YYDDD) PACKED
     03  DEPT-DATE-MAINTAINED        PIC 9(5) COMP-3
                                     DATE FORMAT YYXXX.            2
     03  REST                        PIC X(43).

 FD  DEPT-MASTER-FILE-NEW
     RECORD CONTAINS 80 CHARACTERS.
*
 01  DEPT-MASTER-RECORD-NEW.
     03  DEPT-CODE                   PIC X(4).
     03  DEPT-DESCRIPTION            PIC X(30).
*        ** FORMAT (YYDDD) PACKED
     03  DEPT-DATE-MAINTAINED        PIC 9(7) COMP-3
                                     DATE FORMAT YYYYXXX.          3
     03  REST                        PIC X(42).                    4

 WORKING-STORAGE SECTION.

 01  FILE-STATUS-FIELDS.
     03  EMP-FILE-STATUS-OLD         PIC 99.
     03  DEPT-FILE-STATUS-OLD        PIC 99.
     03  EMP-FILE-STATUS-NEW         PIC 99.
     03  DEPT-FILE-STATUS-NEW        PIC 99.

 01  SWITCHES.
     03  VSAM-ERROR-SWITCH           PIC X VALUE SPACE.
         88  VSAM-ERROR                    VALUE "Y".
     03  EOF1-SWITCHES               PIC X VALUE SPACE.
         88  EOF1                          VALUE "Y".
     03  EOF2-SWITCHES               PIC X VALUE SPACE.
         88  EOF2                          VALUE "Y".
     03  ERROR-SWITCH                PIC X VALUE SPACE.
         88  ERRORS                        VALUE "Y".

 01  VSAM-ERROR-VARS.
     03  VSAM-ERROR-ACTION           PIC X(8).
     03  VSAM-ERROR-FILE             PIC X(10).
     03  VSAM-ERROR-STATUS           PIC X(2).

     EJECT
```

Figure 97 (Part 3 of 5). Sample File and Database Conversion Program

```
       PROCEDURE DIVISION.

           PERFORM 100-INITIALIZE THRU 100-EXIT.

           IF  VSAM-ERROR   THEN
               NEXT SENTENCE
           ELSE
               PERFORM 200-EMPL-PROCESS THRU 200-EXIT UNTIL EOF1 OR
                                                            VSAM-ERROR.

           PERFORM 300-INITIALIZE THRU 300-EXIT.

           IF  VSAM-ERROR   THEN
               NEXT SENTENCE
           ELSE
               PERFORM 400-DEPT-PROCESS THRU 400-EXIT UNTIL EOF2 OR
                                                            VSAM-ERROR.

           PERFORM CLOSE-FILES.

           STOP RUN.
      *===============================================================
           SKIP3
       100-INITIALIZE.

      *    **  OPEN FILES

           MOVE "OPEN FILE" TO VSAM-ERROR-ACTION.

           OPEN INPUT EMPLOYEE-MASTER-FILE-OLD.
           IF  EMP-FILE-STATUS-OLD = "00" THEN
               NEXT SENTENCE
           ELSE
               MOVE "EMPMSTO"       TO VSAM-ERROR-FILE
               MOVE EMP-FILE-STATUS-OLD TO VSAM-ERROR-STATUS
               PERFORM 999-VSAM-ERROR THRU 999-EXIT
           END-IF.

           OPEN OUTPUT EMPLOYEE-MASTER-FILE-NEW.
           IF  EMP-FILE-STATUS-NEW = "00" THEN
               NEXT SENTENCE
           ELSE
               MOVE "EMPMSTN"       TO VSAM-ERROR-FILE
               MOVE EMP-FILE-STATUS-NEW TO VSAM-ERROR-STATUS
               PERFORM 999-VSAM-ERROR THRU 999-EXIT
           END-IF.


       100-EXIT.
           EXIT.

      * ============================================================ *
           SKIP3
       200-EMPL-PROCESS.

           READ EMPLOYEE-MASTER-FILE-OLD
               AT END SET EOF1 TO TRUE
               GO TO 200-EXIT
           END-READ.

           MOVE CORRESPONDING EMPLOYEE-MASTER-RECORD-OLD TO                5
                              EMPLOYEE-MASTER-RECORD-NEW.

           WRITE EMPLOYEE-MASTER-RECORD-NEW.

       200-EXIT.
           EXIT.
      * ============================================================ *
           SKIP3
```

Figure 97 (Part 4 of 5). Sample File and Database Conversion Program

```
 300-INITIALIZE.

*    **  OPEN FILES

     OPEN INPUT DEPT-MASTER-FILE-OLD.
     IF  DEPT-FILE-STATUS-OLD = "00" THEN
         NEXT SENTENCE
     ELSE
         MOVE "DEPMSTO"       TO VSAM-ERROR-FILE
         MOVE DEPT-FILE-STATUS-OLD TO VSAM-ERROR-STATUS
         PERFORM 999-VSAM-ERROR THRU 999-EXIT
     END-IF.

     OPEN OUTPUT DEPT-MASTER-FILE-NEW.
     IF  DEPT-FILE-STATUS-NEW = "00" THEN
         NEXT SENTENCE
     ELSE
         MOVE "DEPMSTN"       TO VSAM-ERROR-FILE
         MOVE DEPT-FILE-STATUS-NEW TO VSAM-ERROR-STATUS
         PERFORM 999-VSAM-ERROR THRU 999-EXIT
     END-IF.

 300-EXIT.
     EXIT.
* ============================================================ *
     SKIP3
 400-DEPT-PROCESS.

     READ DEPT-MASTER-FILE-OLD
         AT END SET EOF2 TO TRUE
         GO TO 400-EXIT
     END-READ.

     MOVE CORRESPONDING DEPT-MASTER-RECORD-OLD TO                    5
                        DEPT-MASTER-RECORD-NEW.

     WRITE DEPT-MASTER-RECORD-NEW.

 400-EXIT.
     EXIT.
* ============================================================ *
 999-VSAM-ERROR.

     SET VSAM-ERROR  TO  TRUE.

     DISPLAY VSAM-ERROR-ACTION
             VSAM-ERROR-FILE
             "STATUS ERROR "
             VSAM-ERROR-STATUS.

 999-EXIT.
     EXIT.
* ============================================================ *
 CLOSE-FILES.
     CLOSE EMPLOYEE-MASTER-FILE-OLD.
     CLOSE EMPLOYEE-MASTER-FILE-NEW.
     CLOSE DEPT-MASTER-FILE-NEW.
     CLOSE DEPT-MASTER-FILE-OLD.
     EXIT.
```

Figure 97 (Part 5 of 5). Sample File and Database Conversion Program

1 The results of date expansion depend upon how the century window is specified. The value of 1900 is used here to indicate the starting year of a fixed 100-year interval because all of the dates being expanded fall within the twentieth century. You should be aware of the range represented by the data being converted when using the MLE enhancements to perform date expansion. The use of other windowing alternatives such as COBOL intrinsic functions or Language Environment callable services may be required if dates in the file fall outside the 100-year interval defined by the YEARWINDOW compiler option. Use of the windowing date expansion intrinsic function DATE-TO-YYMMDD, for example, allows you to specify a century window whose first year falls outside the Twentieth Century, as Figure 98 on page 235 illustrates.

```
WORKING-STORAGE SECTION.

01 2-DIGIT-YEAR-DATE    PIC 9(6) VALUE 981002.
01 EXPANDED-DATE        PIC 9(8) DATE FORMAT YYYYXXXX.
PROCEDURE DIVISION.

    COMPUTE EXPANDED-DATE = FUNCTION
                            DATE-TO-YYYYMMDD(981002,-10).
    DISPLAY EXPANDED-DATE.
```

Figure 98. The DATE-TO-YYYYMMDD Intrinsic Function

The value displayed for data item EXPANDED-DATE is 18981002. This result is obtained by the definition of a century window whose **last** year is the sum of the current year and the second argument to the function. The last year of the century window is therefore 1998-10=1988. Consequently, the century window is the 100-year interval ranging from 1889-1988. Since 98 is greater than 89, the 2-digit year is interpreted as representing the year 1898.

2 The year portion of these Julian and Gregorian dates are all only 2 digits in length. A corresponding DATE FORMAT clause has been added to their data description entries so that the compiler will recognize them as windowed date fields.

3 The same fields have been defined in the output record as expanded dates by the addition of an appropriate DATE FORMAT clause. The compiler will therefore recognize these items as 4-digit expanded date fields.

4 The expansion of the date fields from 2 to 4 digits requires that the extraneous space in the output records be reduced to accommodate the increased length of the expanded date fields.

5 The MOVE CORRESPONDING statement moves each item in the input record to its counterpart in the output record. When the windowed date fields are moved to the matching expanded date fields, the compiler automatically expands the year values using the century window.

Figure 99 illustrates the diagnostic messages generated during the compilation of the sample conversion program.

```
LineID  Message code  Message text

    75  IGYDS1454-I   A group item contained one or more subordinate date fields.

                      Same message on line:     99    127    138

   230  IGYPA3309-I   The windowed date value in "EMP-DATE-JOINED (NUMERIC INTEGER)" will be expanded and moved into
                      "EMP-DATE-JOINED (NUMERIC INTEGER)".

   230  IGYPA3309-I   The windowed date value in "EMP-DATE-TERMINATED (NUMERIC INTEGER)" will be expanded and moved into
                      "EMP-DATE-TERMINATED (NUMERIC INTEGER)".

   230  IGYPA3309-I   The windowed date value in "EMP-DATE-MAINTAINED (NUMERIC INTEGER)" will be expanded and moved into
                      "EMP-DATE-MAINTAINED (NUMERIC INTEGER)".

   230  IGYPA3309-I   The windowed date value in "EMP-BIRTH-DATE (NUMERIC INTEGER)" will be expanded and moved into
                      "EMP-BIRTH-DATE (NUMERIC INTEGER)".

   230  IGYPA3309-I   The windowed date value in "EMP-SECURITY-EXP (PACKED INTEGER)" will be expanded and moved into
                      "EMP-SECURITY-EXP (PACKED INTEGER)".

   272  IGYPA3309-I   The windowed date value in "DEPT-DATE-MAINTAINED (PACKED INTEGER)" will be expanded and moved into
                      "DEPT-DATE-MAINTAINED (PACKED INTEGER)".
```

Figure 99. Compiler Diagnostic Messages after Compilation of Conversion Program TARMUCON

7.5 Full Field Expansion

This section documents in detail the full field expansion solution to the Year 2000 problem using one of the COBOL programs in our sample application. We will cover, step by step, the activities involved in converting the program with internal bridging solution to the full field expansion solution.

7.5.1 Definition

The full field-expansion solution involves explicitly expanding 2-digit year date fields to contain full 4-digit years in your files and databases, and then using those fields in expanded form in your programs. This is the only method by which you can be assured of reliable date processing for all applications.

The Millennium Language Extensions allow you to progressively move toward a full field-expansion solution, using the following steps:

1. Apply the short-term (basic remediation) solution, and use this until you have the resources to implement a more permanent solution.
2. Apply the internal bridging scheme. This allows you to use expanded dates in your programs while your files continue to hold dates in 2-digit year form. In turn, this allows you to progress more easily to a full field expansion solution, because there will be no further changes to the logic in the main body of the program.
3. Change the file layouts and database definitions to use 4-digit year dates.
4. Change your COBOL copybooks to reflect these 4-digit year date fields.
5. Run a utility program (or special-purpose COBOL program) to copy from the old format files to the new format. For a sample program, see Figure 97 on page 230.
6. Recompile your programs and perform regression testing and data testing.

After you have completed the first two steps, the remaining steps in the sequence can be repeated any number of times. You do not need to change every date field in every file at the same time. Using this method, you can select files for progressive conversion based on criteria such as business needs or interfaces with other applications.

When you use this method, you will need to write special-purpose programs to convert the files to expanded form. 7.4, “File and Database Conversion” on page 229 shows a sample of program that expands the date fields of a file.

7.5.2 Applicability

This approach is suitable if there is enough time to convert the entire inventory of applications as well as all the data in a well organized and controlled environment. If part of the inventory remains not Year 2000 ready then bridge programs have to be developed and tested. This approach must be used if in one given program run, date data is to be processed that spans more than one hundred years.

7.5.3 Benefits of This Approach

The benefits of this approach are:

- This is a permanent solution; no more changes are required. This solution will allow your programs to function into and beyond the year 2000.
- The performance is the best of the fix alternatives.
- Maintenance is easier.

7.5.4 Limitations of This Approach

The limitations of this approach are these:

- It requires all changes to databases, copybooks, and programs to be synchronized.
- Date variables which have not been identified as such during the find-and-fix phases can only be found during testing. This is late in the process and therefore more expensive.

7.5.5 Sample Application Program Illustrating This Approach

To illustrate the full field expansion technique we use program TARMU3B from our sample application. We used this program in a previous section to explain the internal bridging technique (see 7.3.5.3, “Year 2000 Compliant Application Program with MLE Enhancements” on page 221). We will show the transition activities performed to convert a program that is currently using an internal bridging solution to a program with fully expanded date fields.

As a prerequisite for a full field-expansion technique, we have expanded the date fields on the Employee and Department VSAM files as described in 7.4, “File and Database Conversion” on page 229.

The program in Figure 100 is an example of the program with fully expanded date fields. The comments at the bottom illustrate the transition we made from the internal bridging program (Figure 96 on page 221) to this fully expanded program.

```
       CBL TEST(ALL,SYM) NODYNAM APOST RENT
       *CBL DATEPROC(FLAG) YEARWINDOW(1950)                          1
       IDENTIFICATION DIVISION.
       PROGRAM-ID.   TARMU3B.
       AUTHOR.       MERRILL BANI / RINA SUMIANTORO.
       DATE-WRITTEN. JUNE 1997 / APRIL 1998.
      * -------------------------------------------------------- */
      * LICENSED MATERIALS - PROPERTY OF IBM                     */
      *                                                          */
      * (C) COPYRIGHT IBM CORP. 1997                             */
      *                                                          */
      * US GOVERNMENT USERS RESTRICTED RIGHTS - USE,             */
      * DUPLICATION OR DISCLOSURE RESTRICTED BY GSA ADP          */
      * SCHEDULE CONTRACT WITH IBM CORP.                         */
```

Figure 100 (Part 1 of 10). TARMU3B after Full Field Expansion

```
* ------------------------------------------------------- */
*REMARKS.                                                 */
*                                                         */
*    VA2000 MVS COBOL TEST APPLICATION                    */
*                                                         */
*    ONLINE UPDATE OF EMPLOYEE DATABASE                   */
*                                                         */
*    1. Read batch input                                  */
*                                                         */
*    2. Verify details                                    */
*                                                         */
*    3. If valid, update employee database                */
*                                                         */
* ------------------------------------------------------- */
 EJECT
 ENVIRONMENT DIVISION.

 INPUT-OUTPUT SECTION.

 FILE-CONTROL.

     SELECT EMPLOYEE-MASTER-FILE
            ASSIGN TO EMPMAST
            ORGANIZATION IS INDEXED
            ACCESS IS RANDOM
            RECORD KEY IS EMP-ID
            FILE STATUS IS EMP-FILE-STATUS.

     SELECT DEPT-MASTER-FILE
            ASSIGN TO DEPTMAST
            ORGANIZATION IS INDEXED
            ACCESS IS RANDOM
            RECORD KEY IS DEPT-CODE
            FILE STATUS IS DEPT-FILE-STATUS.

     SELECT INPUT-FILE
            ASSIGN TO INPFILE
            ORGANIZATION IS SEQUENTIAL
            ACCESS IS SEQUENTIAL
            FILE STATUS IS INP-FILE-STATUS.

 EJECT

 DATA DIVISION.

 FILE SECTION.

 FD  EMPLOYEE-MASTER-FILE
     RECORD CONTAINS 200 CHARACTERS.
*
 01  EMPLOYEE-MASTER-RECORD.
*        ** key field
      03  EMP-ID                     PIC X(6).
      03  EMP-DEPT-CODE              PIC X(4).
      03  EMP-NAME                   PIC X(30).
      03  EMP-ADDR-1                 PIC X(30).
      03  EMP-ADDR-2                 PIC X(30).
      03  EMP-ADDR-3                 PIC X(30).
      03  EMP-ZIP-CODE               PIC X(5).
*        ** format (yyyyddd)
      03  EMP-DATE-JOINED            PIC 9(7).                    2
*        ** format (yyyymmdd)
      03  EMP-DATE-TERMINATED        PIC 9(8).
*        ** format (yyyyddd)
      03  EMP-DATE-MAINTAINED        PIC 9(7).
*        ** format (yyyyddd)
      03  EMP-BIRTH-DATE             PIC 9(7).
*        ** format (yyyyddd)
      03  EMP-SECURITY-EXP           PIC 9(7) COMP-3.
      03  FILLER                     PIC X(32).
```

Figure 100 (Part 2 of 10). TARMU3B after Full Field Expansion

```
 FD  DEPT-MASTER-FILE
     RECORD CONTAINS 80 CHARACTERS.
*
 01  DEPT-MASTER-RECORD.
*        ** key field
     03  DEPT-CODE                  PIC X(4).
     03  DEPT-DESCRIPTION           PIC X(30).
*        ** format (yyyyddd) packed
     03  DEPT-DATE-MAINTAINED       PIC 9(7) COMP-3.                  3
     03  FILLER                     PIC X(42).

 FD  INPUT-FILE
     RECORD CONTAINS 175 CHARACTERS.                                   4
*
 01  INPUT-RECORD.
*        ** key field
     03  INP-EMP-ID                 PIC X(6).
     03  INP-EMP-DEPT-CODE          PIC X(4).
     03  INP-EMP-NAME               PIC X(30).
     03  INP-EMP-ADDR-1             PIC X(30).
     03  INP-EMP-ADDR-2             PIC X(30).
     03  INP-EMP-ADDR-3             PIC X(30).
     03  INP-EMP-ZIP-CODE           PIC X(5).
*        ** format (mm/dd/yyyy)
     03  INP-EMP-DATE-JOINED        PIC 9(10).                        4
*        ** format (mm/dd/yyyy)
     03  INP-EMP-DATE-TERMINATED    PIC 9(10).
*        ** format (mm/dd/yyyy)
     03  INP-EMP-BIRTH-DATE         PIC 9(10).
*        ** format (mm/dd/yyyy)
     03  INP-EMP-SECURITY-EXP       PIC 9(10).

 WORKING-STORAGE SECTION.

 01  OUTPUT-RECORD.
*        ** key field
     03  OUT-EMP-ID                 PIC X(6).
     03  OUT-EMP-DEPT-CODE          PIC X(4).
     03  OUT-EMP-NAME               PIC X(30).
     03  OUT-EMP-ADDR-1             PIC X(30).
     03  OUT-EMP-ADDR-2             PIC X(30).
     03  OUT-EMP-ADDR-3             PIC X(30).
     03  OUT-EMP-ZIP-CODE           PIC X(5).
*        ** format (mm/dd/yyyy)
     03  OUT-EMP-DATE-JOINED        PIC X(10).                        5
*        ** format (mm/dd/yyyy)
     03  OUT-EMP-DATE-TERMINATED    PIC X(10).
*        ** format (mm/dd/yyyy)
     03  OUT-EMP-DATE-MAINTAINED    PIC X(10).
*        ** format (mm/dd/yyyy)
     03  OUT-EMP-BIRTH-DATE         PIC X(10).
*        ** format (mm/dd/yyyy)
     03  OUT-EMP-SECURITY-EXP       PIC X(8).
     03  FILLER                     PIC X(33).

 01  SWITCHES.
     03  ERROR-SWITCH               PIC X VALUE SPACE.
         88  ERRORS                       VALUE "Y".
 01  ADD-RECORD                     PIC X VALUE SPACES.
 01  CHANGE-RECORD                  PIC X VALUE SPACES.

 01  SWITCH-OFF                     PIC X VALUE "N".

*    ** message table
```

Figure 100 (Part 3 of 10). TARMU3B after Full Field Expansion

```
       01  MESSAGE-TABLE.
           03  FILLER  PIC X(30) VALUE "I-RECORD ADDED             ".
           03  FILLER  PIC X(30) VALUE "I-RECORD CHANGED           ".
           03  FILLER  PIC X(30) VALUE "E-DEPARTMENT CODE INVALID  ".
           03  FILLER  PIC X(30) VALUE "E-ZIP CODE NOT NUMERIC     ".
           03  FILLER  PIC X(30) VALUE "E-INVALID DATE             ".
           03  FILLER  PIC X(30) VALUE "I-ENTER EMPLOYEE NUMBER    ".
           03  FILLER  PIC X(30) VALUE "I-ENTER EMPLOYEE DETAILS   ".
           03  FILLER  PIC X(30) VALUE "I-ENTER CHANGE DETAILS     ".
           03  FILLER  PIC X(30) VALUE "I-SCR VALID PF10 TO UPDATE ".
           03  FILLER  PIC X(30) VALUE "E-NAME MISSING             ".
           03  FILLER  PIC X(30) VALUE "E-ADDRESS LINE 1 MISSING   ".
           03  FILLER  PIC X(30) VALUE "E-JOINED > TERMINATED DATE ".
           03  FILLER  PIC X(30) VALUE "E-DATE MISSING             ".
           03  FILLER  PIC X(30) VALUE "E-EMPLOYEE NO NOT NUMERIC  ".

       01  MSG-TABLE-RED REDEFINES MESSAGE-TABLE.
           03  MSG OCCURS 14 TIMES.
               05  FILLER                  PIC X(30).

      *    ** work variables

       01  WORK-VARS.                                                    6
           03  WORK-TODAYS-MMDDYYYY        PIC 9(10).
           03  WORK-MSG-CODE               PIC 99.
           03  WORK-EIB-DATE               PIC 9(9).
           03  WORK-EIB-DATE-RED REDEFINES WORK-EIB-DATE.
               05  FILLER                  PIC X(1).
               05  WORK-EIB-CENTURY        PIC X(1).
               05  WORK-EIB-YYYYDDD        PIC X(7).
           03  WORK-JOINED-YYYYDDD         PIC 9(7).
           03  WORK-JOINED-MMDDYYYY.
               05  WORK-JOINED-MMDD        PIC X(6).
               05  WORK-JOINED-YYYY        PIC 9(4).
           03  WORK-SEC-EXP                PIC 9(5).
           03  WORK-TERMINATED-YYYYDDD     PIC 9(7).

       01  FILE-STATUS-FIELDS.
           03  EMP-FILE-STATUS             PIC XX.
           03  DEPT-FILE-STATUS            PIC XX.
           03  INP-FILE-STATUS             PIC XX.

       01  EMP-LENGTH                      PIC S9(4) COMP VALUE +200.
       01  DEP-LENGTH                      PIC S9(4) COMP VALUE +80.
       01  RESPONSE                        PIC S9(8) COMP VALUE +0.
       01  LAST-EMP-ID                     PIC X(6)  VALUE SPACES.

       01  TARDATE-PARAMETERS.                                           7
           03  TARDATE-DATE                PIC X(10).
           03  TARDATE-DATE-RED1 REDEFINES TARDATE-DATE.
               05  TARDATE-DATE-YYYYDDD    PIC 9(7).
               05  FILLER                  PIC X(3).
           03  TARDATE-DATE-RED2 REDEFINES TARDATE-DATE.
               05  TARDATE-DATE-YYYYMMDD   PIC 9(8).
               05  FILLER                  PIC X(2).
           03  TARDATE-INPUT-FORMAT        PIC X(10).
           03  TARDATE-OUTPUT-FORMAT       PIC X(10).
           03  TARDATE-MESSAGE             PIC X(30).

      *    ** expanded date variables

      *    ** format (yyyyddd)  expanded
      *01  EMP-DATE-JOINED           PIC 9(7) DATE FORMAT YYYYXXX.       8
      *    ** format (yyyymmdd) expanded
      *01  EMP-DATE-TERMINATED       PIC 9(8) DATE FORMAT YYYYXXXX.
      *    ** format (yyyyddd)  expanded
      *01  EMP-DATE-MAINTAINED       PIC 9(7) DATE FORMAT YYYYXXX.
      *    ** format (yyyyddd)  expanded
      *01  EMP-BIRTH-DATE            PIC 9(7) DATE FORMAT YYYYXXX.
      *    ** format (yyyyddd)  expanded
      *01  EMP-SECURITY-EXP          PIC 9(7) COMP-3
      *                                    DATE FORMAT YYYYXXX.
```

Figure 100 (Part 4 of 10). TARMU3B after Full Field Expansion

```
           EJECT
       PROCEDURE DIVISION.

           OPEN I-O   EMPLOYEE-MASTER-FILE
           OPEN INPUT DEPT-MASTER-FILE
           OPEN INPUT INPUT-FILE

      *    **  retrieve todays date

           ACCEPT WORK-EIB-YYYYDDD FROM  DAY YYYYDDD.                    9
           MOVE   WORK-EIB-YYYYDDD TO    TARDATE-DATE-YYYYDDD.           10
           MOVE   "YYYYDDD"        TO    TARDATE-INPUT-FORMAT.
           MOVE   "MM/DD/YYYY"     TO    TARDATE-OUTPUT-FORMAT.
           CALL   "TARDTE3X"       USING TARDATE-DATE
                                         TARDATE-INPUT-FORMAT
                                         TARDATE-OUTPUT-FORMAT
                                         TARDATE-MESSAGE.
           MOVE TARDATE-DATE     TO      WORK-TODAYS-MMDDYYYY.

           PERFORM 200-READ-INPUT-FILE THRU 200-EXIT.

           STOP RUN.

      *================================================================

       200-READ-INPUT-FILE.

           READ INPUT-FILE
               AT END
               GO TO 200-EXIT
           END-READ.

      *    **  validate employee number

           IF INP-EMP-ID = ZERO THEN
               MOVE 6              TO  WORK-MSG-CODE
               DISPLAY MSG(WORK-MSG-CODE)
               GO TO 200-READ-NEXT-INPUT-FILE
           ELSE
              IF INP-EMP-ID IS NOT NUMERIC THEN
                 MOVE 14             TO  WORK-MSG-CODE
                 DISPLAY MSG(WORK-MSG-CODE)
                 GO TO 200-READ-NEXT-INPUT-FILE
              END-IF
           END-IF.

           PERFORM 400-READ-EMPLOYEE-MASTER-FILE THRU 400-EXIT
           PERFORM 500-VALIDATE-INPUT-DATA THRU 500-EXIT.
           IF NOT ERRORS
              PERFORM 600-UPDATE-FILE THRU 600-EXIT
           END-IF.

       200-READ-NEXT-INPUT-FILE.

           DISPLAY " "
           GO TO 200-READ-INPUT-FILE.

       200-EXIT.
           EXIT.

      *================================================================

       400-READ-EMPLOYEE-MASTER-FILE.

           MOVE SPACES TO CHANGE-RECORD ADD-RECORD.
           MOVE SPACES TO EMPLOYEE-MASTER-RECORD.

           MOVE INP-EMP-ID TO EMP-ID
           READ EMPLOYEE-MASTER-FILE
                KEY IS EMP-ID.
```

Figure 100 (Part 5 of 10). TARMU3B after Full Field Expansion

```
*    **  MOVE THE OLD NON-EXPANDED DATE VARIABLES TO THE NEW
*    **  EXPANDED DATE VARIABLES AFTER READING THE FILE.

*    MOVE OLD-EMP-DATE-JOINED      TO EMP-DATE-JOINED.                  8
*    MOVE OLD-EMP-DATE-TERMINATED  TO EMP-DATE-TERMINATED.
*    MOVE OLD-EMP-DATE-MAINTAINED  TO EMP-DATE-MAINTAINED.
*    MOVE OLD-EMP-BIRTH-DATE       TO EMP-BIRTH-DATE.
*    MOVE OLD-EMP-SECURITY-EXP     TO EMP-SECURITY-EXP.

     MOVE LOW-VALUES TO OUTPUT-RECORD.
     MOVE EMP-ID     TO OUT-EMP-ID.

     IF EMP-FILE-STATUS = "23" THEN
         MOVE "Y"                  TO ADD-RECORD
         MOVE SPACES               TO OUT-EMP-DEPT-CODE
                                      OUT-EMP-NAME
                                      OUT-EMP-ADDR-1
                                      OUT-EMP-ADDR-2
                                      OUT-EMP-ADDR-3
                                      OUT-EMP-ZIP-CODE
                                      OUT-EMP-DATE-JOINED
                                      OUT-EMP-BIRTH-DATE
                                      OUT-EMP-DATE-TERMINATED
                                      OUT-EMP-SECURITY-EXP
     ELSE
         MOVE "Y"                  TO CHANGE-RECORD
         MOVE EMP-DEPT-CODE        TO OUT-EMP-DEPT-CODE
         MOVE EMP-NAME             TO OUT-EMP-NAME
         MOVE EMP-ADDR-1           TO OUT-EMP-ADDR-1
         MOVE EMP-ADDR-2           TO OUT-EMP-ADDR-2
         MOVE EMP-ADDR-3           TO OUT-EMP-ADDR-3
         MOVE EMP-ZIP-CODE         TO OUT-EMP-ZIP-CODE
*
         MOVE EMP-DATE-JOINED      TO TARDATE-DATE-YYYYDDD              10
         MOVE "YYYYDDD"            TO TARDATE-INPUT-FORMAT
         MOVE "MM/DD/YYYY"         TO TARDATE-OUTPUT-FORMAT
         CALL "TARDTE3X" USING TARDATE-DATE
                               TARDATE-INPUT-FORMAT
                               TARDATE-OUTPUT-FORMAT
                               TARDATE-MESSAGE
         MOVE TARDATE-DATE         TO OUT-EMP-DATE-JOINED
*
         MOVE EMP-BIRTH-DATE       TO TARDATE-DATE-YYYYDDD              10
         MOVE "YYYYDDD"              TO TARDATE-INPUT-FORMAT
         MOVE "MM/DD/YYYY"         TO TARDATE-OUTPUT-FORMAT
         CALL "TARDTE3X" USING TARDATE-DATE
                             TARDATE-INPUT-FORMAT
                             TARDATE-OUTPUT-FORMAT
                             TARDATE-MESSAGE
         MOVE TARDATE-DATE         TO OUT-EMP-BIRTH-DATE
*
         IF  EMP-DATE-TERMINATED > ZEROS THEN
             MOVE EMP-DATE-TERMINATED  TO TARDATE-DATE-YYYYMMDD       10
             MOVE "YYYYMMDD"           TO TARDATE-INPUT-FORMAT
             MOVE "MM/DD/YYYY"         TO TARDATE-OUTPUT-FORMAT
             CALL "TARDTE3X" USING TARDATE-DATE
                                   TARDATE-INPUT-FORMAT
                                   TARDATE-OUTPUT-FORMAT
                                   TARDATE-MESSAGE
             MOVE TARDATE-DATE         TO OUT-EMP-DATE-TERMINATED
         ELSE
             MOVE SPACES               TO OUT-EMP-DATE-TERMINATED
         END-IF
```

Figure 100 (Part 6 of 10). TARMU3B after Full Field Expansion

```
*
         MOVE EMP-SECURITY-EXP     TO TARDATE-DATE-YYYYDDD              10
         MOVE "YYYYDDD"            TO TARDATE-INPUT-FORMAT
         MOVE "MM/DD/YYYY"         TO TARDATE-OUTPUT-FORMAT
         CALL "TARDTE3X" USING TARDATE-DATE
                               TARDATE-INPUT-FORMAT
                               TARDATE-OUTPUT-FORMAT
                               TARDATE-MESSAGE
         MOVE TARDATE-DATE         TO OUT-EMP-SECURITY-EXP
*
     END-IF.

 400-EXIT.
     EXIT.

*================================================================

 500-VALIDATE-INPUT-DATA.

     MOVE SWITCH-OFF  TO  ERROR-SWITCH.

*    **    verify dept code

     MOVE INP-EMP-DEPT-CODE TO DEPT-CODE
     READ DEPT-MASTER-FILE
          KEY IS DEPT-CODE
          IF DEPT-FILE-STATUS = "23" THEN
             MOVE 3 TO WORK-MSG-CODE
             DISPLAY MSG(WORK-MSG-CODE)
             SET ERRORS TO TRUE
          END-IF

     IF  INP-EMP-NAME IS NOT > SPACES   THEN
         MOVE 10        TO WORK-MSG-CODE
         DISPLAY MSG(WORK-MSG-CODE)
         SET ERRORS TO TRUE
     END-IF

     IF  INP-EMP-ADDR-1 IS NOT > SPACES   THEN
         MOVE 11        TO WORK-MSG-CODE
         DISPLAY MSG(WORK-MSG-CODE)
         SET ERRORS TO TRUE
     END-IF

     IF  INP-EMP-ZIP-CODE IS NOT NUMERIC   THEN
         MOVE 4         TO WORK-MSG-CODE
         DISPLAY MSG(WORK-MSG-CODE)
         SET ERRORS TO TRUE
     END-IF

     IF  INP-EMP-DATE-JOINED > SPACES THEN
         MOVE INP-EMP-DATE-JOINED   TO TARDATE-DATE
         MOVE "MM/DD/YYYY"    TO TARDATE-INPUT-FORMAT                  10
         MOVE "YYYYDDD"       TO TARDATE-OUTPUT-FORMAT
         CALL "TARDTE3X" USING TARDATE-DATE
                               TARDATE-INPUT-FORMAT
                               TARDATE-OUTPUT-FORMAT
                               TARDATE-MESSAGE
         IF  TARDATE-MESSAGE NOT = SPACES THEN
             MOVE 5     TO WORK-MSG-CODE
             DISPLAY MSG(WORK-MSG-CODE)
             SET ERRORS TO TRUE
         ELSE
             MOVE TARDATE-DATE-YYYYDDD TO WORK-JOINED-YYYYDDD
             MOVE INP-EMP-DATE-JOINED TO WORK-JOINED-MMDDYYYY
         END-IF
```

Figure 100 (Part 7 of 10). TARMU3B after Full Field Expansion

```
           ELSE
               MOVE WORK-TODAYS-MMDDYYYY TO OUT-EMP-DATE-JOINED
                                            WORK-JOINED-MMDDYYYY
               MOVE WORK-EIB-YYYYDDD   TO WORK-JOINED-YYYYDDD
           END-IF

           IF  INP-EMP-BIRTH-DATE > SPACES     THEN
               MOVE INP-EMP-BIRTH-DATE   TO TARDATE-DATE
               MOVE "MM/DD/YYYY"    TO TARDATE-INPUT-FORMAT                  10
               MOVE "YYYYDDD"       TO TARDATE-OUTPUT-FORMAT
               CALL "TARDTE3X" USING TARDATE-DATE
                                     TARDATE-INPUT-FORMAT
                                     TARDATE-OUTPUT-FORMAT
                                     TARDATE-MESSAGE
               IF  TARDATE-MESSAGE NOT = SPACES THEN
                   MOVE 5     TO WORK-MSG-CODE
                   DISPLAY MSG(WORK-MSG-CODE)
                   SET ERRORS TO TRUE
               END-IF
           ELSE
               MOVE 13    TO WORK-MSG-CODE
               DISPLAY MSG(WORK-MSG-CODE)
               SET ERRORS TO TRUE
           END-IF

           IF  INP-EMP-DATE-TERMINATED > SPACES THEN
               MOVE INP-EMP-DATE-TERMINATED TO TARDATE-DATE
               MOVE "MM/DD/YYYY"    TO TARDATE-INPUT-FORMAT                  10
               MOVE "YYYYDDD"       TO TARDATE-OUTPUT-FORMAT
               CALL "TARDTE3X" USING TARDATE-DATE
                                     TARDATE-INPUT-FORMAT
                                     TARDATE-OUTPUT-FORMAT
                                     TARDATE-MESSAGE
               IF  TARDATE-MESSAGE NOT = SPACES THEN
                   MOVE 5     TO WORK-MSG-CODE
                   DISPLAY MSG(WORK-MSG-CODE)
                   SET ERRORS TO TRUE
               ELSE
                   MOVE OUT-EMP-DATE-TERMINATED TO OUT-EMP-SECURITY-EXP
                   MOVE TARDATE-DATE-YYYYDDD TO WORK-TERMINATED-YYYYDDD
                   IF  WORK-TERMINATED-YYYYDDD < WORK-JOINED-YYYYDDD
                       MOVE 12    TO WORK-MSG-CODE
                       DISPLAY MSG(WORK-MSG-CODE)
                       SET ERRORS TO TRUE
                   END-IF
               END-IF
           END-IF.

           IF  INP-EMP-SECURITY-EXP > SPACES     THEN
               MOVE INP-EMP-SECURITY-EXP TO TARDATE-DATE
               MOVE "MM/DD/YYYY"    TO TARDATE-INPUT-FORMAT                  10
               MOVE "YYYYDDD"       TO TARDATE-OUTPUT-FORMAT
               CALL "TARDTE3X" USING TARDATE-DATE
                                     TARDATE-INPUT-FORMAT
                                     TARDATE-OUTPUT-FORMAT
                                     TARDATE-MESSAGE
               IF  TARDATE-MESSAGE NOT = SPACES THEN
                   MOVE 5     TO WORK-MSG-CODE
                   DISPLAY MSG(WORK-MSG-CODE)
                   SET ERRORS TO TRUE
               END-IF
           ELSE
               COMPUTE WORK-JOINED-YYYY = WORK-JOINED-YYYY + 1
               IF  WORK-JOINED-MMDD = "02/29/" THEN
                   MOVE "02/28/"  TO WORK-JOINED-MMDD
               END-IF
               MOVE WORK-JOINED-MMDDYYYY TO OUT-EMP-SECURITY-EXP
           END-IF.

       500-EXIT.
           EXIT.
```

Figure 100 (Part 8 of 10). TARMU3B after Full Field Expansion

```
*=============================================================

 600-UPDATE-FILE.

     MOVE SPACES             TO  EMPLOYEE-MASTER-RECORD.
     MOVE INP-EMP-ID         TO  EMP-ID.
     MOVE INP-EMP-DEPT-CODE  TO  EMP-DEPT-CODE.
     MOVE INP-EMP-NAME       TO  EMP-NAME.
     MOVE INP-EMP-ADDR-1     TO  EMP-ADDR-1.
     MOVE INP-EMP-ADDR-2     TO  EMP-ADDR-2.
     MOVE INP-EMP-ADDR-3     TO  EMP-ADDR-3.
     MOVE INP-EMP-ZIP-CODE   TO  EMP-ZIP-CODE.
     MOVE INP-EMP-DATE-JOINED TO  TARDATE-DATE
     MOVE "MM/DD/YYYY"       TO  TARDATE-INPUT-FORMAT                10
     MOVE "YYYYDDD"          TO  TARDATE-OUTPUT-FORMAT
     CALL "TARDTE3X" USING       TARDATE-DATE
                                 TARDATE-INPUT-FORMAT
                                 TARDATE-OUTPUT-FORMAT
                                 TARDATE-MESSAGE.
     MOVE TARDATE-DATE-YYYYDDD TO  EMP-DATE-JOINED.
     MOVE INP-EMP-BIRTH-DATE TO  TARDATE-DATE
     MOVE "MM/DD/YYYY"       TO  TARDATE-INPUT-FORMAT                10
     MOVE "YYYYDDD"          TO  TARDATE-OUTPUT-FORMAT
     CALL "TARDTE3X" USING       TARDATE-DATE
                                 TARDATE-INPUT-FORMAT
                                 TARDATE-OUTPUT-FORMAT
                                 TARDATE-MESSAGE.
     MOVE TARDATE-DATE-YYYYDDD TO  EMP-BIRTH-DATE.

     IF  INP-EMP-DATE-TERMINATED > SPACES THEN
         MOVE INP-EMP-DATE-TERMINATED TO TARDATE-DATE
         MOVE "MM/DD/YYYY"    TO TARDATE-INPUT-FORMAT                10
         MOVE "YYYYMMDD"      TO TARDATE-OUTPUT-FORMAT
         CALL "TARDTE3X" USING   TARDATE-DATE
                                 TARDATE-INPUT-FORMAT
                                 TARDATE-OUTPUT-FORMAT
                                 TARDATE-MESSAGE
         MOVE TARDATE-DATE-YYYYMMDD TO  EMP-DATE-TERMINATED
     ELSE
         MOVE ZEROS           TO  EMP-DATE-TERMINATED
     END-IF.
     MOVE INP-EMP-SECURITY-EXP TO  TARDATE-DATE
     MOVE "MM/DD/YYYY"       TO  TARDATE-INPUT-FORMAT                10
     MOVE "YYYYDDD"          TO  TARDATE-OUTPUT-FORMAT
     CALL "TARDTE3X" USING       TARDATE-DATE
                                 TARDATE-INPUT-FORMAT
                                 TARDATE-OUTPUT-FORMAT
                                 TARDATE-MESSAGE.
     MOVE TARDATE-DATE-YYYYDDD TO  EMP-SECURITY-EXP.

     MOVE WORK-EIB-YYYYDDD    TO  EMP-DATE-MAINTAINED.

*    **  MOVE THE NEW EXPANDED DATE VARIABLES BACK TO THE OLD
*    **  NON-EXPANDED DATE VARIABLES BEFORE UPDATING THE FILE.

*    COMPUTE OLD-EMP-DATE-JOINED      = EMP-DATE-JOINED                8
*      ON SIZE ERROR GO TO 600-OUT-OF-WINDOW-ERROR.
*
*    IF FUNCTION UNDATE(EMP-DATE-TERMINATED) = 00000000
*      MOVE EMP-DATE-TERMINATED TO OLD-EMP-DATE-TERMINATED
*    ELSE
*      COMPUTE OLD-EMP-DATE-TERMINATED  = EMP-DATE-TERMINATED
*        ON SIZE ERROR GO TO 600-OUT-OF-WINDOW-ERROR
*    END-IF.
*
*    COMPUTE OLD-EMP-DATE-MAINTAINED  = EMP-DATE-MAINTAINED
*      ON SIZE ERROR GO TO 600-OUT-OF-WINDOW-ERROR.
*
*    COMPUTE OLD-EMP-BIRTH-DATE       = EMP-BIRTH-DATE
*      ON SIZE ERROR GO TO 600-OUT-OF-WINDOW-ERROR.
*
*    COMPUTE OLD-EMP-SECURITY-EXP     = EMP-SECURITY-EXP
*      ON SIZE ERROR GO TO 600-OUT-OF-WINDOW-ERROR.
```

Figure 100 (Part 9 of 10). TARMU3B after Full Field Expansion

```
           IF  ADD-RECORD = "Y"  THEN
               WRITE EMPLOYEE-MASTER-RECORD
               MOVE 1    TO  WORK-MSG-CODE
               DISPLAY MSG(WORK-MSG-CODE)
               MOVE SPACES TO ADD-RECORD
      *        GO TO 600-EXIT
           ELSE
               REWRITE EMPLOYEE-MASTER-RECORD
               MOVE 2    TO  WORK-MSG-CODE
               DISPLAY MSG(WORK-MSG-CODE)
               MOVE SPACES TO CHANGE-RECORD
      *        GO TO 600-EXIT
           END-IF.

      *600-OUT-OF-WINDOW-ERROR.                                          8
      *    DISPLAY 'OUT OF WINDOW CONDITION HAS OCCURRED'.

       600-EXIT.
           EXIT.
```

Figure 100 (Part 10 of 10). TARMU3B after Full Field Expansion

The following notes apply to Figure 100 on page 237:

1 Take out the DATEPROC and YEARWINDOW compiler options at the . beginning of the program.

2 Rename the 2-digit year dates in the Employee VSAM file back to the original names by taking out **OLD-** from the field names. Also, take out the DATE FORMAT clause. Change the layout of the file definitions of the Employee file to use 2-digit year dates.

3 Change the layout of the file definitions of the Department VSAM file to use 4-digit year dates.

4 Change the layout of the file definitions of INPUT-RECORD file to use 4-digit year dates.

5 Change the layout of the file definitions of OUTPUT-RECORD in the Working-Storage section to use 4-digit year dates.

6 Take out the DATE FORMAT clauses of WORK-VARS that we added previously for the internal bridging program and change the layout of the file definitions to use 4-digit year dates.

7 Take out the DATE FORMAT clauses of TARDATE-PARAMETERS that we added previously for the internal bridging program and change the layout of the file definitions to use 4-digit year dates.

8 Since the date fields of the Employee VSAM file has been changed to 4-digit year dates, the following portions of the program that we previously added to create the internal bridging program should be taken out:

- The declaration of 4-digit year dates in the Working Storage section that was used as expanded date fields.
- The moving of the 2-digit year dates to the 4-digit year dates.
- The moving of the 4-digit year dates back to the 2-digit year dates.

9 Change the ACCEPT statement to receive a 4-digit year date.

10 This full field-expansion program is calling the TARDTE3X module, which is the expanded version of TARDTE3. The following is a sample of how to call module TARDTE3X

```
MOVE EMP-DATE-JOINED      TO TARDATE-DATE-YYYYDDD
MOVE "YYYYDDD"            TO TARDATE-INPUT-FORMAT
MOVE "MM/DD/YYYY"         TO TARDATE-OUTPUT-FORMAT
CALL "TARDTE3X"        USING TARDATE-DATE
                             TARDATE-INPUT-FORMAT
                             TARDATE-OUTPUT-FORMAT
                             TARDATE-MESSAGE
MOVE TARDATE-DATE         TO OUT-EMP-DATE-JOINED
```

For that purpose, we need to change the parameter to call module TARDTE3X:

- Use the expanded field name TARDATE-DATE-YYYYDDD instead of TARDATE-DATE-YYYDDD
- Use "YYYYDDD" as input format
- Use "MM/DD/YYYY" as the output format

Appendix A. Files Used in This Book

This appendix lists the files used in this book. The sample application can be found and downloaded as a zip-file at

`http://www.redbooks.ibm.com/`

Click the *Additional Materials* button on the left side of this web page and then select SG245253 from the list.

The code and report samples are grouped by

- Files used for MA2000
- Files used for MLE

The lowest-level qualifiers of the partitioned data sets are used as sub-directories on the workstation when you unpack the zip-file.

A.1 MA2000

The following tables list the sample files that we used to explain the Maintenance 2000 product.

A.1.1 hlq.SAMPAPPL.COBOL

This PDS contains the COBOL source programs of our sample application.

Table 31. COBOL Source Programs of the MA2000 Sample Application

Member	Description
TARDTE3	Date utility program
TARMU3	Online program to update employee data
TARMU3B	Batch program to update employee data
TARMU5	Batch program to produce report that lists security expiration date of all employees
TARMU6	Batch program to produce report that lists employees whose security expiration date has expired

A.1.2 hlq.SAMPAPPL.COPYBOOK

This PDS contains the copybooks of our sample application.

Table 32. Copybooks of the MA2000 Sample Application

Member	Description
DFHAID	CICS online copybook used by program TARMU3
DFHBMSCA	CICS online copybook used by program TARMU3
TARDTE3C	Copybook used by program TARDTE3
TARMU3BC	Copybook used by program TARMU3B
TARMU3M	Copybook used by program TARMU3 (CICS map definitions)
TARMU5C	Copybook used by program TARMU5
TARMU6C	Copybook used by program TARMU6

A.1.3 hlq.SAMPAPPL.JCL

This PDS contains the JCLs of our sample application.

Table 33. JCLs of the MA2000 Sample Application

Member	Description
TARMU3B	JCL to run program TARMU3B
TARMU5	JCL to run program TARMU5
TARMU6	JCL to run program TARMU6
MERGEMEM	JCL to combine and sort starting-point files

A.1.4 hlq.SAMPAPPL.JCLPRC

This PDS contains the JCL procedures of our sample application.

Table 34. JCL Procedures of the MA2000 Sample Application

Member	Description
TARMU5	JCL Procedure used by JCL TARMU5
TARMU6	JCL Procedure used by JCL TARMU6

A.1.5 hlq.MA2000.SPOINT

This PDS contains the starting-point files created by MA2000 with a record length of 256.

Table 35. Starting-Point Files (LRECL=256)

Member	Description
SIMILAR	Starting-point file generated by the similar item search function
SYNONYM	Starting-point file generated by the synonym search function
DATEFUN	Starting-point file generated by the DATE function search function
YEAR2000	Combined and edited starting-point file

A.1.6 hlq.MA2000.SPOINT80

This PDS contains the starting-point files created by MA2000 with a record length of 80.

Table 36. Starting-Point Files (LRECL=80)

Member	Description
DATASET	Starting-point file generated by the data set starting point file creation option
CICSFIL	Starting-point file generated by the CICS files starting point file creation option

A.2 MLE

The following tables list the sample files that we used to explain the Millennium Language Extensions.

A.2.1 hlq.MLESAMPL.COBOL

This PDS contains the COBOL source programs of our sample application. The copybooks have been copied into the program sources.

Table 37. COBOL Source Programs of the MLE Sample Application

Member	Description
TARDTE3X	Subprogram to re-pattern dates using Language Environment Callable Services
TARMUCON	Program that converts the employee and department files to contain 4-digit year fields
TARMU3BB	Program that illustrates Internal Bridging
TARMU3BF	Program that illustrates Full Field Expansion
TARMU5E	Program that illustrates Basic Remediation
TARMU6E	Program that illustrates Basic Remediation
TARMU6F	Modified version of TARMU6E demonstrating the coordination of multiple century windows within the same program

A.2.2 hlq.MLESAMPL.JCL

This PDS contains the JCLs of our sample application.

Table 38. JCLs of the MLE Sample Application

Member	Description
TARMUCON	JCL to run program TARMUCON
TARMU3BB	JCL to run program TARMU3BB
TARMU5E	JCL to run program TARMU5E
TARMU6E	JCL to run program TARMU6E
VSAM	JCL to create VSAM data set and load it from flat file
COMPILE	JCL to compile and link edit COBOL source programs

A.2.3 Sequential Data Sets

The following table lists the input and output data sets of our sample application.

Table 39. Sequential Data Sets

Member	Description
hlq.EMP.DATA	Employee master file
hlq.DEPT.DATA	Department file
hlq.TARMU3B.INPUT	Sample input file for program TARMU3B
hlq.TARMU5.REPORT	Sample output report of program TARMU5
hlq.TARMU6.REPORT	Sample output report of program TARMU6

Appendix B. Special Notices

This publication is intended to help system programmers and application developers to configure and use the products contained in the IBM COBOL Millennium Language Extensions Automated Suite for OS/390 & MVS/ESA. The information in this publication is not intended as the specification of any programming interfaces that are provided by Maintenance 2000, CCCA, or IBM COBOL for OS/390 with Millennium Language Extensions. See the PUBLICATIONS section of the IBM Programming Announcement for Maintenance 2000, CCCA, or IBM COBOL for OS/390 with Millennium Language Extensions for more information about what publications are considered to be product documentation.

References in this publication to IBM products, programs or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM′s product, program, or service may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact IBM Corporation, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

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

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer′s ability to evaluate and integrate them into the customer′s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any pointers in this publication to external Web sites are provided for convenience only and do not in any manner serve as an endorsement of these Web sites.

The following terms are trademarks of the International Business Machines Corporation in the United States and/or other countries:

AD/Cycle®

AS/400®

BookManager®
CICS®
DB2®
IBM®
Language Environment®
MVS/ESA
RS/6000
System/390®
C/370
COBOL/370
DFSORT
IMS
MVS® (logo)
OS/390
SP
VisualAge®

The following terms are trademarks of other companies:

C-bus is a trademark of Corollary, Inc.

Java and HotJava are trademarks of Sun Microsystems, Incorporated.

Microsoft, Windows, Windows NT, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

PC Direct is a trademark of Ziff Communications Company and is used by IBM Corporation under license.

Pentium, MMX, ProShare, LANDesk, and ActionMedia are trademarks or registered trademarks of Intel Corporation in the U.S. and other countries.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

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

Appendix C. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

C.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see “How to Get ITSO Redbooks” on page 257.

- *VisualAge 2000 Test Solution: Testing Your Year 2000 Conversion*, SG24-2230

C.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038
Lotus Redbooks Collection	SBOF-6899	SK2T-8039
Tivoli Redbooks Collection	SBOF-6898	SK2T-8044
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RS/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RS/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041
RS/6000 Redbooks Collection (PDF Format)	SBOF-8700	SK2T-8043
Application Development Redbooks Collection	SBOF-7290	SK2T-8037

C.3 Other Publications

These publications are also relevant as further information sources:

- *Maintenance 2000 Guide and Reference*, SC18-2309
- *COBOL and CICS Command Level Conversion Aid for OS/390 & MVS & VM User's Guide*, SC26-9400
- *IBM COBOL for OS/390 & VM Programming Guide*, SC26-9049
- *VisualAge COBOL Millennium Language Extensions Guide*, GC26-9266

How to Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, redpieces, and CD-ROMs. A form for ordering books and CD-ROMs by fax or e-mail is also provided.

- **Redbooks Web Site** http://www.redbooks.ibm.com/

  Search for, view, download or order hardcopy/CD-ROMs redbooks from the redbooks Web site. Also read redpieces and download additional materials (code samples or diskette/CD-ROM images) from this redbooks site.

  Redpieces are redbooks in progress; not all redbooks become redpieces and sometimes just a few chapters will be published this way. The intent is to get the information out much quicker than the formal publishing process allows.

- **E-mail Orders**

  Send orders via e-mail including information from the redbook order form to:

	IBMMAIL	**Internet**
In United States:	usib6fpl at ibmmail	usib6fpl@ibmmail.com
In Canada:	caibmbkz at ibmmail	lmannix@vnet.ibm.com
Outside North America:	dkibmbsh at ibmmail	bookshop@dk.ibm.com

- **Telephone Orders**

United States (toll free)	1-800-879-2755	
Canada (toll free)	1-800-IBM-4YOU	
Outside North America	(long distance charges apply)	
(+45) 4810-1320 - Danish	(+45) 4810-1220 - French	(+45) 4810-1270 - Norwegian
(+45) 4810-1420 - Dutch	(+45) 4810-1020 - German	(+45) 4810-1120 - Spanish
(+45) 4810-1540 - English	(+45) 4810-1620 - Italian	(+45) 4810-1170 - Swedish
(+45) 4810-1670 - Finnish		

This information was current at the time of publication, but is continually subject to change. The latest information for customers may be found at http://www.redbooks.ibm.com/ and for IBM employees at http://w3.itso.ibm.com/.

IBM Intranet for Employees

IBM employees may register for information on workshops, residencies, and redbooks by accessing the IBM Intranet Web site at http://w3.itso.ibm.com/ and clicking the ITSO Mailing List button. Look in the Materials repository for workshops, presentations, papers, and Web pages developed and written by the ITSO technical professionals; click the Additional Materials button. Employees may also view redbook, residency and workshop announcements at http://inews.ibm.com/.

IBM Redbook Fax Order Form

Fax your redbook orders to:

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	(+45) 48 14 2207 (long distance charge)

Please send me the following:

Title	Order Number	Quantity

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

- Invoice to customer number
- Credit card number

Credit card expiration date Card issued to Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

List of Abbreviations

AD	application development
BLL	Base Locator for Linkage
CICS	Customer Information Control System
CCCA	COBOL and CICS Command Level Conversion Aid
DASD	direct access storage device
DCB	data set control block
DFSORT	Data Facility Sort
DIF	date identification file
ESA	Enterprise Systems Architecture
IBM	International Business Machines Corporation
ISPF	Interactive System Productivity Facility
ITSO	International Technical Support Organization
JCL	job control language
LCP	language conversion program
MA2000	Maintenance 2000
MB	megabyte
MLE	Millennium Language Extensions
MVS	Multiple Virtual Storage
PDF	Program Development Facility
PDS	Partitioned Data Set
TSO	Time Sharing Option
URL	universal resource locator
VA2000	VisualAge 2000

Index

N

O

P

R

S

ITSO Redbook Evaluation

VisualAge 2000: Facilitating Find and Fix in an OS/390 Environment
SG24-5253-00

Your feedback is very important to help us maintain the quality of ITSO redbooks. **Please complete this questionnaire and Fax it to: USA International Access Code + 1 914 432 8264 or:**

- Use the online evaluation form found at http://www.redbooks.ibm.com
- Send your comments in an Internet note to redbook@us.ibm.com

Which of the following best describes you?
__Customer __Business Partner __Solution Developer __IBM employee
__None of the above

Please rate your overall satisfaction with this book using the scale:
(1 = very good, 2 = good, 3 = average, 4 = poor, 5 = very poor)

Overall Satisfaction _____

Please answer the following questions:

Was this redbook published in time for your needs? Yes____ No____

If no, please explain:

_____

_____

_____

_____

What other redbooks would you like to see published?

_____

_____

_____

Comments/Suggestions: **(THANK YOU FOR YOUR FEEDBACK!)**

_____

_____

_____

_____

_____

SG24-5253-00
Printed in the U.S.A.