Fourth Edition

GC28-1251-03

The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation

IBM

Fourth Edition

GC28-1251-03

The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation

Note

Before using this information and the product it supports, be sure to read the general information under "Notices" on page xv.

Fourth Edition, July 1996

This is a major revision of, and obsoletes, GC28-1251-02.

Order publications through your IBM representative or the IBM branch office serving your locality. Publications are not stocked at the address below.

IBM welcomes your comments. A form for readers' comments may be provided at the back of this publication, or you may address your comments to the following address:

International Business Machines Corporation
Department 55JA, Mail Station P384
522 South Road
Poughkeepsie, NY 12601-5400
United States of America

FAX (United States & Canada): 1+914+432-9405
FAX (Other Countries):
Your International Access Code +1+914+432-9405

IBMLink (United States customers only): KGNVMC(MHVRCFS)
IBM Mail Exchange: USIB6TC9 at IBMMAIL
Internet e-mail: mhvrcfs@vnet.ibm.com
World Wide Web: http://www.s390.ibm.com/

If you would like a reply, be sure to include your name, address, telephone number, or FAX number.

Make sure to include the following in your comment or note:

- Title and order number of this book
- Page number or topic related to your comment

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

Contents

Figures

About This Book

This book provides information to help you:

- Understand the cause and scope of using dates represented by two-digit years
- Identify problems with programs using 2-digit data
- Plan for your installation′s migration to a Year2000-ready environment
- Determine the best technique(s) for reformatting your year-date notation
- Determine the best technique(s) for testing your Year2000 changes
- Select and use IBM and Solution Developer tools appropriate for your needs

The official copy of the book is the on-line version. You can access this document through an anonymous ftp whose URL is:

ftp://lscftp.pok.ibm.com/pub/year2000/y2kpaper.**format_type**

where **format_type** can be one of the following print formats:

Print Format	***format_type***
ASCII text	txt
Self-extracting zip file of ASCII text	exe
Browseable BookManager	book
PostScript	ps
UNIX compressed	ps.z

Refer to "Electronic (Internet/World-Wide Web) Documentation" on page C-8 for further information and related IBM on-line documentation.

Who Should Use This Book

The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation is directed, but not limited to, the following audience:

- Executives requiring an overview of the Year2000 challenge
- Installation managers and system and application programmers needing information to plan, install, modify and test their applications and systems to become Year2000-ready.
- Users of IBM and non-IBM products and services
- Solution Developers that have software products on IBM and non-IBM platforms
- Year2000 focal points and marketing representatives that provide a source of information for both internal and external queries related to the Year2000 phenomenon
- Independent consulting groups
- Programmers who generate and maintain 'in-house' application code

How to Use This Book

The following categories are designed to help you determine which chapters are most suited to your specific needs.

Of General Interest to Everyone

To help your overall understanding of the challenge, you should:

- Read Chapter 1, "The Year 2000 - A Transition" on page 1-1
- Scan Appendix C, "Bibliography" on page C-1 for additional readings related to topics that range from an overview of the Year2000 challenge to management issues to in-depth technical papers and discussions through both written and electronic (Internet) media.

Executive and Senior-Level Management

You should read "Executive Summary" on page xii to learn about the challenges facing the Information Technology (IT) community and the challenge facing **your** information system (IS) staff.

IS Managers

You will find it valuable to read the entire guide to learn more about which actions you need to take in support of your system and application programming staff.

Minimally, however, you should read the following:

- To obtain a high-level overview of the Year2000 challenge, read:
  - "Executive Summary" on page xii
  - Chapter 1, "The Year 2000 - A Transition" on page 1-1.
- For project planning information and recommendations, read Chapter 2, "Planning to Resolve Your Year2000 Exposures" on page 2-1.
- For migration information and recommendations, read Chapter 6, "Migration Consideration for Year2000 Transition" on page 6-1.
- For a list of IBM and Solution Developer tools available to assist in identifying, coding, and testing Year2000 changes, read Chapter 7, "Tool Categories and Available Tools to Ease Year2000 Changes" on page 7-1.

System Programmers

- For an overview, you should read Chapter 1, "The Year 2000 - A Transition" on page 1-1.
- To determine the best technique(s) for reformatting year-date notation, you should read Chapter 4, "Reformatting Year-Date Notation" on page 4-1.
- For migration information and recommendations, read Chapter 6, "Migration Consideration for Year2000 Transition" on page 6-1.
- To assist you in testing your Year2000 modifications, you should read:
  - Chapter 5, "Testing Techniques for Year2000 Changes" on page 5-1
  - Chapter 7, "Tool Categories and Available Tools to Ease Year2000 Changes" on page 7-1.

Application Programmers

- For an overview, you should read Chapter 1, “The Year 2000 - A Transition” on page 1-1.
- To understand how to identify the potential exposures caused by using 2-digit-year representations of dates, read Chapter 3, “Identifying 2-Digit-Year Exposures” on page 3-1.
- To determine the best technique(s) for reformatting year-date notation, you should read Chapter 4, “Reformatting Year-Date Notation” on page 4-1.
- To assist in your testing of your Year2000 modifications, you should read:
  - Chapter 5, “Testing Techniques for Year2000 Changes” on page 5-1.
  - Chapter 7, “Tool Categories and Available Tools to Ease Year2000 Changes” on page 7-1.

Executive Summary

A phenomenon exists in the Information Technology (IT) industry because historically many computer programs make use of dates represented by only two digits (for example, 95 rather than 1995). However common this practice might be, it causes programs (both system and application) that perform arithmetic operations, comparisons, or sorting of date fields to yield incorrect results when working with years outside the range of 1900-1999. IBM refers to this phenomenon as the **Year2000 Challenge**.

Does this Really Mean Me?

The scope of the Year2000 challenge spans the entire IT industry. A data mismatch can exist in any level of hardware or software from microcode to application programs, in files and databases, and is present on **all** platforms (IBM and non-IBM). In recent years, the IT trade press has given ever greater attention to this phenomenon with increasingly ominous article titles.

However dramatic all this may sound, consider the following scenarios to help put the phenomenon and its business ramifications into perspective. Imagine if in the first quarter of the year 2000 your company cannot process its 1999 end-of-year billing or end-of-year payroll properly; your corporate credit card holders are refused most transactions because their accounts appear delinquent; your 1999 year-end profit data cannot be calculated properly; and your utility companies cut off their services due to your apparent late bill payments. Similarly, your household and personal financial situation could encounter a similar dilemma if your creditors do not also strive to meet this challenge.

Although referred to as the Year2000 challenge and often as the 'Year2000 Problem', this is really a 2-digit-year problem. Your IS organization needs to plan for, and address, the date changes well in advance of 1 January 2000. This is not only a future challenge; it has existed in the banking industry since as early as 1970 when application programs encountered problems with amortization and interest table calculations for the standard 30-year mortgage. Consider also, the standard 5-year automobile loan, a 15-year mortgage, a long-term insurance policy, a data base that retains birth dates (which includes an ever increasing set of dates over 100 years). Even before 1 January 2000, your computing environment might not function as expected because your systems and/or application programs will not process dates later than 31 December 1999 properly.

But I′ve Been Told...

There are many misconceptions associated with the cause of the Year2000 challenge. These include:

- This is a problem that occurs only after 1999.

  As noted above, expect to experience problems well in advance of the year 2000. Application programs manipulating such data are likely to produce incorrect results today.

- This is a hardware problem rather than the problem for your IS organization.

  To the contrary, this problem comes mostly from application programs and data using two digits for year representations. These programs have been purchased from Solution Developers, written in-house by your system and application programmers, leased from various software vendors, or shared among the IS community. Frugality was the rule when such programs were first written and every byte of saved storage was important, and human nature seems to dictate that writing and keyboard entry of two digits is more practical than four digits. Saving even a small amount of space in a data base was more important in the 1960′s and 1970′s than it is today, and most programmers never expected such code to still be in use 20 or 30 years later. This short-sighted programming practice has only recently surfaced as a problem, and only a rare program provided for four digits for the Year2000.

- This is a problem that only occurs on mainframe systems and/or legacy applications.

  Any system or program (large or small) can be affected if it uses two digits for year representation. In fact, such problems have been encountered in recently developed desktop software as well.

What′s My Role and What Can I Expect?

For many institutions, you can expect the initial size of this Year2000 work effort to appear overwhelming and the overall project to appear enormous, as it should. This is not a trivial project. Expect the need for your sponsorship to dedicate personnel, dedicate computer resources, approve the requisition of potential Solution Developer tools and/or service support, and a budget to fund it all.

The solutions provided here are not particularly difficult to implement, and the technical solutions are not complex when viewed on a program-by-program basis. It is the overall project size and the program/data inter-relationships that introduces the true project complexity. Expect the need to appoint a cross-department and cross-divisional focal point to manage it all.

What Is IBM Doing to Assist Me?

The project is manageable within the available time-frame and support is currently available from both IBM and many Solution Developers. IBM is committed to address the Year2000 Challenge within its own product offerings, to deliver Year2000-ready software, and to assist you in your efforts.

The IBM document (*The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation*), for which this summary is provided, is the result of an IBM multi-divisional effort to articulate the overall IBM support with respect to the impending Year2000 Transition. It is intended to provide the Information Systems (IS) community with planning, methodology, and guidance information needed to convert their application systems to correctly manipulate dates outside the range of 1900-1999.

The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation is available through your IBM representative or in softcopy form through the Internet. That document provides your IS personnel both generic and IBM-specific information on:

- How to plan a Year2000 transition
- How to detect and remove Year2000 exposures
- How to test and migrate to a revised Year2000-ready system
- Where to find further information (An extensive bibliography lists many of the most prominent publications and trade press articles concerning the Year2000 challenge.)

This document also provides information that addresses:

- **Multiple Platform Solutions** (including IBM product-specific information for):
  - All IBM and non-IBM IS platforms
  - All IBM operating systems
  - IBM programming languages
- **Current IS Tools** (from IBM and non-IBM software vendors):

  Tools that are available today to help your IS personnel identify potential exposures in your applications, assist their efforts in the categorization of potential exposures, and assist the testing and transition to a Year2000-ready application environment.
- **Consulting Services:** IBM consulting service offerings that provide skills and expertise for the understanding and/or the solution of your applications′ Year2000 transition.

Although this paper is designed mainly to provide in-depth information to your IS technical staff (that is, your applications and systems programmers), the support that it demands must come from you. This will not be a small undertaking for your IS staff within any aspect of your IS environment. If your business relies on computer processing, and that processing includes date manipulation, your business may be at risk. This paper provides direction for your organization to begin a pro-active response to this challenge.

So, What′s the Bottom Line?

Your immediate attention to this effort, and a directive to your IS organization to plan for and solve your Year2000 challenge, is critical. When you face your stockholders who will ask, “Now that you spent xxx dollars on this project, what do we have now that we didn′t have in 1995?,” simply respond: “**We still have a viable business**, one that accepts billing, correctly computes interest and invoices, and calculates our stock dividends. Not all our competition can make that same claim!”

Notices

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 product, program, or service that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, programs, or services except those expressly designated by IBM, are the user′s responsibility.

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:

IBM Director of Licensing
IBM Corporation
500 Columbus Avenue
Thornwood, New York 10594
USA

Trademarks

IBM Trademarks

The following terms are trademarks of the IBM Corporation in the United States and/or other countries:

- ACF/VTAM
- AD/Cycle
- ADSTAR
- AFP
- AIX
- AIXwindows
- AIX/6000
- AnyNet
- APL2
- AS/400
- AT
- BookManager
- BookMaster
- C Set ++
- CallCoordinator/2
- CallPath
- CallPath SwitchServer/2
- CallPath/400
- CAMkit
- CICS
- CICS/ESA
- CICSPlex
- CICS/MVS
- CICS/VSE
- COBOL/370
- COBOL/400
- CUA
- C/370
- C/400
- DATABASE 2
- DataHub
- DataPropagator
- DB2
- DB2/400
- DFSMS
- DFSMSdfp
- DFSMSdss
- DFSMSdhm
- DFSMS/MVS
- DFSMSrmm
- DFSMS/VM
- DFSORT
- DirectTalk/2
- DirectTalk/6000
- EOCF/2
- ESCON

- Facsimile Support/400
- FASTService
- First Failure Support Technology (FFST)
- FlowMark
- GDDM
- geoManager
- geoGPG/6000
- HACMP/6000
- Hardware Configuration Definition
- IBM
- IBMLink
- ImagePlus
- IMS
- IMS/ESA
- InfoCrafter
- InfoExplorer
- Integrated Language Environment
- ISPF
- ISSC
- LANDP
- Language Environment
- LoadLeveler
- MERVA
- MQSeries
- MVS/DFP
- MVS/ESA
- NetDoor
- NetView
- NTuneMGN
- NTuneNCP
- OfficeVision
- OfficeVision/MVS
- OfficeVision/400
- OPC
- OpenEdition
- Operating System/400
- OS/2
- OS/390
- OS/400
- PowerPC
- PROFS
- ProductManager
- PSF
- QMF
- RACF
- ReDiscovery
- Resource Measurement Facility
- RISC System/6000
- RMF
- RMONitor
- RPG/4000
- RS/6000
- SMARTdata UTILITIES
- SOMobjects
- SP
- SQL/DS
- SwitchServer/2
- Sysplex Timer
- SystemView
- System/38
- S/390
- TRANSFORMATION 2000
- Trouble Ticket
- TURBOWAYS
- Ultimedia
- YEAR 2000
- VisualGen
- VisualLift
- VM/ESA
- VSE/ESA

Lotus Trademarks

The following terms are trademarks of the Lotus Development Corp. in the United States and/or other countries:

- Approach
- cc:Mail
- Freelance Graphics
- Lotus
- Notes
- Organizer
- ScreenCAM
- SmartCenter
- Soft-Switch
- Word Pro
- 1-2-3

Non-IBM Trademarks

The following terms are trademarks of other companies as follows:

Trademark	Company
ADAMS/400	Namtrig Incorporated
C/QUE	Systemware
Canada's National Newspaper	Globe and Mail
COBOL Analyst	SEEC, Inc.
Change Action	Mazda Computer Corp.
DateServer	Computer Software Corp.
DATE/2000	DATE/2000
Globe and Mail	Globe and Mail
BYPASS2000	HAL Informatica slr
IntelliCONSOLE	IntelliWare Systems, Inc.
IntelliRESOURCE	IntelliWare Systems, Inc.
JHS	Systemware
MPS	Systemware
The OLR API	Data Base Architects, Inc.
The OLR System	Data Base Architects, Inc.
OnLine Help	Data Base Architects, Inc.
OnLine Note pad	Data Base Architects, Inc.
OnLine Reference	Data Base Architects, Inc.
Portal 2000	DTS Software, Inc.
RECY2000	Informission Group, Inc.
REFINE Language Tools	Reasoning Systems
Challenge 2000	Micro Focus
Software Refinery	Reasoning Systems
SuperWylbur	SuperWylbur Systems, Inc.
SystemVision	ADPAC Corp.
TransCentury	TransCentury Data Systems
Version Merger	Princeton Softech
X/PTR	Systemware

Summary of Changes

Summary of Changes
for GC28-1251-03

This book contains information previously presented in *The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation*, GC28-1251-02.

The following summarizes the changes to that information.

New Information

- “Hardware” on page A-26 lists IBM network hardware products that are or scheduled to be Year2000-ready by year-end 1996.
- Chapter 8, “IBM Consulting and Services” on page 8-1 has been expanded to include the following new sections:
  - “Test Support Services” on page 8-2 describes the test services available from ISSC′s TRANSFORMATION 2000.
  - “IBM Availability Services” on page 8-4 describes some of IBM′s availability services which can be used to assist you with the task of product migration.

Changed Information

- “How to Change Date and Time for Testing” on page 5-4 has been updated to provide more specific information concerning Year2000 testing in an environment in which you have reset (advanced) the system clock.
- Appendix A, “IBM Year2000-Ready Key Program Products and Hardware” on page A-1 that lists those IBM products scheduled to be Year2000 ready by year-end 1996 has been expanded.
- Appendix C, “Bibliography” on page C-1 has been updated to reflect current press releases, magazine articles, and currently available electronic documentation related to the Year2000 challenge.

Summary of Changes
for GC28-1251-02

This book contains information previously presented in *The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation*, GC28-1251-01.

The following summarizes the changes to that information.

New Information

- Appendix B, "Year2000-Ready Solution Developer Products" on page B-1 lists non-IBM, Year2000-ready products Solution Developers have requested IBM include in this document.

Changed Information

- Appendix A, "IBM Year2000-Ready Key Program Products and Hardware" on page A-1 that lists those IBM products scheduled to be Year2000 ready by year-end 1996 has been expanded.
- Appendix C, "Bibliography" on page C-1 has been updated to reflect current press releases, magazine articles, and currently available electronic documentation related to the Year2000 challenge.

Summary of Changes
for GC28-1251-01

This book contains information previously presented in *The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation*, GC28-1251-00.

The following summarizes the changes to that information.

New Information

- An additional encoding scheme, the use of the CYY and CCYY is added to "Solution #3: A 2-Digit Encoding/Compression Scheme" on page 4-6 in Chapter 4, "Reformatting Year-Date Notation."
- DFSORT/MVS V1R13 will be enhanced to provide the ability to sort, merge, and transform 2-digit year data using a sliding window technique. "DFSORT" on page 7-23 in Chapter 7, "Tool Categories and Available Tools to Ease Year2000 Changes" provides a detailed description of the new function and the enhanced control statements used to implement them.
- Appendix C, "Bibliography" on page C-1 has been updated to reflect current press releases, magazine articles, and currently available electronic documentation related to the Year2000 challenge.
- An authorization form, "Year2000-Ready Solution Developer Product and Tool Authorization" is provided in the back of this document for Solution Developers to request that IBM list their Solution Developer product and tool offering in future updates to this or related documents.

Changed Information

- Appendix A, "IBM Year2000-Ready Key Program Products and Hardware" on page A-1 that lists those IBM products scheduled to be year2000 ready by year-end 1996 has been expanded.
- Figure 7-5 on page 7-71 that lists Solution Developer products available to assist your year2000 resolution has been expanded.

This book includes terminology, maintenance, and editorial changes. Technical changes or additions to the text and illustrations are indicated by a vertical line to the left of the change.

Chapter 1. The Year 2000 - A Transition

Although 1 January 2000 is still several years away, you need to plan for and address the 1999 to 2000 date change well in advance of that date. Even before we reach 1 January 2000, your computing environment might not work as expected because your systems and/or application programs will not process dates later than 31 December 1999 properly. However dramatic the following sounds, consider these scenarios to help put the problem into perspective. Imagine: all stock exchanges close down indefinitely because of invalid transactions; credit card companies refuse most transactions because cardholders appear delinquent on their payments; mortgage companies are besieged by angry borrowers who have just received delinquency notices in error and been charged extra interest; and utility companies cut off service to many customers due to apparent late bill payments.

The potential exposure mostly comes from, but is not limited to, the use of a 2-digit-year (YY) format, instead of a 4-digit (YYYY) format, for year representation within programs, files, databases, and processes. For example, the year 1996 is represented as '96', the year 1999 as '99', and so on. Thus, January 1, 2000 is represented as 01/01/00 (if using MMDDYY format in your data) and might be interpreted as January 1, 1900. This causes programs that perform arithmetic operations, comparisons, or sorting of date fields to yield incorrect results when manipulating year-date data of 2000 and beyond. The potential exposures that may be encountered, and their variations (listed in Section "Year2000 Exposure Classification" on page 1-2) have been referred to by IBM as the Year2000 challenge.

The scope of the Year2000 challenge spans the entire Information Technology industry. A data mismatch can exist in any level of hardware or software, from microcode to applications, in files and databases, and is present on **all** (IBM and non-IBM) Information System (IS) platforms.

Some general **misconceptions** of the Year2000 challenge include:

1. This is a problem that occurs only when/after the century rolls over.

   The challenges of handling the Year2000 changes may occur well before the year 2000 arrives. Typically, forecasting applications that deal with future dates will encounter problems well in advance of the year 2000. In fact, applications are already facing the problem; for example, the financial applications that deal with life insurance or bond policies that have expiration dates that go beyond the year 2000. In such an application, the programs need to handle the dates beyond the year 2000 when checking policy expiration.

2. This is a hardware clock problem that should be resolved by the computer vendors.

   To the contrary, this problem comes mostly from application programs and data using two digits for year representations even though the hardware clock and/or system timer services can provide a 4-digit-year format.

3. This is a problem that only occurs on mainframe systems and/or legacy applications.

   Any system or program can be affected if it uses only two digits for year representation: any file, database, log with 2-digit-year fields, and any data entry, query, update and output processing that employs 2-digit-year fields. It can also affect programs, systems, databases, and other functions and processing that receive input from these programs and data.

Year2000 Exposure Classification

The Year2000 phenomenon is further compounded by the numerous variations used to express year and date notation in data and the mathematical calculations performed on those date notations. These variations can be classified as the following exposure types:

- Incorrect century[1]

  Problems can occur when the first two digits in a year are assumed to be 19 and:

  – Ignored during data entry and update, or
  – Hard-coded for output.

- Dates used as a special value:

  Special values of the last two digits in a year might be used for special purpose, for example 99, 365/99, or 12/31/99 might be used to indicate "no expiration date" or 00 to indicate an "unknown year."

- Incorrect field format determination:

  Some existing programs determine the date-time format (that is, MMDDYY, DDMMYY, YYMMDD) by testing an appropriate part of the date field. For example, checking if the first two characters of the date field are values within an acceptable month, day, or year range (such as 1-12, 1-31, or ≥32).

- Arithmetic calculation:

  The arithmetic calculations that operate on dates with 2-digit year representation might have potential exposures. For example, a person with a birthday of 10 November 1961 will be considered to be −61 years old rather than 39 years old on 10 November 2000 if the years 1961 and 2000 are represented by 61 and 00, respectively. Generally, a program that is not expecting a signed value, will ignore the sign; thus, in this example, the −61 might be interpreted as the absolute value 61. Not only is the value still incorrect, it is even less detectable than the incorrect −61. Further, if such incorrect data is then stored in a data base, it is considered a *data integrity* exposure.

- Sequence:

  When only two digits are used to represent a year, programs that collate year data will sort that data out of sequence in some cases. For example, the year 2000 (if represented as 00) will be ordered prior to the year 1999 (if represented as 99).

- Data integrity:

[1] Although IBM recognizes that the 21st century begins at 0000 hours, 1 January 2001, for purposes of this document, we are defining the 20th—21st century boundary to be between 2400 hours, 31 December 1999 and 0000 hours, 1 January 2000. This allows a discussion of the 21st century to include all dates with a 20yy format inclusive of the year 2000. Hence, the year 2100 is likewise relegated to the 22nd century.

In programs where historical dates are used, for example all events occurring in 1800, 1900, and 2000 are not distinguishable when the years are represented by only 2 digits.

- Leap year calculation:

  A specific year is a leap year if it is either evenly divisible by 400 **or** evenly divisible by 4 and not evenly divisible by 100. For example, the year 1900 was not a leap year but the year 2000 is a leap year. Some potential exposures caused by the identification of the year 2000 as a non-leap year are:

  – Day-in-year calculations. The year 2000 has 366 days, not 365.

  – Day-of-the-week calculations. 28 February 2000 is a Monday and 1 March 1 is a Wednesday, not a Tuesday which is February 29, 2000.

  – Week-of-the-year calculations. The 11th week of the year 2000 is 5 through 11 March, not 6 through 12 March.

Scope of Year2000 Transition

The computing environment today is mostly constructed and integrated around the theme of heterogeneous, open, distributed, and client-server computing. The computers from different vendors are networked together to provide an integrated and distributed computing environment. Even though there are many different kinds of platforms in such an environment, they all tend to have the following common components:

Computer Hardware

- CPUs
- storage devices
- input/output devices
- communication devices
- and so on

Computer Software

- Microcode
- Operating systems
- Middleware, such as database management systems, telecommunication control programs, transaction monitoring programs, and so on.
- Programming languages, such as COBOL, Fortran, PASCAL, PL/I, C, assembler
- Solution Developers and ‘3rd party’ software providers
- Application programs

People

Those who develop and maintain the system, those who operate the system, those who provide its input and use its output, those who manage and ensure the quality of the system, and those who provide manual processing activities in a system. These people include:

- End users
- Management

- Auditors, quality assurance people
- System analysts
- System designers
- Programmers
- Operations personnel

Data

The information that the system records and processes

Procedures

Formal policies and instructions for operating the system

Figure 1-1 illustrates these components in an organization′s networked computing environment. At each computing node (viewing the figure vertically), information systems (IS) personnel follow procedures to access the data by means of computer software and hardware. The computing nodes (viewing the figure horizontally) are networked together by communication hardware and software to communicate to other computing nodes internal and/or external to the organization. The data is thereby accessible anywhere and anytime while the computing nodes are connected.

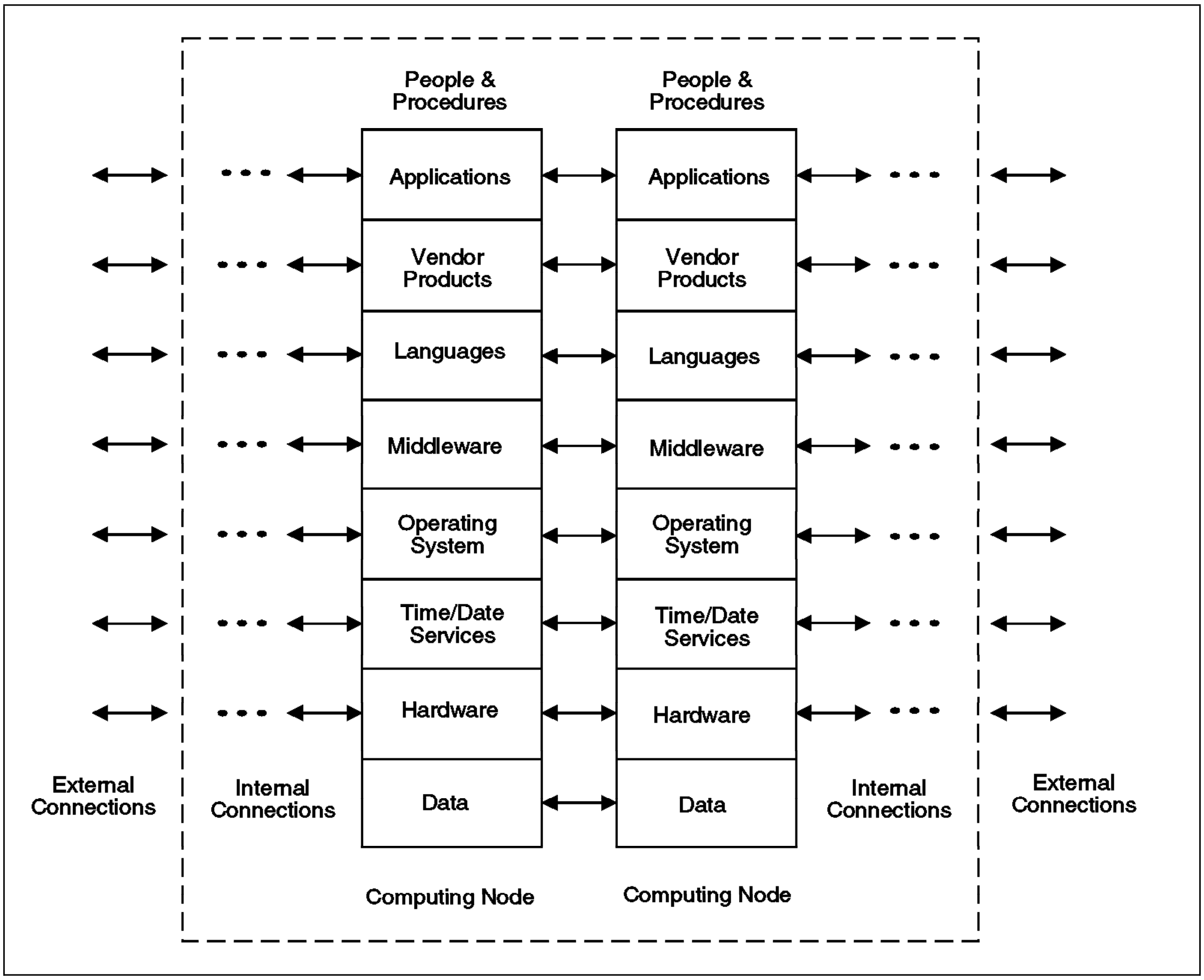

Figure 1-1. A Corporate Networked Computing Environment

The Year2000 phenomenon potentially has both vertical and horizontal impact on such a computing environment as pictured above. Vertically, the Year2000 phenomenon can originate from or affect the key components of the computing systems, that is, hardware, software, people, data, and procedures. Horizontally, the phenomenon can propagate as the contaminated data flows to other computing systems inside and outside of the organization. In short, the scope spans the entire information technology industry.

Although this is a complicated and far-reaching problem, it is not a technically difficult problem to resolve when viewed on a module-by-module or routine-by-routine basis. The degree of complexity is directly related to the inter-relationships between routines and programs and the data passed between them. **This is not a trivial programming exercise.** Your realization of a Year2000-ready system demands immediate management commitment, dedicated resources (personnel and hardware), strategic planning, rapid development, acceptance testing, a well-planned migration, and an adequate budget. The following chapters address these issues and provide basic approaches for some specific issues.

Chapter 2. Planning to Resolve Your Year2000 Exposures

To successfully address the challenges present within your computer system, you need to obtain your management′s understanding, support, and an uncompromising commitment to provide resources to meet your needs. Expect the need for a knowledgeable executive sponsor to address budgetary, personnel and hardware resource requirements, cross-department and cross-divisional requirements, and overall scheduling and project management. It is imperative that this effort begin with and be managed through a central focal point responsible for critical project aspects such as overall scheduling, coordinating, and setting a consistent methodology through all project phases.

This chapter and the following focus on those project phases: planning, exposure identification, exposure elimination, testing, migration, and the selection of available tools. The larger your computing environment, the more diverse its software, the more decentralized its physical environment, the greater control you must exercise, and the greater the communication that must exist across the individual projects.

The time to begin both planning and your Year2000 transition is now. Consider the following:

- Getting requirements and design changes into the development cycle takes time. The review and modification of the application takes time. Securing resources and skills takes time. Handling the problem in real time will disrupt your customer services, and the business impact will be significant.
- Most organizations are already short-handed when addressing their current workload and the challenges they possess. Therefore, with this additional effort, perform a risk assessment and identify what is critical to the success of your business and determine and prioritize those work items.
- A significant amount of code rework may be required to complete your Year2000 transition. It is not merely a problem that can be fixed by expanding the data fields. You must make changes to your data dictionary, data bases, files, programs, and so on.
- Within some institutions, programs are already producing incorrect output, and many organizations that aren′t today, can expect problems tomorrow. For example, insurance companies, when calculating rates for persons born in ′94, might find themselves assessing 101-year-old rates on a new-born scale, or potentially assessing a new-born at the same rate as a centenarian.

For some organizations, it may not be mandatory to act now, but it could save code redesign, especially data rework. Each day that passes prior to your year 2000 transition allows more data to be added to your data bases and the potential for additional routines and programs that are added to your system. Also, expertise that is present today will no longer be available later.

Finally, there have been projections that the availability of consulting and outsourcing services to meet Year2000 transition needs will become increasingly limited as we approach the year 2000. For organizations with applications that handle future dates or those with thousands of application programs, the consequence of delaying the resolution of the problem could be disastrous to their continued future success.

Planning Considerations

You need to plan for changes across all aspects of your IS environment. The following task categories might prove useful when approaching this task:

1. Identify and communicate the organization goal

   The goal is to have the function and operation of an organization Year2000-ready before any disruption caused by 2-digit-year data occurs. **Year2000-Ready** means that products, programs, files, databases, and processes have or produce no logical or arithmetic inconsistencies when dealing with dates beyond 1999.

2. Identify the deliverables and associated schedules for the following:

   - Hardware
   - Software
   - Documentation
   - Training
   - Maintenance
   - Operations
   - Administration
   - Acceptance criteria for all deliverables

3. Analyze job assignments

   For each task,

   - Identify the responsible person or organization, for example
     - Customers
     - Management
       - Chief Executive Officer
       - Chief Financial Officer
       - Chief Information Officer
       - Software Development Managers
       - Operation/Administration Managers
       - Budget/Finance Managers
       - others
     - Computer vendors
     - Solution Developers
     - IT outsourcing vendors
     - Consulting/Integration services providers
     - System analysts
     - System designers
     - System/Application Programmers
     - Operations personnel
     - End Users
     - Auditors, quality assurance people
   - Measure the estimated completion time
   - Identify precedences/dependencies
   - Identify resources and skills
   - Identify critical path schedule
   - Measure efforts
   - Measure costs
   - Identify technical factors
   - Analyze potential benefits
     - Return on investment
     - Achievement of business goals
     - Potential quality and acceptance of the approach

  - Your business keeps running
- Analyze risk factors
  - Complexity of the task
  - Resource/time constraints
  - Length of project
  - Critical development skills

4. Measure the criticality of each task and prioritize

   Evaluate and determine how critical the functions of each entity are to the business success of the organization, and prioritize the sequence of providing Year2000-ready solutions. An entity could be an individual, a department, a division, a business unit, customers, vendors, and so on, that are involved in the operations of the organization. The factors contributing to how critical you consider a task to be might include pressure of demand from end users for Year2000-ready systems, legal issues, financial issues, or political issues.

   Impact to Business

   To determine the impact to your business, consider including these tasks:

   - Critical to the operation of the business (such as legal compliance)
   - Critical to the uninterrupted operation of the business (such as payroll)
   - Required to support the business (such as management and financial reports)
   - Required to support the business; however, the importance and timetable for the activity is lower than an item above (such as regular scheduled reports)
   - Desirable, but not absolutely required to support the business.

   Impact to Operations

   Once classified by task, then determine impact severity. For example, you could use categories such as:

   Fatal Operations will ABEND or terminate

   Critical Operations will produce an incorrect result. (For example, expiration dates for food or pharmaceutical products are calculated as over 100 years old, not one or several days old.)

   Marginal Minor inconvenience, annoyance, or irritation. (For example, inventory reports collate dates of 00 prior to 99.)

   Based on the impact to a particular process, evaluate the desirability of reworking a particular piece of code. Here is an opportunity for your IS management and your business strategists to affect overall business practice. Together these groups should consider possibilities such as:

   - Abandoning the business process
   - Combining the process with other processes
   - Replacing the process with a new state-of-the-art process

5. Establish a ‘critical event horizon’

   Business environments are unique. The initial date your institution will begin experiencing Year2000 problems is also unique. If you prepare business forecasts of a 3-year cycle, the fourth quarter of 1996 might be your critical event horizon. If you deal in automobile loans, 1995 might be your critical event horizon. It is likely to be a very rare institution that will not experience some form of Year2000 difficulty until 1999 or 2000.

6. Provide data administration

   - Identify the scope and responsibility of migrating the affected data

     – Exclusive. The affected data object is created and processed exclusively by this business area and is independent of any other business area. This could be at an individual, department, or the business area level with further decomposition and analysis.

     – Primary responsibility. The business area defines the affected data object and other business areas should use that definition or negotiate for its redefinition.

     – Secondary responsibility. The affected data object is defined and created by a different business area in the enterprise, and is distributed only within the scope of the enterprise. Each business area defines its own use of the object which is provided by the business area with the primary responsibility.

     – External exposures

       - The affected data object is defined and created by either this or a different business area in the enterprise, and is distributed beyond the scope of the enterprise.

       - Data is created outside your enterprise and then imported and used within it.

   - Determine formats of the data dictionary

   - Determine procedures for changing and entering data elements

   - Determine procedures for collaborative data sharing and use

7. Decide project technical and management approaches

   - Programming standards, conventions, and guidelines

   - Platform for application development

   - Hardware/software

   - Development methodology

   - Development and test procedures

   - Prototyping and parallel development

     – Commonly used in software development projects and should be applied wherever appropriate.

     – Apply a divide-and-conquer approach to partition the Year2000 project into smaller projects so that development and testing can proceed in parallel.

     – Parallel development can shorten the development cycle. This is extremely critical when dealing with time-sensitive projects such as this Year2000 project.

   - Process/data modeling

   - Data dictionary

   - Documentation structure, layout, and standards

   - Reviews and walk-throughs

- Quality assurance procedures
- Testing methodology
- Automated tools
- Migrations of and bridgings to existing Year2000-ready systems
- Estimated future costs of maintenance

8. Identify project constraints, interfaces, and dependencies.

   - End users/customers
     - Availability of test and other data
     - Availability of facilities and services
     - Responsibility for reviews
     - Responsibility for end user tests
     - Other actions.
   - Special contract negotiations
   - Outsourcing and consulting services
     - Justification for the outsourcing and consulting services
     - Obligations for the outsourcing and consulting services providers
   - Interfaces and dependencies with other projects
   - Supporting services and facilities required
   - Hardware and software to be used
   - Solution Developer-automated tools to be used
   - Risks and alternative solutions
   - Other assumptions.

9. Provide standards, guidelines, quality assurance, and review procedures

   - Year2000-ready standards for purchasing of hardware/software vendor products.
   - Year2000-ready system requirements on request for proposal for outsourcing and integration services providers.
   - Organizational Year2000-ready standard/guidelines/process for specification, design, development, and testing of new and existing software.
   - Year2000-ready checklists for potential exposures.
   - Standard for machine-human interface (Refer to "Guidelines" on page 4-12 for a list of standards.)
   - Procedures for submitting and processing proposed changes. The procedures should evaluate why change is needed, consequence of not making the change, and its effect on product, cost, and schedule

- Procedures for sign-offs and approvals. The procedures should solicit comments from knowledgeable and affected people about likely effects on product, documents, schedule, and costs.
- Procedures for future follow-up.

Inventory Your Software Portfolio

Once you have put your Year2000 plan in place, you can then begin the task of converting your software programs for Year2000 readiness.

The first step requires that you thoroughly understand your computing environment and compile an inventory of all the software programs within that environment. Such an inventory allows you to:

- analyze your portfolio for definition and movement of date-related data elements and the use of date-related calculations and manipulation
- identify and remove Year2000 exposures
- track and control changes to your portfolio to more easily monitor and prevent injecting new Year2000 exposures into your inventory while your Year2000 resolution work progresses
- test the new (Year2000-ready) version of the software programs in your portfolio.

Once you have completed the above activities, you are ready to migrate from your current computing environment to your Year2000-ready environment. The following chapters discuss these activities in detail and provide options and suggestions for your consideration.

Chapter 3. Identifying 2-Digit-Year Exposures

To identify the potential exposures caused by using 2-digit-year representations of dates, you first need to locate references to all date-related data.

Locating References

Locating date-related data and code is itself a major piece of the work effort that you must address. The most complete method starts with an inventory of every programming entity used in your IS center. Once compiled, you can review each program individually. Alternatively, use the following, more systematic, approaches to locate 2-digit-year data and date-related code. (Solution Developer products are available to assist this effort. Refer to "Solution Developer Tools" on page 7-71 for a list of those Solution Developers and their products.)

1. Review the following documents for direct or indirect references to date-related data and formats. Then trace these references back to the application source code to locate references in that code as well.
    - Requests for proposal
    - Statements of work
    - Planning documents that describe future IS needs
    - Existing studies about the current system
    - Software development standards and process documents
    - Software quality assurance requirements
    - System requirements specifications
    - System design specifications
    - Program specifications
    - User instructions and procedures
    - Data dictionaries
2. Review program information for date references to include, for example, date variables, date functions or routines, and character strings. Character strings might include the following in either your code or its comments:
    - AS-OF, ASOF
    - BEGIN, BEG, BGN
    - CCYY
    - CYYDDD, CYYDDMM, CYYMMDD
    - CURR, CURRENT
    - DATE, DAT, DTE, DT
    - DAY, DA, DD
    - DDMMYY, DDDYY
    - DIFFDATE
    - DOB
    - DOH
    - END
    - EXPIRE, EXP
    - JULIAN
    - MONTH, MON, MO, MMM
    - MDY, MMDDYY, MMYY
    - START
    - TERM
    - TIME
    - TIMESTAMP, TIME-STAMP
    - TIMEDATE
    - THISDATE
    - TOD, T-O-D
    - WEEK
    - WEEKDAY
    - YEAR, YR, YY
    - YMD, YYMMDD, YYDDD
    - and so on.
    - Data entry forms, screen display formats, report formats

- Definitions of data fields, records, structures, files, and databases
- Source code, computer program listings, cross-reference reports
- Command languages, for example, JCL, REXX, CLIST, EXEC, and CL
- Data indexes and catalogs, table sizes
- Data dictionaries
- Date/time service routines
- Sort routines

3. Use a test system.

   Install an isolated, non-production system with a duplicated image of your system and application software. In large systems, this could be an LPAR (logical partition of the mainframe), or for other platforms consider dedicating a separate machine, segregated (in either time or place) from any other system(s). This segregation is crucial to guarantee that you will avoid contamination due to system clock advancement or untested, and yet imperfect, modified code.

   - With a changed date/time setting

     Set the system date and time to a future date and time value after 1 January 2000. During such testing, be certain to use compatible data that is synchronized with the revised application software, that data crosses a 100-year boundary, and be certain to update your current operating procedures to reflect this new data requirement as well. For example, the 100-year window could be 1900-1999 or 1995-2094, or you could use both range types. The more varied are your window type(s), the more incorrect code you will uncover. This testing will help you identify many (but not all) Year2000 exposures.

   - With changed data

     If you only change date fields in a routine, you are not introducing new logic into that routine. Although the fields are increased from 2-digit to 4-digit fields, you only need to recompile the routine to generate those new field lengths. If you didn′t change the logic, you needn′t test that logic. Typical testing is appropriate in a separate test system using new data, but Year2000-specific testing is really unnecessary. In this testing, ensure that the changes in the operand lengths produce the expected results. If the results match those produced prior to your changes, then the application is successfully performing data calculations using the new 4-digit data rather than the previous 2-digit data, and you have met you migration criteria.

   Refer to Chapter 5, “Testing Techniques for Year2000 Changes” on page 5-1 for a more detailed discussion of testing techniques and issues.

4. Use a combination of the above approaches.

Tracing References Back to Their Source

For any potential exposure identified, identify all direct and indirect references of this exposure. You can do so by tracing the flow of the data to identify its immediate source and destination and then repeating the tracing process until all sources and destinations of all potential data exposures are identified.

Determining the Impact of 2-Digit-Year Data Fields

Once you have located date-related data fields by one or more of the above approaches, you can classify the use or reference of those fields into one of the following categories:

- No Impact
  - The program uses a 4-digit-year representation in all occurrences.
  - The program uses a 2-digit-year representation within a program, but does not have any internal exposures, nor does it externalize the 2-digit year in any way.
  - The program uses a 2-digit-year representation within a program, but does not have internal exposures. The 2-digit year **is** externalized, but can not be referenced (for example, for display-only) by another program.

  Note: Such output might be labelled as 'cosmetic only' and for interpretation by a human only, but this too might have its own set of ramifications.

  - A municipality that tracks school-age children will more frequently begin 'inviting' centenarians to enroll in kindergarten. If a printout of residents reads: Birthdate: 10/14/89, what will the clerk compiling the list of 6-year-old children assume?
  - Another type of exposure is externalized by 'display-only' dates that have been coordinated with a hard-coded '19' for the century. Such 2-digit exposures might exist for terminal display or special forms where the '19' is pre-printed. Therefore, be aware of, and consider, potential impact of date fields in all situations; these might not always be obvious exposures.
- Impact
  - The program uses a 2-digit-year representation within a program. It does not have internal exposures, but the 2-digit year is externalized and could be referenced by another program.
  - The program uses a 2-digit-year representation and has internal exposures.

Investigating How Other Software Entities Use the Data

You also need to investigate the ways data is shared among software entities. The greater the degree and scope of data sharing, the more global is your task and the more critical is the need to prevent the further propagation of and data 'contamination' by 2-digit-year data. Except possibly in a rather small IS environment, you cannot know how output is used by all other programs that might access it. Several factors that affect how data is shared or used among software entities follow. These factors and the types of data sharing provide some of the links making up this data 'web'.

Data Sharing

Three factors affect the sharing of data between modules:

1. The number of data items passed between modules. (The more data passed, the tighter the relationship.)
2. The amount of control data passed between modules. (The more control data passed, the tighter the relationship.)
3. The amount of global data elements shared between modules. (The more global data elements shared, the tighter the relationship.)

Several types of data sharing can occur between two modules in a program: Two modules can communicate:

- Through a variable or data structure (for example, array, table, or record) that is passed directly as a parameter between the two modules. The data is used for problem-related data processing not for program control purposes.
- Through a variable or data structure (for example, array, table, or record) that is passed directly as a parameter between the two modules. However, only part of the data in these composite data elements is needed in the call, that is, more data is passed than needed. A change in one of the data structures to accommodate a change in either the calling or called module can affect other modules as well.
- By passing data from one module to another to control the order of instruction execution. (For example, a control flag is set in one module and tested in a CASE or WHEN statement in another module).
- By passing data between modules through some mutually agreed upon location in a global data area (for example, FORTRAN COMMON and PL/I EXTERNAL features). A change in one module might then require changes in other modules sharing the same data area.
- By one module reaching into the internals of another module to get or deposit data or control its function. (For example, a branch from one module′s code into another module. A change in either module might require a thorough analysis of the internals of both modules to determine how to deal with the consequences of the change.)

If contaminated (2-digit year data) is shared among modules, you must identify the exposure caused by the data sharing on the receiving side of the transaction.

Once you have identified Year2000 exposures, apply the appropriate techniques (as provided in Chapter 4, "Reformatting Year-Date Notation" on page 4-1) to reformat these date and time representations.

Chapter 4. Reformatting Year-Date Notation

This chapter provides a number of techniques that you can employ to correct improper date notation and use. Because some techniques are appropriate only to unique situations, this section also lists the advantages, disadvantages, and IBM recommendations for their use.

When selecting a proposed Year2000 solution, evaluate the following factors:

1. What is the external impact due to incompatible date format changes?

   That is, what other programs or what output will be affected and to what extent will those programs require change if this solution is implemented for this particular program?

2. How current are the program modules that reference the date formats externalized by the exposures?

   That is, are there any plans to either eliminate or replace this particular program or routine, the programs that input to it, or those that receive or use its output?

3. What functions will be impaired due to Year2000 exposures?

   That is, will any mission-critical function within your company be compromised due to not reworking or replacing a particular program?

Solutions and Techniques

As you identify Year2000 exposures by the approaches described in Chapter 3, "Identifying 2-Digit-Year Exposures" on page 3-1, your next step is to rework the current program and data exposures to make your applications Year2000-ready. You can apply the following solutions to remove potential Year2000 exposures. Each solution is presented with an example technique to change the potential exposure. These suggested techniques require both program and data changes. Several solutions and techniques and their associated pros and cons follow:

Solution #1: Conversion to Full 4-Digit-Year Format

This solution is a 4-digit solution that externalizes a 4-digit-year format.

This approach requires changes to both the data and the programs by **converting all references and/or uses** of 2-digit-year format (YY) to 4-digit-year format (YYYY). It also requires that you convert all software programs that reference or use the updated data simultaneously, or use a 'bridging' mechanism to perform the conversion between old and new data and programs. Refer to "Bridge Programs Help Stage Format Conversions" on page 4-10 for more details on bridge programs. (You can accomplish this program and data conversion in steps.) Otherwise, you will immediately encounter data integrity problems caused by the inconsistency of date/time data formats.

To ease your migration, you might consider ignoring any non-impact (cosmetic) data fields in the YY format. A cosmetic date is one, that if externalized, is only interpreted by humans. Such occurrences might include the date on an output separator page or a display-only date on a screen in a panel-driven application.

Note: Be careful when selecting those situations that you decide to ignore and call cosmetic only. Be certain that they will not cause any data integrity exposures or ambiguity or are not accessed by any other program. Such instances of non-problem YY formats appear in a report header that shows the printing date of the report. The date is meant for human understanding only, not computer program manipulation. Consider the potential for future change. For example:

- Today′s reports might be written to a data set tomorrow
- Display-only dates today may prove useful as a collating value when archiving that output tomorrow to meet a new business or government standard.
- Even when viewed by a human, 2-digit dates can prove ambiguous if the data spans 100 years.

If you allow the end user to continue to input 2-digit dates for compatibility and ease of data entry, then the responsibility to translate that data into a full 4-digit date falls to you, the application or systems programmer. One possible solution is to apply a context-sensitive prompt to allow the user to select a century indicator. For example, allow all dates to be entered as 2-digit dates and automatically prefix those with the current century unless the date is a future date or historical date. What constitutes "future" or "historical" is your decision but could be any date other than today′s current day, week, month, year, and so on. Using this scheme, a future date in context of a loan maturity date could be set to 20yy, or a historical date automatically forces the user to select a century from a 'choose a century' (...16, 17, 18) prompt list.

Pros and Cons

Pros

1. Can provide 4-digit-year format. It is considered to be the **only** complete, permanent, and obvious solution.
2. Provides increased security against potential inappropriate decisions today if you do not selectively ignore 'cosmetic-only' situations.
3. Can ease your migration if you selectively ignore 'cosmetic-only' situations.

Cons

1. Need to convert the year data from 2-digit format to 4-digit format in all cases.
2. Requires that you relocate adjacent fields in the date field layout, and usually requires that you increase record lengths.
3. Inherent future risk in initial assessment that determined a particularly situation can be ignored as 'cosmetic only'.
4. Increased DASD space usage required due to data field expansion of data (consider including not only active but also archive data) and duplicate DASD space during conversion.
5. Might experience a performance impact due to increased time in processing and date access.
6. Some programming languages allow integer dates that are offset from a base date to be stored in files, data bases or passed as parameters between programs. Such integer dates provided by COBOL intrinsic functions, Language Environment callable

services, the CICS FORMATTIME command DAYCOUNT option, and other similar functions must conform to the standard YYYYMMDD format. This standard eliminates potential ambiguous data and errors due to each integer-date system using a unique starting date. Therefore, the potential for mixing incompatible integer dates when passed outside a single source module is extremely high and must be avoided.

Solution #2: Windowing Techniques

This is a 2-digit solution that externalizes either 2-digit or 4-digit-year formats. This approach requires changes to your programs only; no data changes are required.

CAUTION:
These approaches can be applied only to dates within a maximum 100-year period at any one time. This solution is considered temporary because there is no guarantee that in the future, your applications will not expand to process dates that are more than 100 years apart. Therefore, this approach always carries with it a potential future exposure. (For example, humans are living longer. Therefore data bases that include birthdays (medical, civil, insurance, and so on) and the applications that access that data are already at risk with many dates spanning 100+ years.)

Two types of **windowing** techniques have been defined: the fixed window technique and the sliding (rolling) window technique.

Fixed Window Technique

The **fixed window** technique uses a static 100-year interval that generally crosses a century boundary. This technique determines the century of a 2-digit year by comparing the 2-digit year against a window of 100 years. The user specifies the number of years in the past and future relative to a specific year within the 100-year interval.

Consider this specific example: if the years of date-related data of your application fall in the range of 1 January 1960 to 31 December 2059, you can use a 2-digit year to distinguish dates prior to the year 2000 from the year 2000 and beyond. If using the current system year of 1995, the number of years in the past and future are specified as 35 and 64, respectively. Program logic determines the century based on the following data checking. If the 2-digit year representation of a specific year is xy then if:

- $xy \geq 60$, then it is a 20th century date (19xy)
- Otherwise (that is, $xy \leq 59$), it is a 21st century date (20xy).

If, for example, you need to maintain a window of 35 past years and 64 future years, such that next year, 1996, your application can successfully deal with dates in the range 1961 through 2060, you need to adjust this program checking every year. The inherent future risk when employing this technique is obvious, and when compared to the sliding window technique is far less desirable.

Pros and Cons

Pros

1. No need to expand the 2-digit-year data to a 4-digit format.
2. Can provide 4-digit-year format for data reference.
3. Can distinguish years from different centuries using only 2-digit-year format (provided the years being processed are in the range of 100 years at any one time).
4. Can be useful if the particular program is being phased out, and a temporary solution is appropriate.

Cons

1. Potential exposures exist when/if the function of the software application needs to process years beyond the range of 100 years.
2. Expect a performance impact in direct proportion to the quantity of date processing the particular application handles due to the overhead of 2- to 4-digit-year conversion.
3. All programs that use the fixed window technique may need to be manually updated on a yearly basis depending on how your date routine is packaged.
4. All programs that accept output from the fixed window technique must use the same assumptions (current date, past and future windows).
5. Retaining a 2-digit year representation does not provide collating sequence support. Nor does the use of a fixed window technique provide indexing sequence support when 2-digit years are used as index keys in indexed files. You will need to provide additional processing to obtain correct collating and indexing sequence output.

Sliding Window Technique

The **sliding window** technique uses a self-advancing 100-year interval that generally crosses a century boundary. This technique determines the century of a 2-digit year by comparing the 2-digit year against a window of 100 years. The user specifies the number of years in the past and future relative to the system year (generally the current year) that the system sets[1] and maintains. Your applications can access the date that the system sets and automatically advances. This is the main advantage of using a sliding window over the fixed window (where the window is immovable without manually revising the programs each year).

As appropriate to your application environment, you can maintain more than one window. For example, you could set one window to process historical dates, one for mortgage dates, one for birth dates, and so on; and the program adjusts the system date and past and future windows to meet the specific application's needs.

[1] IBM product, Language Environment, provides an option whereby you can set the system year to other than the current year. This flexibility then allows you to set a 100-year range you require; it need not even contain the current year. Refer to Chapter 7, "Tool Categories and Available Tools to Ease Year2000 Changes" on page 7-1.

Consider this specific example. If the dates in your application fall into a range of 35 years in the past and 64 years into the future, based on the current year, 1995, your program can accept and accurately deal with dates of 1960 through 2059. Next year, 1996, the window advances and your application accurately deals with dates of 1961 through 2060.

Graphically, Figure 4-1 on page 4-5 illustrates this example using the current (1995) 100-year window and that same window when the current system date has progressed to the year 2024.

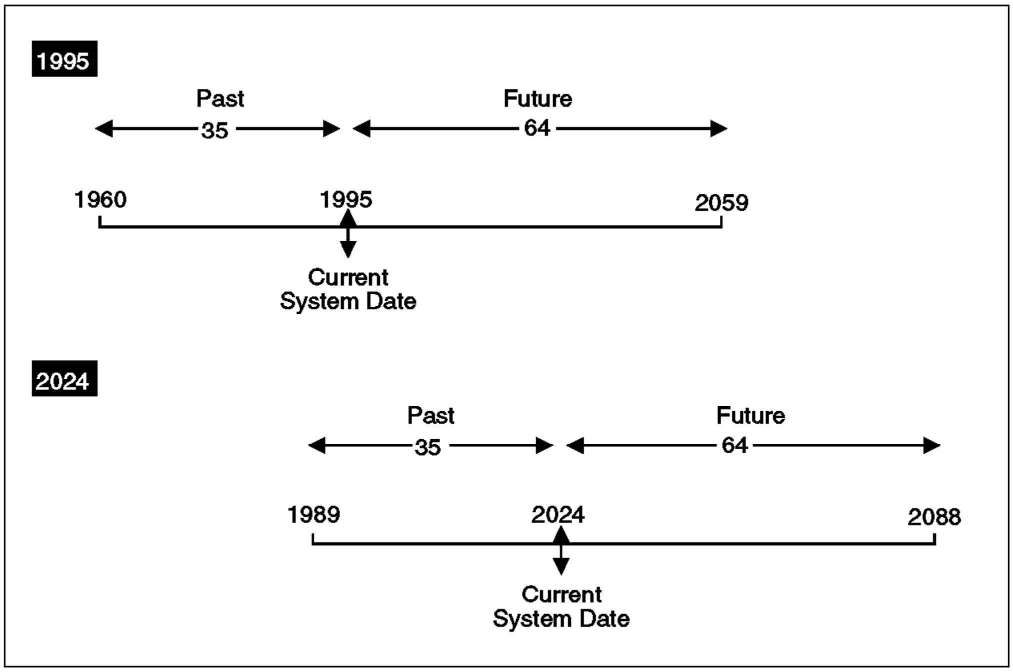

Figure 4-1. Graphical Representation of the Sliding Window Technique. (Using two current system dates, 1995 and 2024, as an example.)

A sliding window approach requires programming logic to interpret the meaning of all 2-digit year data. Such additional programming logic could be packaged into a common data/time service routine, callable from a 2-digit year data exploiter. This would reduce the programming overhead and impact to the calling programs. IBM product, Language Environment, provides common date/time service routines with sliding window features. By default, Language Environment uses a window of 80 years in the past and 20 years into the future that automatically adjusts based on the current year date. For details on using and setting the past/future window range, refer to Chapter 7, “Tool Categories and Available Tools to Ease Year2000 Changes” on page 7-1. For an example of how DFSORT/MVS is implementing a sliding window to sort, merge, and transform 2-digit year data to 4-digit year data, refer to “DFSORT” on page 7-23

Pros and Cons

Pros

1. No need to expand the 2-digit-year format to a 4-digit format.
2. Can provide 4-digit-year format for data reference.
3. Can distinguish years from different centuries using only 2-digit-year format (provided the years being processed are in the range of 100 years at any one time).

4. No need to convert the date data to a new date representation scheme.

Cons

1. Potential exposures exist when/if the function of the software application needs to process years beyond the range of 100 years
2. Potential performance impact in direct proportion to the quantity of date processing the particular application handles
3. All programs that accept output from the sliding window technique must use the same assumptions (current date, past and future windows)
4. Retaining a 2-digit year representation does not provide collating sequence support. Nor does the use of a sliding window technique provide indexing sequence support when 2-digit years are used as index keys in indexed files. You will need to provide additional processing to obtain correct collating and indexing sequence output.

Solution #3: A 2-Digit Encoding/Compression Scheme

This is a 2-digit solution that externalizes only a 2-digit-year format. It requires changes to both your data and your programs. It also requires that you convert, simultaneously, all applications that reference or use the updated data.

Example techniques that are useful when using this solution include **encoding** or **compressing** 4-digit-year data into 2-digit existing space. This section presents several specific examples, many others exist and might prove more applicable to your specific needs.

CAUTION:
Apply this approach with caution. It is considered to be the least desirable approach and should only be used if absolutely necessary. Be certain that the new encoding or numbering scheme does not affect the proper functioning of your programs after all the data changes are implemented.

This solution is considered temporary because there is no guarantee that in the future, your applications will not expand to process dates that are outside the encoding limits.

Some examples include:

- **Example 1:** Convert the numbering scheme from decimal to hexadecimal. However two hexadecimal digits can provide only a maximum of 255 years, for example:

  D‘1900’ + X′FF′ = 1900 + 255 = 2155

  Specific date conversions might be:

  – Convert the year 1900 represented by D‘00’ to X′00′
  – Convert the year 1999 represented by D‘99’ to X′63′
  – Convert the year 2000 represented by D‘00’ to X′64′

- **Example 2:** Convert the data type from the 2-byte character representation of the 2-digit year to a 1-byte **unsigned packed decimal** (two digit) representation, and use the freed byte to append two **unsigned packed decimal** digits to represent a 4-digit year. For example:

– Convert the year 1900 (represented by character string '00' (= EBCDIC X′F0F0′)) to unsigned packed decimal X′00′ and prefix unsigned packed decimal X′19′ in front of X′00′ to yield X′1900′ in unsigned packed decimal.

– Convert the year 1999 (represented by character string '99' (= EBCDIC X′F9F9′)) to unsigned packed decimal X′99′ and prefix unsigned packed decimal X′19′ in front of X′99′ to yield X′1999′ in unsigned packed decimal.

– Convert the year 2000 (represented by character string '00' (= EBCDIC X′F0F0′)) to unsigned packed decimal X′00′ and prefix unsigned packed decimal X′20′ in front of X′00′ to yield X′2000′ in unsigned packed decimal.

- **Example 3:** Convert the numbering scheme from decimal to a user-defined numbering scheme. The mapping between the new and old schemes can be defined by a table or mapping function; and the conversion between the two numbering schemes can be done by table lookup or functional mapping. Figure 4-2 on page 4-7 presents one such possible user-defined table that provides for values up to 1295 within a 2-digit field. This scheme uses the characters 0-9 and A-Z to represent decimal values 0-35, respectively. This base 36 notation is thereby capable of extending the hexadecimal example on page 4-6 by 1040 more years. For example:

$$D`1900' + base_{36}`ZZ' = 1900 + 1295 = 3195$$

Figure 4-2. Example User-Defined Date/Year Conversion Table

2-Character Year (Encoded) Value	Converted Data/Year Value	Year (When Using 1900 as the Base Year)
00 - 0Z	00 - 35	1900 - 1935
10 - 1Z	36 - 71	1936 - 1971
20 - 2Z	72 - 107	1972 - 2007
30 - 3Z	108 - 143	2008 - 2043
40 - 4Z	144 - 179	2044 - 2079
⋮	⋮	⋮
R0 - RZ	972 - 1007	2872 - 2907
⋮	⋮	⋮
Z0 - ZZ	1260 - 1295	3160 - 3195

- **Example 4:** Pack a 3-digit date field with a 4-digit year date by the use of the CYY format. Using a conversion table or offset, you can indicate, for example, that C=0, 1, or 2 represents 19, 20, or 21, respectively. When your application appends the C to the YY field, your system produces full, 4-digit year dates, the range of which depends on the conversion mechanism. Using decimals 0-9 to represent 19-28, this scheme provides a solution from 1900 through 2899, but it is likely to require end-user procedural changes, education, and typical learning curve time and errors.

  One advantage to setting C=0 to represent 19 and so on is that it might provide a compatible, non-disruptive change to some existing application routines if such a field is currently prefixed to your YY data field and set to 0.

  A variation on this same scheme would include the use of the CCYY format where the CC can be used to represent the actual century indicator, 18, 19, 20 and so on, or an encoded value for example, ″00″, ″01″, and ″02″, to represent 19, 20, 21, respectively (see “Solution #1: Conversion to Full 4-Digit-Year Format” on page 4-1).

When adding either the C or CC prefix to the YY field for CYY or CCYY representation, C or CC can be extracted from a separate field, one that is not necessarily adjacent to or preceding the YY field. This then can relieve any restrictions you might currently have due to your date field length. It does, however, require further programming logic and data manipulation.

Pros and Cons

Pros

1. No need to expand the 2-digit-year data format to 4-digit data format. (For example, there is no need to increase the fields in data bases and tables to accommodate dates above 99 which would increase DASD usage.) Further, this saves the effort that would be required to rebuild your database(s).
2. Can distinguish years from different centuries using the 2-character-year format.
3. If you use a COBOL COMP-3 format, you can pack a CCYYMMDD date into an existing 6-byte field (with one byte left over). This technique allows you to retain the original field size and eliminates your need to relocate adjacent fields. Applications that use the data for calculations run faster because the data is already packed.
4. If you use a flagged Julian format (CYYDDD) where C is used as the ‘century indicator’, the format does not require expansion of the date field.

Cons

1. Depending upon the choice of data representation you implement, this scheme can be applied only to a limited date range. For example, you are limited to 255 years when using hexadecimal representation.
2. All programs that use this scheme and need to access the output of the 2-character conversion must change simultaneously.
3. Due to data conversion (calls and processing) you might experience a performance impact in direct proportion to the quantity of date processing the particular application handles.
4. Depending upon the choice of data representation you implement, you might experience incorrect data sequencing if you do not add further programming logic.
5. Encoded dates require conversion whenever you work with that data. Therefore, the presence of encoded dates will add another layer of complexity to such tasks as problem determination.
6. You must convert the data before it can be displayed in Gregorian format, and some encoded data can only be viewed in hexadecimal format. This is both impractical for human reading and also impractical or impossible to print.

Using a Common Date/Time Service Routine

In large system applications it is common to find that more than one date/time service is in use. However, some date/time service routines may have Year2000 exposures of their own, for example, the routine(s) only provides a 2-digit-year format. Fixing the exposed date/time routine(s) is one possible solution. Selecting a vendor date/time routine that is certified as Year2000-ready for consolidation and/or replacement of your 'in-house' date/time service routines is another alternative.

While fixing your current date/time routine(s) exposures, you may find it worth your effort to consolidate all your date/time service routines into one **common date/time service routine**. If you then detect any Year2000 exposures during or following the consolidation, you can reformat your program and data and decide on the appropriate solution(s) you will use. That is, you can package any new code, encoding and conversion routines, windowing-specific data, and so on into the common date/time service routine. This common date/time service routine package might then be considered a 4-digit solution that externalizes both 2-digit and 4-digit-year formats. The benefit of using such a common date/time service routine is lower future maintenance because all services are consolidated rather than replicated throughout your applications.

Considerations When Selecting Solutions

As described in "Determining the Impact of 2-Digit-Year Data Fields" on page 3-3, potential 2-digit-year exposures can be classified into two categories (no impact and impact). When selecting an appropriate solution(s) for the 'impact' categories, be certain to consider not only the applicability of the solution(s) on the module itself but also the potential impact and adjustments on the external modules that receive data from this module. You have three basic choices; you can:

- Change your application
- Change your application and the data
- Invest in a new application (which could also require some date data changes)

Certainly, most IS organizations will build their 2000-ready system on a combination of these choices. When more than one solution appears feasible, weigh its appropriateness based on:

- Time available
- Resources available (personnel and hardware)
- Project cost (individual application conversions and overall)

As today's IS environment becomes increasingly more complex and sophisticated, the instances of program and data isolation decreases. Networking, open and distributed computing allow data to flow from site to site, system program to application program (or application program to system program), and so on. You must ensure that these layers of software 'speak the same language'. As you add in-house code, Solution Developer-written code, and migrate your operating system, be certain to review that software for date format compatibility.

Solution Applicability

Different combinations of solutions are applicable to different situations. Evaluate solutions based on a ‘best-solution combination’ basis when considering both a module itself and other related modules. For example, when applying:

- **Solution #1** (full 4-digit solution) to a certain module, another module that receives data from this module could receive:
  - 2-digit-year data as before, provided there is no exposure for itself or
  - 2-digit-year data as before and apply Solution #2 (windowing techniques) for its own exposures or
  - 4-digit-year data and apply Solution #1 (full 4-digit solution) for its own exposure removal.
- **Solution #2** (windowing techniques) to a certain module, another module that receives data from this module may either receive 2-digit-year data as before and apply Solution #2 itself or receive 4-digit-year data and apply Solution #1 (full 4-digit solution) for its own exposure removal.
- **Solution #3** (the encoding or compression technique) to a certain module, another module that receives data from this module may need to apply Solution #3 as well to maintain data consistency with the data representation scheme. Another alternative is to apply Solution #3 to the impacted module, and then convert that 2-digit-year format to 4-digit-year format before externalizing that data to another module. This receiving module can then proceed with 4-digit data, and if necessary, apply Solution #1 (full 4-digit solution) to adopt the 4-digit-year data for its own exposure removal.

Bridge Programs Help Stage Format Conversions

Bridge programs are often used to convert data from one record format to another. If you use such a program, it should define the:

- Input date format and encoding method
- Output date format and encoding method
- Logic that converts the data from input format to output format based on their encoding methods.

You can apply bridge programs during program execution or file and/or database conversion. For application during program execution, the conversion occurs each time data is passed between programs or between program and source data using different record formats. For application during file and/or database conversion, the bridge program reads one record at a time from the source, transparently converts the record format, and writes out the data in the new format to the destination. The process is incremental and can continue until all the records in the source are converted.

Bridge programs for data format conversion provide the following benefits:

- Granularity when changing the code and/or data

  With the scope of the Year2000 project, it is not practical (if possible) to change all the code and data at once. Bridge programs allow the gradual conversion of the programs and/or data and still maintain the compatibility between different data formats. For example, you can change some of your programs to adopt a new data format and still be able to communicate to programs using the old format (after conversion by the bridge programs).

Therefore, changes to the remainder of your programs can be performed in an incremental manner as convenient.

- Flexibility when choosing appropriate solutions

  Bridge programs allow you to select the appropriate mix of different solutions to best meet your specific circumstances while maintaining the compatibility between different data formats. For example, you can design your programs so that they can process data in different formats. You can then have active data in a 4-digit-year format and archive the same type of data in 2-digit-year format. The bridge program distinguishes the data in these various formats by reading the records and, when necessary, converting the data to the appropriate format.

Other Programming Situations

Other programming situations you should consider might include:

- The possibility that a data format has become outdated and will not function correctly beyond 31 December 1999 (or earlier)

  Such data formats might be outdated even earlier and have already been superseded by another method by the Solution Developer. For example, The MVS platform will no longer support VSAM catalogs for processing when the system date is beyond 1999. To support data sets which need to have explicit expiration dates beyond the end of 1999, or to create cataloged data sets after 1999 on MVS systems, you must use ICF catalogs. Using VSAM catalogs on the MVS platform (including OS/390) will no longer function, this requires a programming change to take advantage of the alternative solution.

- When migrating to year2000 support, your applications (operations) might support only 2-digit-year format, only 4-digit-year format, or both formats.

  It is possible that the 2-digit values are assumed to be 19xx dates. Therefore, be aware that all these must be eventually updated or the functions will fail or will give unpredictable results after 31 December 1999.

- Changes to operations procedures

  Be certain to educate your operators about command changes so that they know when they must use a full 4-digit date (for example, 2000, to avoid implying 1900 if they only enter 00).

- When erroneous data would be produced for a limited and known timeframe and changes are not justified

  You might have a situation that is best handled manually to meet a short time period where programming changes simply aren't justified. Consider using 2-digit-year data if a timeframe such as a single 24-hour period (31 December 1999 to 1 January 2000) or a single week (25 December 1999 through 1 January 2000) would be the only time your application will not provide correct results. For example, a program that looks at a sales report to compare the current day's merchandise movement with the previous 7 days. Because there would only be 8 reports containing both 1999 dates and 2000 dates, you might decide to handle the problem manually rather than changing the code.

Note: Don't fail to use a certain amount of common sense when deciding what applications to change, which to replace, and which to ignore. Do not lose your perspective of your institution's business needs and priorities and the impact and cost a particular application's change might have on attaining those goals.

Guidelines

While retaining a perspective of any external impact, module currency, and what functions are impaired due to Year2000 exposures, use the following guidelines when applying Year2000 solutions.

Note: This is not intended to be an exhaustive guideline, but rather a foundation upon which to start your specific Year2000 date-data resolution.

1. Establish an in-house 'date standard'. Conformance to the ISO Standard 8601 listed below would be a valuable starting point. The earlier such a standard is in place, the sooner your IS organization will avoid creating new date issues and the propagation of current ones. You can refer to:

   - ANSI X3.30-1985 (R1991) *Representation for Calendar Date and Ordinal Date for Information Interchange*
   - ANSI X3.51-1994 *Information Systems -- Representations of Universal Time, Local Time Differentials, and United States Time Zone References for Information Interchange*
   - ISO 8601:1988 *Data elements and interchange formats -- Information interchange -- Representation of dates and times*

   These standards and ordering information are listed in the section entitled "By Title" in Appendix C, "Bibliography" on page C-1.

2. Minimize potential impact to external references due to incompatible date format changes. For example,

   - Maintain the 2-digit-year format as an option when 4-digit format is required for an application program interface (API) that provides 2-digit-year data references.

3. **Avoid** any ad-hoc solutions; such solutions inevitably require a future problem investigation and removal, and should be considered temporary solutions only. For example:

   - Do not determine the century of a 2-digit year by comparing the 2-digit year against a hard-coded threshold, for example, 60. If the 2-digit year is greater than or equal to 60, then the year is a 20th century year; otherwise, it is a 21st century year.
   - Do not fix the leap year calculation formula by adding logic to check if the current year is the year 2000. This solution will temporarily fix the leap year calculation problem by singling out the year 2000, but it does not fix the leap year calculation problem for other years in the multiple of 400.

4. A 2-digit-year format might be acceptable for human-only viewing purposes, for example, screen panels, hardcopy reports, and so on. However, any such data can be, and often is, added to a data set and then read by another program. A 'log' that can be used as input to any program should **not** be considered in this (non-impact, for human viewing only) category.

5. When changing the date format of any 'log', ensure that all the contributing programs adopt the new date format as well.

6. Consider the Year2000 solutions listed in this document (see "Solutions and Techniques" on page 4-1) for applicability in the following order.

   - Using a common date/time service routine (a 4-digit 'solution' - that can support both 2- and 4-digit formats). This is:
     - Considered a long-term solution.
     - **The recommended solution** for its support of both 2- and 4-digit-year formats that provide a long-term solution and no impact to 2-digit-year data references.
   - Solution #1 (conversion to a full 4-digit-year format that externalizes 4-digit formats)
     - Considered a long-term solution.
     - Only supports 4-digit-year formats that will have impact on 2-digit year data reference.
   - Solution #2 (windowing techniques that externalize both 2- and 4-digit formats)
     - Considered a temporary solution and **should only be used when Solution #1 is not practical**. (This is an arguable issue, because there are applications that deal only with years in the range of 100 years. However, there is no guarantee that the functions of the applications will never change in the future and then require 4-digit-year formats.)
     - Has potential exposures when the function of the program needs to process years beyond the range of 100 years.
     - Use this solution only when:
       - Processing is always limited to the current date data, for example, at the time of IPL or time of job creation.
       - Expanding the date-data field is costly, and the function of the software program will be phased out before any exposure occurs.
   - Solution #3 (2-digit encoding/compression scheme that externalizes 2-digit formats)
     - Has potential exposures when the function of the program needs to process years beyond the range that can be covered by the encoding or compression scheme.
     - Should be used only when expanding the date-data field is costly and the function of the software program will be phased out before any exposure occurs.

Chapter 5. Testing Techniques for Year2000 Changes

Testing can be formalized into a 4-phase process, as follows:

Test Type	Used to Test
Unit testing	A single program module
Integration testing	A related group of program modules
System testing	The entire software program
Acceptance testing	The entire software program with live data for production readiness

Ideally, these phases should be completed sequentially. However, when development work is done in parallel, module coding, unit testing, integration testing are commonly integrated, followed by system testing and acceptance testing.

During the process of testing, apply a combination of verification and validation techniques. Unit and integration testing are primarily used for program verification. These two forms of testing comprise structural testing, which is used to uncover errors injected during program coding. System and acceptance testing are used for program validation, and these two forms of testing comprise functional testing, which is used to uncover errors that occurred when implementing requirements or design specifications. The following sections will cover some useful testing techniques and scenarios for Year2000 testing.

Structural Testing Techniques

Structural testing ensures sufficient testing of a function′s implementation and helps determine that all structures of the system are integrated to form a cohesive unit.

Operations Testing

Apply operations testing to determine whether the system is ready for normal system (production) operations. In contrast, recovery processing (discussed below) is intended for abnormal system operations. Considering the potential scope and magnitude of your Year2000 transition, every aspect of the normal operation might be impacted to some extent as you revise programs and/or data for Year2000 readiness. Operations testing ensures that, prior to production, your IS staff can properly administer the applications using the new support mechanisms, documentation, procedures, and training as you complete your Year2000 transition.

Stress Testing

Apply stress testing to determine if the system can function when transaction volumes are larger than normally expected. The typical areas that are stressed include disk space, transaction speeds, output generation, computer capacity, and interaction with people. When testing Year2000 changes, it is essential to verify that the existing resources can handle the normal and abnormal volumes of transactions after the restructuring of the code and the possible expansion of the data fields. For example, apply stress tests to determine:

- if existing CPU capacity is sufficient to meet expected user turnaround time when Solution #3 (refer to "Solution #3: A 2-Digit Encoding/Compression Scheme" on page 4-6) is applied and uses more CPU cycles and processing time for code conversion.
- if existing disk capacity is sufficient to accommodate the additional disk space and provide acceptable disk access time when Solution #1 (refer to "Solution #1: Conversion to Full 4-Digit-Year Format" on page 4-1) is applied and expands the year data field from two to four digits.

Recovery Testing

Apply recovery testing to ensure that the system can restart processing after losing system integrity. This is essential for systems in which the continuity of operation is critical to end users. Recovery processing normally involves the ability to go back to the last checkpoint, then reprocess up to the point of failure. The success of the recovery depends heavily on complete backup data and checkpointing. Any data integrity or unresolved exposures that lead to inconsistent data or code after you have implemented appropriate Year2000 solutions will affect the completeness of backup data. On the other hand, checkpointing is very time oriented and sensitive. Any mis-handling of the time-related data might invalidate system checkpointing. The recovery testing is thus critical in a Year2000 testing environment. It can also involve manual functions (such as hardware or operating system failure), loss of data base integrity, operator error, or loss of input capability. Recovery testing should include all aspects of the recovery processing.

Functional Testing Techniques

Functional testing is designed to ensure that the system and end-user requirements and specifications are achieved. Functional testing focuses on the results of processing rather than how processing is implemented. To accomplish this, create test cases to evaluate the functional correctness of the system and programs. Functional testing techniques include:

Requirements Testing

Apply requirements testing to verify that the system performs its function correctly and that it remains functional over a continuous period of time. Functional checklists such as user requirements, design specifications, compliance of organization′s polices and procedures are used to create test cases to ensure that these requirements are still satisfied following your Year2000 transition. Note that if the Year2000 solutions are merely restructuring code and reformatting data without major redesign of the applications or systems, most requirements testing can be covered by another method, regression testing.

Regression Testing

Apply regression testing to ensure that all aspects of a system remain functionally correct after changes have been made to a program in the system. Because the potential exists for a tremendous amount of data and programs to be involved in your Year2000 transition, any change to an existing program in the system can have a snowballing or cascading effect on other areas in the system. A change that introduces new data or parameters, or an incorrectly implemented change can cause a problem in previously tested parts of the system, simply because of the way data can be shared between software entities.

Regardless of how an error was introduced or propagated, regression testing needs to be conducted to retest even unchanged parts or programs of the system. Normally, tests that have been previously run are reused to ensure that the same results are achieved. In most cases, regression testing is automated because the test cases and the results are already known.

Error Handling Testing

A normal error-handling cycle is an iterative process that either prevents errors from occurring, or recognizes and corrects errors that have occurred. Error-handling testing is necessary to determine the ability of the system to properly process incorrect transactions that can be reasonably expected as types of error conditions. For example, programs that accept only 4-digit-year-data-entry format need to provide error messages for data entry in 2-digit-year format, and vice versa for programs that accept only 2-digit-year-data-entry format. When changing from 2-digit-year format to 4-digit-year format, you need to apply error-handling testing to verify the appropriate error-handling functions.

Manual Support Testing

Apply manual support testing to evaluate the adequacy of the processes used by people (end users) who must handle the new data generated from the automated applications with Year2000 support. Types of data from these applications include data entry and report generation. Any new data format should be easy to understand and not ambiguous. This method includes testing the interfaces (for example screens, procedures, operation manuals, and online HELP panels) between end users and the application program. End users should be trained and use procedures provided by the system personnel. Testing should be conducted without any other assistance.

Intersystem Testing

Applications are frequently connected with other applications to provide a higher or deeper level of functionality. Data may be shared between applications or systems. Multiple applications or systems may be involved in such an environment. This is the typical environment for Year2000 projects. Intersystem testing is required to ensure that the connection functions properly between the applications. This test determines that the proper parameters and data are correctly passed between applications, and proper coordination and timing of each function exists between applications.

Parallel Testing

Parallel testing is used to determine whether the processing and results of a new version of an application are consistent with the processing and results of the previous version of the application. It should be applied when the old and new versions of the application are similar. For Year2000 solutions without any major function redesign, this is the ideal technique. Parallel testing requires that the same input data be run through the two versions of the application. However, if the new application changes data formats, such as reformatting the year-date notation to 4-digit format, you must modify test input data before testing.

The efficiency and effectiveness of parallel processing is highly dependent on the degree of difficulty encountered in verifying output results and preparing common input. It may be difficult to automatically verify the results of processing by comparing the results on a tape or disk file. Some automated test tools or customized solutions can be used to prepare input and verify output more quickly.

How to Change Date and Time for Testing

By nature, Year2000 exposures are time-sensitive and time-driven. Basic Year2000 testing requires that you set the system date and time to a point where Year2000 exposures can be detected, then removed.

Be cautious before resetting the system timer. Some system resources and functions are time-sensitive and may be activated or de-activated when you reset the system clock. Such effects can occur when you either set the system clock forward or backward. Without careful planning, you could cause the loss of these system resources and/or functions, some of which might prove very difficult and time-consuming to recover.

The most vulnerable resources and functions subject to expiration include:

- user IDs
- passwords
- data files and databases
- authorization/protection
- licences/services
- network access
- automation functions (as well as unexpected activation)
- hierarchical storage management

Security products (such as RACF) and other date-dependent functions require particular attention. Setting the system clock ahead can cause immediate loss of system programmer or end-user access and data expiration. Further, to maintain security in this environment, consider implementing an installation exit (available in RACF) to disable expiration date processing.

Ensure that you do not contaminate your production system or production data bases when running various test scenarios. **Be certain to heed the following requirements:**

For example, if your system is set up to scratch all files that are 1-year old, all files will be scratched when the system clock is changed to 2000/01/01 or later from any date prior to 1999/01/01.

Testing on System/390 Platforms

Requirements:

System Requirements When Changing The Test System Clock:

- Provide a separate, isolated test system (such as a separate logical partition (LPAR) for large operating systems)
- Provide a separate set of test data
- Turn off RACF or other date-dependent functions.

If resources allow, you can also provide separate storage devices (DASD) as a further measure to protect your production system and data.

Each LPAR or VM guest has its own logical clock which is separate and distinct from both the hardware TOD clocks and from the clocks of all other LPARs or guests running on the same physical machine. Any program, application, or system, running in the image gets the same future date.

Any of the S/390 operating systems with a clock set to a future date can be run in a Processor Resource/Systems Manager (PR/SM) LPAR, as a VM guest, or on a dedicated machine. This is also true for images in LPARs, guests on VM, or when running a native MVS system. Refer to the individual operating systems below for details.

Note: Non-IBM systems that have features similar to PR/SM might not function in the same way as an IBM PR/SM LPAR.

You can change the system date through either of the following methods:

1. Set the system timer on a test system image. **However, never share data or data sets between such a test system and any other system unless all clocks are synchronized to that same date.** For example,

   - **On MVS:** you can set the clock in an MVS image to a future date. To set the clock, reply to the TOD clock prompt message IEA888A [GMT DATE=...,CLOCK=...] LOCAL DATE=...,CLOCK=... REPLY U, OR GMT/LOCAL TIME in the test LPAR image. (Do not set the time of day (TOD) using the Sysplex Timer.)

   - **On VM/ESA:** systems will IPL correctly with a year of 2000 or later. For VM/ESA Version 1 Releases 2.1 and 2.2, you must apply APAR VM57927. Beginning with VM/ESA Version 2 Release 1, this capability is included in the base VM system.

     In order to change the date of your VM/ESA system, reply 'Yes' to the Change TOD clock (Yes|No) prompt during IPL, and change the date and time to the desired date and time. To specify a year of 2000, simply enter 00 as the year (01 for 2001, 02 for 2002, and so forth).

     When the system date is changed, issue a SHUTDOWN REIPL command immediately after completion of the system IPL as indicated in message HCPITM1161I. This is required because by the time VM issues the Change TOD clock (Yes|No) prompt, components of the control program (CP) have already developed sensitivities to the actual value of the TOD (time of day) clock. These sensitivities might cause irregular scheduling and dispatching behavior if the value of the TOD clock is changed by more than several minutes.

- **On VSE:** use command SET DATE=, CLOCK=, ZONE= to initialize the system timer during IPL time. The DATE parameter is specified in the format mm/dd/yy, whereby yy is interpreted as 20yy, when yy < 50, and as 19yy otherwise.
- **On TPF:** use the ZATIM (alter time) functional message to change the system time-of-day (TOD) clock, change the subsystem local standard time (LST) clock, and synchronize the TOD clock to a Sysplex Timer. Do so as follows:
  - Issue the ZDTIM functional message to determine the time base before you issue this functional message.
  - If you issue this functional message when the TPF system is above 1052 state, you must cycle the TPF system to 1052 state to complete the time adjustments.
  - You can issue this functional message with the TOD parameter only in 1052 state.
  - In a loosely coupled system, all other active processors must be in 1052 state in order to change the TOD clock without the BP option.
  - The ZATIM functional message does not adjust time-initiated functions (that is, functions that were started by using the CRETC macro).

2. Intercept the call to date and time routines or system timer services. Change the date and time value returned from the routines/services to a specific value that will cause exposures. For example,

   - On MVS, trap the MVS TIME macro (SVC 11) and the STCKSYNC macro.

   Notes:

   a. There are Solution Developer tools available that provide the function of time simulation. Refer to "Solution Developer Tools" on page 7-71 for a list of Solution Developer tools. For any time simulation tool, the change of time should be on application-level programs and should not affect the system functions and operations. The tool should allow the users to specify the scope of the applications with time change. Once the scope is specified, the change of time should be within that pre-defined scope and transparent to others applications.

   b. When you request full coverage of time references, you must ensure that all forms of time references are intercepted. For example, some date/time service routines use hardware instructions to reference time. Such tools will not intercept the STCK on MVS and control block references.

Testing on Non-System/390 Platforms

For the non-System/390 platforms (AS/400 and PCs), you can affect the system date change as follows:

- **On AS/400:** use the CL command (change system value (CHGSYSVAL)) with the QDATE parameter. For example, CHGSYSVAL QDATE('101300') changes the system date to October 13, 2000 on a system using MMDDYY date formats (QDATFMT). To make sure all jobs on the AS/400 are using this new date, you should then power down the AS/400 (PWRDWNSYS) and conduct your testing after the subsequent IPL.

- **On PC:** use the configuration utility that sets the time and date or execute the DATE command in DOS Version 3 Release 3 or later.

Basic Testing Scenarios

The scenarios for Year2000 testing depend heavily on the system environment and applications. Some basic Year2000 testing scenarios that are common for most installations are suggested here:

- Set the clock to test process cycles and automatic functions that are activated on a regular basis. These scenarios can be used to identify Year2000 exposures that need to be fixed as well as to validate programs after applying Year2000 solutions.
  - Daily
  - Weekly
  - Semi-monthly
  - Monthly
  - Bi-monthly
  - Quarterly
  - Semi-annually
  - Annual
  - Automatic archiving
  - Automatic restart/restore
  - On demand
- Test the setting and display of special dates, including:
  - 1900/2/29 - should fail - the year 1900 is not a leap year
  - 1996/2/29 - should succeed - the year 1996 is a leap year
  - 2000/2/29 - should succeed - the year 2000 is a leap year
  - 00/01/01 - should display an unambiguous 4-digit-year date, the value of which depends on the application. For example, 1900/01/01, 2000/01/01, and so on
  - 1999/12/31 - should be able to distinguish between a regular end-of-year 1999 date and a special meaning date. For example, a never-expiring date indicator.
- Test the processing of time-sensitive data with different combinations of data and time
  1. Use the current system clock and then test data with dates:
     - before 2000/01/01
     - after 2000/01/01
  2. Set system clock before the year 2000, for example 1999/12/31, and then test data with dates:
     - before 2000/01/01
     - after 2000/01/01
  3. Set the system clock after 2000/01/01 and then test data with dates:
     - before 2000/01/01
     - after 2000/01/01

Basic Scenarios to Test Your PC System Clock

Some older models of the PC may not have the capability to set or roll over the system clock beyond the year 2000 because the Basic Input/Output System (BIOS) is unaware of the century digits. Refer to "IBM Personal Computer (PCs) - Hardware Timer Setting" on page A-26 for PC-specific, internal clock setting information. Some suggested scenarios for testing for year2000-readiness of your PC system clock follow:

- Test if the system clock can be set beyond the year 2000
  1. Set the system clock to 2000/01/01, 00:01:00
  2. Check the date
  3. If the date is set correctly, power off, power on, and then re-check the date
- Test the system clock automatic update function
  1. Test the system clock automatic update function when the power is on
     a. Set the system clock to 1999/12/31, 23:58:00
     b. Keep power on
     c. Wait until the clock reaches the year 2000
     d. Check the date
     e. If it is set correctly, power off, and re-check the date
  2. Test the system clock automatic update function when the power is off
     a. Set the system clock to 1999/12/31, 23:58:00
     b. Power off
     c. Wait until the clock reaches the year 2000
     d. Power on
     e. Check the date
- Test the time update by the operating system
  1. Test the time update after suspension of a time-sensitive program
     a. Set the system clock to 1999/12/31, 23:58:00
     b. Suspend a time-display program without a 'wake-up' timer
     c. Keep the power on
     d. Wait until the clock reaches the year 2000
     e. Resume time-display program and check the date
  2. Test time update after suspension and 'wake-up' of time-sensitive program
     a. Set system clock to 1999/12/31, 23:58:00
     b. Suspend a time-display program with the 'wake up' timer set at 2000/01/01, 00:01:00
     c. Keep the power on
     d. Wait until the time display program 'wakes up'
     e. Check the date

Chapter 6. Migration Consideration for Year2000 Transition

To accomplish a successful migration and eliminate your current Year2000 exposures, you will need to follow a well-architected migration plan and prepare to execute that plan prior to actually performing the migration. This section provides an outline that you can use as a checklist of those steps that will help you plan, prepare, and execute your migration. Note that some steps may not be necessary for your specific environment.

Plan for Migration

1. Plan for Migration
   - Determine the sequence of steps needed for migration
   - Review the migration procedures with your system administration staff and your end-user community
   - Determine the resources/time required for migration
   - Assign individuals/organizations to each migration step
   - Document the migration sequence and responsibilities
   - Develop a schedule for migration of the new system to reach production mode
2. Examine all changed data that use new/changed date format
   - Determine the source of the new data in the existing systems
   - Determine the data that can be converted automatically
   - Determine the data that must be converted manually
3. Design bridges/interfaces among packages and reusable modules to maintain compatibility, if needed
   - Design bridges/interfaces to application packages, if needed
   - Design bridges/interfaces to reusable application systems, if needed
   - Design bridges/interfaces to old systems that will coexist with the new systems, if needed
   - Design tests for the verification and validation of these bridge facilities, if needed
4. Design procedures for manual data conversion
   - Update documents/procedures that will be used for manual data entry
   - Determine checking mechanism for the accuracy and completeness of manually entered data
   - Design the new screens with new date format for manual entry of new data and review with your end-user community
   - Design/update the software to load the manually prepared data into the new system
   - Run a rehearsal of the manual data entry and estimate the impact of the new data format on data entry time
5. Design procedures for automated data conversion

- Design new software or use automated tools for automated data conversion
- Determine a checking mechanism for the accuracy and completeness of automatically converted data
- Design recovery procedures for conversion of data errors caused by missing data
- Estimate the resources and time for automated data conversion

6. Develop the data conversion systems
   - Develop subsystems to convert existing data
   - Develop subsystems for the entry of new data
   - Develop bridges/interfaces to old systems which will remain in production
   - Develop bridges/interfaces to application packages and reusable modules
   - Verify and validate the accuracy of the data conversion systems
7. Plan the hardware installation for new system, if needed
8. Plan for final system testing
   - Determine the testing strategy
   - Develop the detailed test plan and schedule
   - Determine the types of tests to be conducted on the new system
   - Plan the testing environment
     – Design the migration tests for the systems and applications
     – Determine what testing software or tool(s) will be used for each type of testing
     – Determine what testing libraries will be used for each specific set of programs and data
     – Install needed testing software. For example, test data generator, test utilities, debugging utilities, and so on.
     – Build test libraries, and test data
     – Coordinate the testing with your development team and your system administration staff
9. Documentation and training
   - Update your ‘corporate standard guideline’
   - Update your technical documentation. For example, development guidelines and testing handbooks
   - Update the production procedures
   - Update the user documentation
     – Update the on-line documentation, including HELP screens, on-line manuals, computer-aided training, and so on.
     – Test the on-line HELP and training aids with your end users to evaluate the acceptance of the new on-line information

- Update all hard-copy documentations to reflect changes and review those documentation with your end users
- Plan and conduct training program to smooth the migration process

Perform Migration

1. Update production procedures, if necessary
2. Install the Year2000-ready production system environment
   - Coordinate with vendors for the hardware installation, if needed
   - Install the hardware of the new production system, if needed
   - Coordinate with system programmers/operators for installation of the Year2000-ready software
   - Install the Year2000-ready system/vendor/application software on the new production system
3. Perform data conversion process
   - Load existing data into the new system′s databases
   - Execute the data conversion programs or automated tools for data conversion
   - Load manually-prepared data through data entry
   - Integrate the existing and converted data
   - Test the integrated data to verify data integrity
4. Perform final system/migration testing
   - Plan the sequence in which separately developed subsystems will be tested and verified in reasonable combinations.
   - Verify that the portions of the system that have no changes still runs properly as changes are made to other portions of the system
   - Verify that the program handles all its transactions correctly and remain stable for a defined period of time
   - Verify that the system can accept input from, and provide output to, other systems with which it interfaces as interfaces change
   - Verify end-user acceptance of the new system to certify the system as acceptable for production.
5. Activate the new system in production
   - Switch the new system to production mode
   - Run the new system in parallel with the old system
   - Phase out the old system as the new system becomes stable
6. Migration review
   - Monitor and evaluate system performance, throughput, and reliability
   - Determine what system tuning is needed based on system status records
   - Track and evaluate user acceptance of the new system
   - Determine and document what system and application function enhancements are needed

- Plan and schedule the system and application function enhancements
- Coordinate system function enhancements with vendors
- Design and develop required in-house application-function enhancements
- Determine when the system and/or application enhancements will be applied
- Apply the enhancements, once available, to the new system

Chapter 7. Tool Categories and Available Tools to Ease Year2000 Changes

With the critical time constraint of the Year2000 challenge, customers require Year2000-ready applications to be available shortly. In addition, customers demand Year2000-ready applications of high quality and reasonable cost. It would not be possible to achieve this without the use of powerful and productive tools. There are a variety of tools available from Solution Developers that can help you confront the software challenge of the 1999 to 2000 date change. Refer to "Solution Developer Tools" on page 7-71 for a partial list of Solution Developer tools.

To be effective and efficient in providing the ability to rapidly change programs and data to properly handle the year 2000, the tools must have certain characteristics. Some important tool characteristics and tool types that are necessary to make these changes are summarized in this chapter. This chapter also provides lists of tool products being marketed by Solution Developers and IBM as general information and as a reference to obtain further information.

Tool Characteristics

The tools should provide necessary features such as:

- interactive environment
- batch processing capability
- graphic user interface
- ease of use
- rapid prototyping
- speedy and easy editing/updating
- stepwise refinement
- backward recovery

In addition, consider both the software ***development*** and ***deployment/target*** environment of the tools such as:

- Platform
  - Host only
  - Workstation only
  - PC only
  - Client/Server-based
  - Cooperative processing with host
  - LAN based
- Prerequisite hardware (both minimum required and recommended for PC specification, if necessary) such as:
  - memory/storage required
  - DASD needed for tool installation
  - DASD needed for tool usage
- Prerequisite software that is required and supported such as:
  - Host operating system(s) - for example MVS, VM, VSE, OS/400, Windows NT, OpenVMS, AIX

  - PC/Workstation operating system(s) - for example DOS, Windows, OS/2, Unix, MacOS
  - Network operating system(s) - for example Netware, Banyan, Windows NT
  - Communication protocols - for example TCP/IP, SNA, IPX, NetBios
- Languages supported or generated such as: COBOL, C, C++, PL/I, FORTRAN, RPG, PASCAL, Smalltalk, Assembler
- Client/server models supported such as:
  - Transactional - for example CICS, Encina, Tuxedo, Remote Procedure Call (RPC)
  - Conversational - for example APPC/CPI-C, NetBios, IPX, TCP/IP, SNA
  - Database server - for example DB2 family, Sybase, Oracle, EDA/SQL, SQL Server

Once you run the appropriate tools against your operating system to reformat Year2000 exposures, be certain that the 'newly created' system maintains its ability to:

- Achieve good machine performance
- Process a reasonably large number of users
- Process reasonably large databases
- Process high-traffic volumes
- Provide networking access
- Provide recovery from failures
- Provide security
- Provide audibility or accounting
- Provide ease of maintenance.

Tool Categories

The following sections highlight some useful tool types for handling the year 2000 challenge. A brief description is provided for each of the types. Most of these tools are used in normal prototyping and application development. Tool types can be categorized as follows:

Impact Analysis

To analyze the impact to your programs, you can use tools to:

- ***analyze complexity***

  Determines the complexity of a software design or code using a metric such as 'fan-in/fan-out', degree of nesting, or other characteristics. These tools provide complexity analysis to allow you to estimate the effort required to change the date/time-related items in your source code.

- ***analyze impact***

  Analyzes the program modules and related data to determine what is impacted and related. These tools are very time-efficient but do not always guarantee the accuracy of their analysis because they tend to over-estimate what is affected. When an impact analysis tool indicates that some data is affected, it does not always mean that the data will need to be changed (such as to reformat the date-related data and programs for Year2000 readiness).

- ***analyze metrics***

  Collects, analyzes, and reports the results of metrics quantification and analysis activities. These tools can analyze and predict how much work, in quantification, will be needed to reformat the date related data and programs for the year 2000, based on a metrics or cost model. You must validate the accuracy of the metrics you intend to use, that is, its predictions against actual human performance in real-life situations.

- ***analyze database***

  Investigates the structure and flow within a database to observe the characteristics of the database and determine if certain measurements/requirements can be realized. For example, analyze the year fields of the databases for any use and cross reference of 2-digit years.

Project Management

To help manage the project, you can use tools to:

- ***inventory software***

  Determines all code, JCL, databases, and other programs that constitute your system to provide a complete list for impact analysis. The list can be further divided into lists of sub-systems when partitioning and prioritization of the project is necessary.

- ***track changes***

  Tracks and logs all requests for code and/or data changes. Requests are tracked through completion or resolution. Any inconsistent/missing changes of data or programs due to date format changes will then be minimized.

Program Level Analysis

To analyze programs and data, at a program level, you can use tools to:

- ***analyze data flow***

  Shows the flow of data among modules in procedures or programs. Determines if a data-flow diagram is complete, consistent, and adheres to those rules established that govern flow. This provides both high- and low-level views of data flow within the system, and can be used to verify completeness of the program and data changes.

- ***diagram logic structure***

  Diagrams how program modules call sub-modules, and what data and control information these program modules share. These tools display multiple program views.

- ***diagram data structure***

  Diagrams the representation of appropriate parts of a data model as the structure is used by a database management systems structure and relational structures.

- ***diagram relationships***

  Illustrates multiple relationships of a program module or data element at the same time. This is useful to understand how data is shared among the programs that have access to it.

- ***diagram decomposition***

Allows a high-level overview specification, for a design or data model, to be successfully decomposed into smaller entities for further observation and analysis. It facilitates the partitioning of a project that is too large to tackle all at once, such as the Year2000.

- ***slice programs***

  Allows you to view all the code affecting a given variable or statement. Forward slicing starts with a name or statement, and indicates what that name or statement affects. Backward slicing starts with a name or statement, and indicates all the parts of the program that *could* affect it.

- ***analyze logic***

  Inspects the use of control logic within a program, determines if it is proper, and mechanizes the specified design. It is useful for verification and validation of the correct manipulation of time when windowing techniques are used.

Code Editing and Restructuring

To help edit and restructure code for your programs, you can use tools to:

- ***power browse***

  Allows you to scan and inspect code. Scanning can be switched between program (data) structure charts and code. These tools are more powerful than regular text editors. They can provide, for example, syntax checking, sophisticated capability to find data or information, or the ability to edit multiple programs. These types of tools can also include reverse engineering tools or maintenance workbench tools.

- ***find dates***

  Locates date-oriented data, variables, declarations, comments, or other information in code for investigation of potential reformatting.

- ***comparison***

  Compares two software programs, files or data sets to identify commonalities and/or differences. It is extremely useful for verification of program changes when date reformatting is done by simple field expansion.

- ***cross reference***

  Lists where variables, procedures, or other items are located in the code. These tools speed up browsing and provide a limited form of impact analysis.

- ***expand fields***

  Automatically expands 2-digit-year fields into 4-digit-year fields. It saves editing time tremendously and provides complete coverage of field expansion.

- ***analyze interfaces***

  Determines if a range of variables in the programming interfaces is correct, as the variables are referenced across the reference boundaries. It can designate 4-digit-year format as a standard interface and enforce the standard in the programming interfaces.

- ***analyze standard/consistency***

Determines whether prescribed development standards have been followed. Identifies inconsistency in conventions used in requirements, designs, or programs. These tools can help you introduce standard or more uniform names to date-oriented fields or keywords, and improve the consistency and accuracy of data. These tools enforce consistent indentation and alignment.

- ***trace requirements***

  Traces how the requirements are realized in the design and code.

- ***modularize code***

  Generates modular code and top-down control flow. Reduces the scope of complex programs by creating separate modules. Identifies routines that are frequently referenced or changed, for example date/time services routines, and creates re-usable code libraries.

- ***standard date subroutine***

  Creates reusable program modules that have correctly implemented date handling. These date subroutines can replace individually developed date subroutines to standardize the use of a date routine. These tools also reduce the chance of error and the cost of development, maintenance, and testing.

Code Generation

To generate code for your programs, you can use tools to:

- ***generate database code***

  Generates database code directly from the data structure diagram.

- ***paint screen***

  Generates code for the screens of a computer-user dialog or data entry when the screens need update for reformatting of date.

- ***generate dialog***

  Generates dialogs that conform to specified standards, for example, 4-digit-year input/output standard in any dialog.

- ***generate reports***

  Generates code for the structure and layout of a report, along with calculations of derived fields in the report.

- ***generate code***

  Generates executable code from high-level specifications.

Automate Testing

To automate testing for your programs, you can use tools to:

- ***simulate***

  Represents certain features or functions of the behavior of a physical or abstract system. One example of such a tool is a clock simulator that can change your system clock, while being transparent to your programs. Tools such as these provide an easy way to quickly expose your programs to Year2000 scenarios.

- ***analyze tests***

Determines the test case coverage on a set of programs being tested (whether a segment of code had been tested by other testing).

- ***generate test data***

  Generates test data directly from a specification and facilitates a sequence of testing steps.

- ***test data libraries***

  Organizes test data for use. These libraries are most useful in regression tests.

- ***test drivers***

  Automates testing by triggering test cases during testing. These tools often provide test input, execute the test cases, compare actual test output with expected results, and report test results.

IBM Tools for MVS

The IBM COBOL Family for MVS & VM

For application code written in COBOL for S/370 and S/390 (for MVS and VM) platforms, IBM has developed COBOL compilers. These products target the host application development environment.

IBM′s COBOL mainframe products for S/370 and S/390 are:

- **COBOL for MVS & VM** (new name for IBM COBOL/370) – the compiler
- **Language Environment for MVS & VM** (new name for Language Environment/370) – the run-time library
- **CoOperative Development Environment** (CODE/370) – edit/compile/debug tool

Features

This COBOL compiler, COBOL for MVS & VM, is available today and provides full 4-digit year support with features including:

- Intrinsic functions
- Sliding century window

COBOL for MVS & VM provides full Year2000 support. It provides ANSI COBOL Standard Intrinsic Functions which give full date manipulation capability with 4-digit-year support. Language Environment for MVS & VM provides additional date manipulation with the Language Environment Callable Services. COBOL for MVS & VM will return a 4-digit year when the COBOL application program queries the system for the current date. OS/VS COBOL and VS COBOL II return the system current date with a 2-digit year.

Companies need productive application development environments. A language compiler is only a portion of what application programmers need today to develop and maintain code. Tools that generate statistics from code; modularize, migrate, or restructure code; or search class libraries; are becoming more and more important.

These tools assist in migrating your existing COBOL applications to COBOL for MVS & VM:

- **CCCA** – converts source code
- **COBOL Structuring Facility** – analysis, reporting, restructuring
- **CICS Application Migration Aid** – converts CICS source
- **COBOL Report Writer Precompiler** – supports Report Writer code
- **EDGE Portfolio Analyzer** – load inventory tool
- **WITT** Product Family – testing
- **ReDiscovery** Product Family – cataloging of parts.

IBM COBOL for MVS & VM

IBM COBOL for MVS & VM (program product # 5688-197) is the high-performance compiler with a supporting runtime environment (Language Environment for MVS & VM (program product # 5688-198)), which facilitates multiple-language interaction. This is different from the single packaging of compiler and runtime environment for OS/VS COBOL and VS COBOL II. An additional component, CoOperative Development Environment/370 (CODE/370) (program product # 5688-194) provides an integrated workstation development environment for developing host applications. It includes an editor, compiler invocation facility, and a debug tool for the host.

Applications written using COBOL for MVS & VM can interface with a variety of IBM products, such as SQL/DB, DB2, CICS, GDDM, ISPF, and Data Window Services available on MVS/ESA. The consistent interlanguage communication support, common protocols, and suite of callable services provided by Language Environment for MVS & VM is designed to allow easier access to in-house applications or vendor packages written in COBOL for MVS & VM.

IBM COBOL for MVS & VM Release 2 brings object-oriented programming to the MVS COBOL programmer. Object-oriented extensions to the COBOL language syntax are provided for MVS. SOM is the core OO architecture for SOMobjects for MVS and is IBM′s strategic architecture for building and manipulating class libraries. Enabling to SOM on MVS allows IBM COBOL programmers to develop class libraries using native COBOL language with object-oriented extensions. The COBOL SOM objects permit COBOL object-oriented applications to access SOM objects implemented in other languages, in addition to full interoperability with existing COBOL applications and data.

IBM COBOL for MVS & VM provides:

- Intrinsic functions, which reduce the need for extensive algorithms
- Programmer access to all of the elements in a table at once, reducing the need for explicit loops
- Consistent interlanguage communications, common services, and common functions, which helps extend the useful life of existing applications
- Improved dynamic calls
- Support for Year2000
- Capabilities to help application programmers incrementally enhance applications
- Help in maintaining and enhancing the investment in existing programmer skills.

The following new features are included in IBM COBOL for MVS & VM Release 2:

- Object-oriented language extensions – allowing developers to create mission-critical business applications that run only on the host or as part of a client/server application. These extensions are based on a subset of the evolving ANSI OO COBOL Standard.
- Support for the direct creation of SOM objects on the host via COBOL language syntax
- Optional SOM Interface Definition Language (IDL) generation
- Access to existing SOM-based class libraries
- Improved interoperability with C and C++
- Source-level compatibility with IBM VisualAge for COBOL for OS/2 and IBM COBOL Set for AIX
- Performance enhancements

IBM COBOL for MVS & VM Release 2 is compatible with VS COBOL II and IBM COBOL/370 Release 1. For SOM applications, customers must also order SOMobjects for MVS (program product # 5696-822).

IBM COBOL for MVS & VM provides facilities to acquire and integrate packaged software, consistent with the vendor′s terms, into existing applications irrespective of the language used; use existing code in new applications (code reuse) regardless of the source code language used; and invoke functionality between applications with improved interlanguage communication (ILC).

IBM COBOL for MVS & VM includes VS COBOL II language features, enhanced compiler options, virtual storage constraint relief, structured programming language for improved programmer productivity, enhanced double-byte character set (DBCS) support, streamlined system interfaces, expanded code optimization, flexible run-time options, and support for the VSE/ESA 31-bit addressing feature.

With IBM COBOL for MVS & VM, the ″RES multitasking restriction″ is lifted. Thus, independent COBOL applications can be executed under different tasks in the same MVS address space. In particular, COBOL applications can be executed from both halves of an ISPF split screen.

Most applications were coded with 2-digit-year data. COBOL for MVS & VM provides intrinsic functions not available in our earlier COBOL compilers. Calendar functions include:

- CURRENT-DATE
- DATE-OF-INTEGER
- DAY-OF-INTEGER
- INTEGER-OF-DATE
- INTEGER-OF-DAY
- WHEN-COMPILED

Example of obtaining the 4-digit year with the COBOL Intrinsic Functions:

- DATE-OF-INTEGER gives YYYYMMDD
- DAY-OF-INTEGER gives YYYYDDD

IBM Language Environment for MVS & VM

IBM Language Environment for MVS & VM (program product # 5688-198) is IBM′s common runtime environment for enterprise applications written in COBOL, PL/I, C, and FORTRAN. Language Environment for MVS & VM is designed to provide defined calling conventions, enhanced interlanguage communication, callable services, language-specific services (for example, common facilities messages), common math functions, application utilities, system services, and subsystem support for Customer Information Control System (CICS) and Information Management System (IMS).

With Language Environment for MVS & VM, application programmers can extend and integrate their applications and packages, as well as reuse code with greater flexibility, due to interlanguage communication and native language restrictions. Application packages may be extended with the language of choice. Language Environment for MVS & VM enables existing applications to function as before with little, if any, changes required, thus helping preserve company investment in those applications. Language Environment for MVS & VM condition handler permits programs to handle errors in a predictable, logical manner without highly-specialized routines.

Language Environment for MVS & VM replaces the existing language-specific run-time libraries and provides a common runtime environment for all languages that conform to the Language Environment architecture. In a single product, Language Environment for MVS & VM combines essential run-time services, such as routines for message handling, condition handling, and storage management. All these services are available through a set of interfaces that are consistent across programming languages. With Language Environment for MVS & VM, application programmers can use one runtime environment for their applications, regardless of the programming language or system resource needs.

Today, IBM Language Environment for MVS & VM is the run-time library for the Language Environment-enabled compilers IBM COBOL for MVS & VM, C/C++ for MVS/ESA, and PL/I MVS & VM. IBM realizes the importance our customers place on complete, open, high-quality solutions. Therefore, IBM has established an *IBM Language Environment Partner Program* at the IBM Santa Teresa Laboratory. This program encourages and assists tool and application package vendors to support and take advantage of Language Environment services. Companies with existing 3GL applications will benefit from the wider choice of tools and application packages.

Language Environment for MVS & VM provides a number of advantages over other packages; its capabilities include:

- Ability to parse dates in an infinite number of formats by using a picture string feature as a parsing guide
- Provide NLS support by using a built-in table of defaults based on a country code
- A sliding window feature

  IBM Language Environment for MVS & VM provides a valuable short term solution with the century window feature. If you are unable to change all of your applications and data at the same time, the century window feature allows 2-digit years to be interpreted in a 100-year window. Any 100-year window can be selected. You pass a 2-digit year to Language Environment and Language Environment returns a 4-digit year based on the 100-year window.

  The advantage to the century window is that you need to change only the application code and not the databases or files with 2-digit years. This allows you to change the application programs one at a time or groups at a time without effecting your data files. Note, this only works for dates that range less than 100 years. For example, dates of birth may not be appropriate for this solution; a person born in '94' could be over 100 years of age.

  The disadvantage to the century window is that it doesn't last forever. If your application has a long life expectancy, you many need to go back and replace the century window with full 4-digit year support. Some of your applications will be replaced prior to this replacement.

IBM CoOperative Development Environment/370 (CODE/370)

IBM CoOperative Development Environment/370 (CODE/370) (program product # 5688-194) — the common editor, compiler, debug tool — provides a cooperative environment, allowing application programmers to more productively develop and maintain host IBM 3GL applications from the workstation. CODE/370 provides a consistent, graphical user interface across different platforms and languages, a language-sensitive editor, language-sensitive help, a compiler invocation facility, and an interactive debug tool. CODE/370 combines the richness of the S/370 or S/390 subsystem environments and the power of IBM Language Environment for MVS & VM to provide a host Debug Tool which allows programmers to find bugs, fix bugs, and test applications. The Debug Tool is available either as a stand-alone 3270 host debug tool (for programming shops where workstations are not available) or with an optional graphical workstation user interface.

CODE/370′s cooperative environment allows application programmers to perform host programming tasks, such as compiling and debugging, from a workstation. Through cooperative processing, users perform functions locally on the workstation while interactively accessing the programs, data, and compilers residing on a host system. The optional workstation interface combines the Edit and Compile / Link functions together with the Debug Tool graphical user interface (GUI).

The powerful workstation-based editor integrates a rich set of functions that will speed up your application-development activities. The editor works with any type of source. The language-sensitive features help optimize coding efficiency in COBOL, C, PL/I, REXX, and JCL. Source can be stored in an MVS data set, a VM file, or an OS/2 file. As you edit source in any of these formats, CODE/370 maintains the sequence numbers and date stamps.

This method of workstation/host tool integration offers the best use of the two environments: the S/370 subsystem environments′ ″live″ debug capabilities and the workstation GUI′s easy editing capabilities. The integration also allows programming shops to grow at their own pace into developing host applications on workstations.

When programmers use the Debug Tool, the debug session is recorded in a log file, permitting edit and replay of a Debug Tool session. This allows the Debug Tool to be used to capture test cases (for future program validation) or to further isolate a problem within an application. This also allows both interactive and batch debugging of a programmer′s application.

CODE/370 provides:

- Support for COBOL/370 Rel 1, COBOL for MVS & VM Rel 2, C/370 and PL/I MVS & VM
- Limited support for VS COBOL II Rel 3.1, 3.2, and 4.0 and OS PL/I Ver 2 Rel 1, 2, and 3
- A 32-bit, user-programmable editor
- Several language-sensitive editing features for REXX and JCL
- Enhanced support for debugging under CICS:
  - Pseudo conversational transaction support for COBOL applications
  - Support for the full range of single-terminal Send and Receive messages
- Program Generator, an independent compile/link program that allows compiling from inside and outside the editor
- REXX and JCL programs can be submitted to the host from the Editor window
- An OS/2 desktop tool called WorkFrame/2
- Advanced Program to Program Communication (APPC) protocol support for cooperative sessions between the MVS host and the workstation
- Support for debugging COBOL applications consisting of multiple enclaves and multiple processes
- Debug Tool support for exception handling of COBOL IGZ exceptions
- The ability to perform initial installation of CODE/370′s workstation feature from a LAN server.

Other Host COBOL Compilers for MVS & VM

IBM has developed three COBOL products for the MVS & VM mainframe: OS/VS COBOL, VS COBOL II, and COBOL for MVS & VM.

OS/VS COBOL was withdrawn from marketing in June of 1992 and withdrawn from service in June of 1994. This section discusses issues involved with migrating from OS/VS COBOL. The two COBOL mainframe products that will continue to be available are COBOL for MVS & VM and VS COBOL II.

Migrating from OS/VS COBOL: As announced in June 1992, OS/VS COBOL products are discontinued, effective June 1994. A service extension for OS/VS COBOL is available in certain geographic locations.

Because OS/VS COBOL (program product # 5740-CB1) has been discontinued, we are encouraging companies to upgrade their COBOL technology to COBOL for MVS & VM (compiler) with Language Environment for MVS & VM (run-time library). This is especially important for your Year2000 solution providing 4-digit year support. Depending on which product is currently being used and how fast a company is willing to migrate, there are three possible migration paths that you can follow.

Migration Paths: The paths are:

1. OS/VS COBOL------------------ COBOL for MVS & VM

   Companies with OS/VS COBOL are encouraged to migrate directly to IBM COBOL for MVS & VM (compiler) and IBM Language Environment for MVS & VM (run-time library) if the IBM Language Environment for MVS & VM prerequisites are satisfied. See the IBM Language Environment for MVS & VM Licensed Program Specification for prerequisite information.

2. OS/VS COBOL---- VS COBOL II---- COBOL for MVS & VM

   Companies with OS/VS COBOL who do not yet satisfy the IBM Language Environment for MVS & VM prerequisites must migrate to VS COBOL II first. When the IBM Language Environment for MVS & VM prerequisites are satisfied, then a company can move to IBM COBOL for MVS & VM.

Each successive COBOL product contains more capabilities than the previous product.

VS COBOL II: VS COBOL II (program product # 5668-958) builds on the functions of OS/VS COBOL but has a variety of features that give companies many advantages over earlier IBM COBOL products. VS COBOL II includes additional language features, enhanced compiler options, virtual storage constraint relief, structured programming language for improved programmer productivity, enhanced double-byte character set (DBCS) support, streamlined system interfaces, expanded code optimization, flexible run-time options, and support for VSE/ESA′s 31-bit addressing feature.

Companies that do not yet satisfy the IBM Language Environment for MVS & VM prerequisites should run VS COBOL II.

Migration Path:

VS COBOL II---- COBOL for MVS & VM

Companies with VS COBOL II can migrate to IBM COBOL for MVS & VM once the IBM Language Environment for MVS & VM prerequisites are satisfied.

Benefits of COBOL Migration:OS/VS COBOL is the ANSI 74 compiler. ANSI 85 introduced many significant functions which are provided to customers in either VS COBOL II or COBOL for MVS & VM and its companion product, COBOL for VSE. Customers who have migrated to the newer COBOL Standard have additional functionality, increased developer productivity and exploitation of S/390 hardware capabilities. Some benefits of upgrading COBOL technology are:

- Improved Interlanguage Communication (ILC)
- Condition management features of Language Environment, which bring PL/I-like condition handling to COBOL and C in MVS & VM
- Language Environment callable services, including a date/time service routine that interprets a 2-digit year to a 4-digit year to accommodate the year 2000
- Improved application performance of COBOL for MVS & VM compared to VS COBOL II
- Increased maintenance productivity from restructuring tools such as IBM′s COBOL Structuring Facility
- CoOperative Development Environment/370 (CODE/370), which provides increased programmer productivity compared to traditional host development tools
- Object-oriented extensions in COBOL for MVS.

For more information on the value of migrating from OS/VS COBOL or VS COBOL II to COBOL for MVS & VM, obtain a copy of *Why Migrate to COBOL/370 and LE/370?*, which is available from your IBM representative (COBMGVAL PACKAGE on MKTTOOLS). This document contains examples of customer benefits of migrating to COBOL for MVS & VM.

COBOL and CICS/VS Command Level Conversion Aid (CCCA)

COBOL and CICS/VS Command Level Conversion Aid (CCCA) (program product # 5785-ABJ) is an effective tool designed to make it easier to convert old COBOL source code and copy modules to the new COBOL Standard. CCCA converts OS/VS COBOL, DOS/VS COBOL, and COBOL 74 Standard VS COBOL II (either VS COBOL II Release 1 and 2 or VS COBOL II Release 3 and 4 (CMPR2)) source code to COBOL 85 Standard VS COBOL II Release 3 or 4 (NOCMPR2) or to IBM COBOL for MVS & VM.

In cases where a statement is no longer supported and has no equivalent statement in the target COBOL, CCCA flags the statement. CCCA can be used to convert from OS/VS COBOL to COBOL for MVS & VM, just as it is used to convert from OS/VS COBOL to VS COBOL II. The source file output for compiling under VS COBOL II can also be used for compiling under COBOL for MVS & VM.

When converting from OS/VS COBOL, CCCA can be automatically invoked by COBOL/SF, a re-engineering tool. This lets users convert programs before structuring them with COBOL/SF.

CCCA is designed to identify and convert source code incompatibility, to reduce the effort required to convert programs, and to minimize conversion errors. The conversion process can be customized by users to meet unique conversion requirements. Installation and usage are easy, fast, and straightforward.

CCCA key benefits are:

- Identification and conversion of source code
- Reduction of the effort required to convert programs
- Minimization of conversion errors
- Enhanced programmer productivity during migration.

CCCA provides facilities to:

- Convert most syntax differences between OS/VS COBOL, DOS/VS COBOL, or VS COBOL II Release 1 or 2 and the current release of VS COBOL II and COBOL for MVS & VM programs
- Convert EXEC CICS commands
- Remove and/or convert the base locator for linkage (BLL) section mechanism and references
- Eliminate conflicts between user-defined names and words reserved for VS COBOL II
- Convert both source programs and copy modules
- Create conversion management reports
- Produce a statement-by-statement diagnostic listing showing the result of the conversion process for each program
- Change and/or create COBOL conversion modules
- Allow foreground conversion of CICS programs
- Perform conversion from various levels of COBOL into other COBOL levels through an open converter design
- Read from PDSEs, not just PDSs.

COBOL Structuring Facility

IBM COBOL Structuring Facility/MVS & VM (COBOL/SF) (program product # 5696-737) reduces the amount of time needed to maintain code by automatically transforming complex unstructured programs into structured programs. Improved maintenance productivity frees programmers to focus on creating new applications. COBOL/SF promotes application redevelopment by providing a set of complexity metrics for inventory analysis. Information provided on program complexity and control flow can promote code reuse and speed up the development of new applications.

COBOL/SF provides a connection to COBOL and CCCA that automates the conversion of COBOL code to a higher-level standard and new COBOL technology. COBOL/SF also provides access to VIA/Renaissance (ViaSoft), which does program slicing to extract program logic from COBOL source code and generate self-contained program modules. These three tools can be used together to automatically convert, restructure, and modularize COBOL applications for use in maintenance, code sharing, new development or modularizing any date calculation or processing.

COBOL/SF automatically produces a structured COBOL program from unstructured source code. In addition, statistical metrics, structure charts to aid in program understanding, and modularization analysis reports are provided.

Three steps form the basis for creating program parts that can be used in a client/server application architecture: converting OS/VS COBOL to VS COBOL II or to COBOL for MVS & VM, restructuring, and modularizing to create functional program units. If the functional units are wrapped with an object-oriented COBOL wrapper, then developers can use these primitive parts for new OO applications.

COBOL Structuring Facility provides:

- Analysis of potential problem areas in source code and expert advice on how to manually improve source code quality
- Improved reporting capability and visual program display
- An on-line tutorial
- Comprehensive on-line documentation
- Automatic structuring of COBOL code containing:
  - CICS HANDLE commands
  - COBOL for MVS & VM intrinsic functions
  - Language Environment for MVS & VM support
- Cross-reference browser
- Conformance to CUA 1991 standards
- DBCS support consistent with IBM COBOL for MVS & VM enablement.

COBOL/SF is more than a one-time restructuring tool. It can be used regularly to maintain structured programs for ease of maintenance and program understanding. By offering modularization advice, COBOL/SF also proves useful as a redevelopment tool to aid in isolating reusable program functions such as standard date routines. Programming experts recommend restructuring an application whenever 10 to 15% of a program has changed due to a program error or enhancements.

CICS Application Migration Aid

The CICS Application Migration Aid (program product # 5695-061) is designed to assist developers in converting CICS applications from the macro-level API to the command-level API. Applications written in assembler or COBOL can be used with the migration aid.

Old CICS transactions used a macro-level interface. However, CICS/ESA V3.3 (which is required when using Language Environment for MVS & VM) does not support the macro level. Command level is a requirement for applications to run on CICS/ESA V3.

The CICS Application Migration Aid simplifies and speeds the conversion of COBOL and Assembler language application programs from macro to command level. The tool completely converts simple macros and provides guidance on converting more complex macros.

Conversion is key not only to using Language Environment for MVS & VM but also to obtain the benefits of the CICS command-level application interface (API), which is common across the CICS family.

COBOL Report Writer Precompiler

The COBOL Report Writer Precompiler (program offering # 5798-DYR) has two functions. It can permanently convert Report Writer statements to valid COBOL statements that can be compiled in IBM COBOL for MVS & VM or IBM COBOL for VSE/ESA. Or, it can be used to precompile applications containing Report Writer statements so the code will be acceptable to the IBM COBOL for MVS & VM or IBM COBOL for VSE/ESA compiler.

When used to precompile, the Precompiler automatically invokes the IBM COBOL compiler – as though Report Writer statements in the source program are being processed by the IBM COBOL for MVS & VM or IBM COBOL for VSE/ESA compiler itself. The fact that two separate processes are involved is transparent to users.

EDGE Portfolio Analyzer - COBOL Migration Tool

The EDGE Portfolio Analyzer Version 1 (program product # 5633-009) can significantly reduce the effort necessary to migrate from an earlier version of an IBM host COBOL compiler to the latest level of this product.

This migration tool analyzes existing application load libraries and provides data and statistical information about:

- Which release of a compiler and what compiler and linkage editor options are used to produce an existing load module
- Application programs:
  - Containing shared subroutines
  - Requiring re-linking of runtime modules as a result of any arbitrary change
  - Impacted by implementation of a new runtime package
  - Not expected to be affected by the implementation of a new runtime package
  - Containing interlanguage dependencies that may prove obstacles to migration
- Load modules not conforming to the installation's established compiler and linkage editor standards

Customers can save many hours of tedious investigation and move more quickly into the new compiler products.

Workstation Interactive Test Tool (WITT)

Workstation Interactive Test Tool (WITT) Product Family currently consists of WITT for OS/2 (program product # 5648-031), WITT for AIX (program product # 5765-222), WITT for Windows (program part # 10H7744), WITT for SUN (program product # 5765-407), and WITT for HP (program product # 5765-406).

COBOL programmers can use the Workstation Interactive Test Tool (WITT) as a GUI test tool that automatically facilitates the creation of reusable unit test, function test, system test, and regression test cases. These reusable test cases can be created when the program is newly developed and should be run continuously during the years that the program is maintained. Once created, the test cases can be played back interactively or in an unattended batch mode. The execution of a set of WITT test cases provides an audit trail of errors that a tester can provide to a programmer to document program failures.

WITT automatically records and plays back all keystroke and mouse movements and compares and identifies test inconsistencies between benchmark test cases and test cases run after enhancements and fixes are made to a program. WITT also allows scripting in 2/REXX to add program intelligence to the test cases. This makes WITT test cases easy to update, maintain, and reuse for future testing.

WITT/OS2 installs on an OS/2 desktop and allows the testing of MVS, VM, IMS, CICS, OS/400, and OS/2 PM and Text (ie, Micro Focus, CICS/OS2) applications. WITT/PM installs on an OS/2 desktop and allows the testing of only OS/2 PM and Text applications. X/WITT installs on an AIX desktop and allows the testing of AIX X Window applications, as well as remote client applications in X Windows environments on SUN, HP, APOLLO, and MVS. WITT/Windows installs on a Windows 3.1 desktop and allows the testing of Windows workstation applications.

ReDiscovery

ReDiscovery Product Family consists of ReDiscovery/2 (program product # 5871-AAA 70G3659) and ReDiscovery/MVS (program product # 5655-067).

COBOL developers can reuse code to minimize new coding and enable code sharing between applications. This reduces the cost of application development, test, and maintenance while helping to improve code quality. ReDiscovery is a tool that facilitates the introduction of a reuse initiative into application development by cataloging parts/modules and improves productivity. Reusing information in complex systems that span multiple platforms is extremely challenging. There are a large number of modules to sort through when maintaining an application. Some research, design, and development of new tools or other processes may be duplicated. By using function and data that already exist, programmers can increase the speed of new development and maintenance.

ReDiscovery is a productivity enhancement tool that allows programmers to organize and manage information about each file on their computer system. Some examples of files are: software parts and components, test cases, data files, source files for documents and manuals, JCL files, etc.

By using ReDiscovery to quickly locate parts and the attributes that describe them, programmers can perform some time-consuming and complex tasks quickly and easily. For example:

- **Maintaining Applications** - identifying all modules and data that will require modification due to an error or new requirement in the existing application
- **Language Migration** - identifying all the instructions that need to change across all the modules that are migrating
- **Rightsizing** - identify parts regardless of platform as input to the client-server migration process and downsize, consolidate, or upsize.

ReDiscovery facilitates the introduction of a reuse initiative into an application development organization. Many applications in an organization may contain reusable functions that could reduce development time. Therefore, collecting, organizing, and sharing data about these applications is essential for a successful reuse program.

In addition, ReDiscovery can assist in locating date-related code by searching the ReDiscovery catalogs for keywords such as ‘YY’ or ‘YEAR’ or ‘YR’.

The IBM PL/I Family for MVS & VM

For application code written in PL/I for S/370 and S/390 (for MVS and VM) platforms, IBM has developed PL/I compilers. These products target the host application development environment.

IBM′s PL/I mainframe products for S/370 and S/390 are:

- **PL/I for MVS & VM** – the compiler
- **Language Environment for MVS & VM** (new name for Language Environment/370) – the run-time library
- **CoOperative Development Environment** (CODE/370) – edit/compile/debug tool

Features

This PL/I compiler, PL/I for MVS & VM, is available today and provides full 4-digit year support with features including:

- Built-in functions
- Sliding century window

PL/I for MVS & VM provides full Year2000 support. It provides Built-in Function which provides 4-digit-year support. Language Environment for MVS & VM provides additional date manipulation with the Language Environment Callable Services. PL/I for MVS & VM will return a 4-digit year when the PL/I application program queries the system for the current date.

Companies need productive application development environments. A language compiler is only a portion of what application programmers need today to develop and maintain code. Tools that generate statistics from code; modularize, migrate, or restructure code; or search class libraries; are becoming more and more important.

These tools assist in developing PL/I applications with PL/I for MVS & VM:

- **EDGE Portfolio Analyzer** – load inventory tool
- **WITT** Product Family – testing

IBM PL/I for MVS & VM

IBM PL/I for MVS & VM (program product # 5688-235) is the high-performance PL/I compiler for the MVS/ESA and VM/ESA environments.

PL/I for MVS & VM provides the capability of integrating PL/I applications into IBM Language Environment for MVS & VM. Language Environment for MVS & VM, a common run-time environment, supports PL/I for MVS & VM, COBOL for MVS & VM, and C/C++ for MVS/ESA. Together, Language Environment and its supported languages create a common run-time environment. This integration allows you to take advantage of features from both PL/I and Language Environment. In addition, common function across the supported languages and platforms improves usability as well as programmer productivity.

Applications written using PL/I for MVS & VM can interface with a variety of IBM products, such as SQL/DB, DB2, CICS, IMS, and Data Window Services available on MVS/ESA. The consistent interlanguage communication support, common protocols, and suite of callable services provided by Language Environment for MVS & VM is designed to allow easier access to in-house applications or vendor packages written in PL/I for MVS & VM.

IBM PL/I for MVS & VM together with Language Environment for MVS & VM provide:

- Consistent interlanguage communications, common services, and common functions, which helps extend the useful life of existing applications
- Improved dynamic calls
- Support for year 2000
- Capabilities to help application programmers incrementally enhance applications
- Help in maintaining and enhancing the investment in existing programmer skills

IBM PL/I for MVS & VM provides facilities to acquire and integrate packaged software, consistent with the vendor′s terms, into existing applications irrespective of the language used; use existing code in new applications (code reuse) regardless of the source code language used; and invoke functionality between applications with improved interlanguage communication (ILC).

IBM PL/I for MVS & VM includes enhanced compiler options, virtual storage constraint relief, structured programming language for improved programmer productivity, enhanced double-byte character set (DBCS) support, streamlined system interfaces, expanded code optimization, and flexible run-time options.

IBM Language Environment for MVS & VM

IBM Language Environment for MVS & VM (program product # 5688-198) is IBM′s common runtime environment for enterprise applications written in COBOL, PL/I, C, and FORTRAN. Language Environment for MVS & VM is designed to provide defined calling conventions, enhanced interlanguage communication, callable services, language-specific services (for example, common facilities messages), common math functions, application utilities, system services, and subsystem support for Customer Information Control System (CICS) and Information Management System (IMS).

With Language Environment for MVS & VM, application programmers can extend and integrate their applications and packages, as well as reuse code with greater flexibility, due to interlanguage communication and native language restrictions. Application packages may be extended with the language of choice. Language Environment for MVS & VM enables existing applications to function as before with little, if any, changes required, thus helping preserve company investment in those applications. Language Environment for MVS & VM condition handler permits programs to handle errors in a predictable, logical manner without highly-specialized routines.

Language Environment for MVS & VM replaces the existing language-specific run-time libraries and provides a common runtime environment for all languages that conform to the Language Environment architecture. In a single product, Language Environment for MVS & VM combines essential run-time services, such as routines for message handling, condition handling, and storage management. All these services are available through a set of interfaces that are consistent across programming languages. With Language Environment for MVS & VM, application programmers can use one runtime environment for their applications, regardless of the programming language or system resource needs.

Today, IBM Language Environment for MVS & VM is the run-time library for the Language Environment-enabled compilers IBM COBOL for MVS & VM, C/C++

for MVS/ESA, and PL/I MVS & VM. IBM realizes the importance our customers place on complete, open, high-quality solutions. Therefore, IBM has established an *IBM Language Environment Partner Program* at the IBM Santa Teresa Laboratory. This program encourages and assists tool and application package vendors to support and take advantage of Language Environment services. Companies with existing 3GL applications will benefit from the wider choice of tools and application packages.

Language Environment for MVS & VM provides a number of advantages over other packages; its capabilities include:

- Ability to parse dates in an infinite number of formats by using a picture string feature as a parsing guide
- Provide NLS support by using a built-in table of defaults based on a country code
- A sliding window feature

  IBM Language Environment for MVS & VM provides a valuable short term solution with the century window feature. If you are unable to change all of your applications and data at the same time, the century window feature allows 2-digit years to be interpreted in a 100-year window. Any 100-year window can be selected. You pass a 2-digit year to Language Environment and Language Environment returns a 4-digit year based on the 100-year window.

  The advantage to the century window is that you need to change only the application code and not the databases or files with 2-digit years. This allows you to change the application programs one at a time or groups at a time without effecting your data files. Note, this only works for dates that range less than 100 years. For example, dates of birth may not be appropriate for this solution; a person born in '94' could be over 100 years of age.

  The disadvantage to the century window is that it doesn′t last forever. If your application has a long life expectancy, you many need to go back and replace the century window with full 4-digit year support. Some of your applications will be replaced prior to this replacement.

IBM CoOperative Development Environment/370 (CODE/370)

IBM CoOperative Development Environment/370 (CODE/370) (program product # 5688-194) — the common editor, compiler, debug tool — provides a cooperative environment, allowing application programmers to more productively develop and maintain host IBM 3GL applications from the workstation. CODE/370 provides a consistent, graphical user interface across different platforms and languages, a language-sensitive editor, language-sensitive help, a compiler invocation facility, and an interactive debug tool. CODE/370 combines the richness of the S/370 or S/390 subsystem environments and the power of IBM Language Environment for MVS & VM to provide a host Debug Tool which allows programmers to find bugs, fix bugs, and test applications. The Debug Tool is available either as a stand-alone 3270 host debug tool (for programming shops where workstations are not available) or with an optional graphical workstation user interface.

CODE/370′s cooperative environment allows application programmers to perform host programming tasks, such as compiling and debugging, from a workstation. Through cooperative processing, users perform functions locally on the workstation while interactively accessing the programs, data, and compilers residing on a host system. The optional workstation interface combines the Edit and Compile / Link functions together with the Debug Tool graphical user interface (GUI).

The powerful workstation-based editor integrates a rich set of functions that will speed up your application-development activities. The editor works with any type of source. The language-sensitive features help optimize coding efficiency in COBOL, C, PL/I, REXX, and JCL. Source can be stored in an MVS data set, a VM file, or an OS/2 file. As you edit source in any of these formats, CODE/370 maintains the sequence numbers and date stamps.

This method of workstation/host tool integration offers the best use of the two environments: the S/370 subsystem environments′ ″live″ debug capabilities and the workstation GUI′s easy editing capabilities. The integration also allows programming shops to grow at their own pace into developing host applications on workstations.

When programmers use the Debug Tool, the debug session is recorded in a log file, permitting edit and replay of a Debug Tool session. This allows the Debug Tool to be used to capture test cases (for future program validation) or to further isolate a problem within an application. This also allows both interactive and batch debugging of a programmer′s application.

CODE/370 provides:

- Support for COBOL/370 Rel 1, COBOL for MVS & VM Rel 2, C/370 and PL/I MVS & VM
- Limited support for VS COBOL II Rel 3.1, 3.2, and 4.0 and OS PL/I Ver 2 Rel 1, 2, and 3
- A 32-bit, user-programmable editor
- Several language-sensitive editing features for REXX and JCL
- Enhanced support for debugging under CICS:
  - Pseudo conversational transaction support for COBOL applications
  - Support for the full range of single-terminal Send and Receive messages
- Program Generator, an independent compile/link program that allows compiling from inside and outside the editor
- REXX and JCL programs can be submitted to the host from the Editor window
- An OS/2 desktop tool called WorkFrame/2
- Advanced Program to Program Communication (APPC) protocol support for cooperative sessions between the MVS host and the workstation
- Support for debugging COBOL applications consisting of multiple enclaves and multiple processes
- Debug Tool support for exception handling of COBOL IGZ exceptions
- The ability to perform initial installation of CODE/370′s workstation feature from a LAN server.

Other Host PL/I Compilers for MVS & VM

IBM has developed three PL/I products for the MVS and VM mainframe: OS PL/I V1, OS PL/I V2, and PL/I for MVS & VM.

OS PL/1 V1 will be withdrawn from service in December of 1995. This section discusses issues involved with migrating from OS PL/I V1. The two PL/I mainframe products that will continue to be available are PL/I for MVS & VM and OS PL/I V2.

Because OS PL/I V1 (5734-PL1, 5734-PL2, 5734-PL3, 5734-LM4) will be discontinued, we are encouraging companies to upgrade their PL/I technology to PL/I for MVS & VM (compiler) with Language Environment for MVS & VM (run-time library). Depending on which product is currently being used and how fast a company is willing to migrate, there are three possible migration paths that can be followed.

Migration Paths: The three paths are:

1. OS PL/I V1------------------ PL/I for MVS & VM

   Companies with OS PL/I V1 are encouraged to migrate directly to IBM PL/I for MVS & VM (compiler) and IBM Language Environment for MVS & VM (run-time library) if the IBM Language Environment for MVS & VM prerequisites are satisfied. See the IBM Language Environment for MVS & VM Licensed Program Specification for prerequisite information.

2. OS PL/I V1---- OS PL/I V2---- PL/I for MVS & VM

   Companies with OS PL/I V1 who do not yet satisfy the IBM Language Environment for MVS & VM prerequisites must migrate to OS PL/I V2 first. When the IBM Language Environment for MVS & VM prerequisites are satisfied, then move to IBM PL/I for MVS & VM.

3. OS PL/I V2---- PL/I for MVS & VM

   Companies with OS PL/I V2 can migrate to IBM PL/I for MVS & VM once the IBM Language Environment for MVS & VM prerequisites are satisfied.

EDGE Portfolio Analyzer - PL/I Migration Tool

The EDGE Portfolio Analyzer Version 1 (program product # 5633-009) can significantly reduce the effort necessary to migrate from an earlier version of an IBM host PL/I compiler to the latest level of this product.

This migration tool analyzes existing application load libraries and provides data and statistical information about:

- Which release of a compiler and what compiler and linkage editor options are used to produce an existing load module
- Application programs:
  - Containing shared subroutines
  - Requiring re-linking of runtime modules as a result of any arbitrary change
  - Impacted by implementation of a new runtime package
  - Not expected to be affected by the implementation of a new runtime package
  - Containing interlanguage dependencies that may prove obstacles to migration
- Load modules not conforming to the installation's established compiler and linkage editor standards

Customers can save many hours of tedious investigation and move more quickly into the new compiler products.

Workstation Interactive Test Tool (WITT)

Workstation Interactive Test Tool (WITT) Product Family currently consists of WITT for OS/2 (program product # 5648-031), WITT for AIX (program product # 5765-222), WITT for Windows (program part # 10H7744), WITT for SUN (program product # 5765-407), and WITT for HP (program product # 5765-406).

COBOL programmers can use the Workstation Interactive Test Tool (WITT) as a GUI test tool that automatically facilitates the creation of reusable unit test, function test, system test, and regression test cases. These reusable test cases can be created when the program is newly developed and should be run continuously during the years that the program is maintained. Once created, the test cases can be played back interactively or in an unattended batch mode. The execution of a set of WITT test cases provides an audit trail of errors that a tester can provide to a programmer to document program failures.

WITT automatically records and plays back all keystroke and mouse movements and compares and identifies test inconsistencies between benchmark test cases and test cases run after enhancements and fixes are made to a program. WITT also allows scripting in 2/REXX to add program intelligence to the test cases. This makes WITT test cases easy to update, maintain, and reuse for future testing.

WITT/OS2 installs on an OS/2 desktop and allows the testing of MVS, VM, IMS, CICS, OS/400, and OS/2 PM and Text (ie, Micro Focus, CICS/OS2) applications. WITT/PM installs on an OS/2 desktop and allows the testing of only OS/2 PM and Text applications. X/WITT installs on an AIX desktop and allows the testing of AIX X Window applications, as well as remote client applications in X Windows environments on SUN, HP, APOLLO, and MVS. WITT/Windows installs on a Windows 3.1 desktop and allows the testing of Windows workstation applications.

DFSORT

DFSORT V1R13 will enhance its Year2000 capabilities by providing the ability to sort, merge, and transform 2-digit years according to a specified sliding or fixed century window. New Y2C, Y2Z, Y2P, and Y2D formats, in conjunction with a new Y2PAST installation and run-time option, allow you to handle 2-digit year data in the following ways:

- Set the appropriate century window for your applications. For example, set a century window of 1915-2014 or 1950-2049.
- Order 2-digit character, zoned decimal, packed decimal, or decimal year data, according to the century window, using DFSORT′s SORT and MERGE control statements. For example, order 96 (representing 1996) before 00 (representing 2000) in ascending sequence, or order 00 before 96 in descending sequence.
- Transform 2-digit character, zoned decimal, packed decimal, or decimal year data to 4-digit character year data, according to the century window, using DFSORT′s OUTFIL control statement. For example, transform 99 to 1999 and 04 to 2004.

These DFSORT enhancements allow you to continue to use 2-digit years for sorting and merging, and assist those situations when you want to change 2-digit-year data to 4-digit-year data.

Additional information about DFSORT/MVS and its year2000 enhancements is available on the World Wide Web at URL:

http://www.storage.ibm.com/storage/software/sort/srtmhome.htm

Sliding Century Window

A new installation and run-time option allows you to specify a sliding or fixed century window to be used with 2-digit years. Y2PAST=s specifies a **sliding** century window starting s years before the current year. For example, if the current year is 1996, Y2PAST=80 starts the century window at 1996 − 80 = 1916, providing a century window of 1916 through 2015. In 1997, this century window automatically slides to 1917 through 2016.

Y2PAST=f specifies a **fixed** century window starting at f. For example, Y2PAST=1950 starts the century window at 1950, providing a century window of 1950 through 2049. Thus, Y2PAST allows you to control how DFSORT interprets the 2-digit years 00-99 on a site-wide or application-specific basis.

As an example, both Y2PAST=1915 and Y2PAST=81 used in 1996 give a century window of 1915 through 2014, and result in the following interpretation of 2-digit year formatted data by DFSORT:

yy	Interpreted as:
00	2000
14	2014
15	1915
61	1961
62	1962
99	1999

2-Digit Year Formats

New formats allow you to identify 2-digit character, zoned decimal, packed decimal and decimal year data for special DFSORT processing as follows (yy represents 2-digit year data in the examples below):

Format	Meaning
Y2C	identifies 2-digit, 2-byte character year data such as C‘yy’, C‘mm/dd/yy’, or C‘yy.mm.dd’
Y2Z	identifies 2-digit, 2-byte zoned decimal year data such as Z‘yy’, Z‘mmddyy’, or Z‘yymmdd’
Y2P	identifies 2-digit, 2-byte packed decimal year data such as P‘yy’, P‘dddyy’, or P‘yymmdd’
Y2D	identifies 2-digit, 1-byte decimal year data such as X‘yy’ or P‘yyddd’

Sorting and Merging 2-Digit Years

You can use the new Y2C, Y2Z, Y2P, and Y2D formats in DFSORT′s SORT and MERGE statements to identify specific 2-digit year data to be ordered according to the century window.

A simple example of the control statements to sort a C‘mm/dd/yy’ field (assume the current year is 1996) follows:

```
* Set the century window to 1962 through 2061
  OPTION Y2PAST=34
* Sort C′mm/dd/yy′ as C′yymmdd′
  SORT FIELDS=(7,2,Y2C,A,  * sort yy using century window
               1,2,CH,A,   * sort mm
               4,2,CH,A)   * sort dd
```

These control statements provide the following sort results:

Input Data (CH)	Sorted Output Data (CH)
06/22/15	03/18/62
10/03/00	09/01/99
11/14/61	10/03/00
08/16/14	08/16/14
09/01/99	08/17/14
03/18/62	06/22/15
08/17/14	11/14/61

Transforming 2-Digit Years to 4-Digit Years

You can use the new Y2C, Y2Z, Y2P, and Y2D formats in the OUTREC operand of DFSORT′s OUTFIL statement to identify 2-digit year data to be changed to 4-digit year data according to the century window.

A simple example of the control statements to transform a P′yyddd′ field follows:

```
* Set the century window to 1970 through 2069
  OPTION COPY,Y2PAST=1970
* Change P′yyddd′ to C′yyyy/ddd′
 OUTFIL FNAMES=Y4,
  OUTREC=(1,1,Y2D,        * change X′yy′ to C′yyyy′ using
*                         *   century window
          C′/′,           * insert C′/′
          2,2,PD,M11)     * change P′ddd′ to C′ddd′
```

This code provides the following transformation results:

Input Data (HEX)	Transformed Output Data (CH)
92012F	1992/012
70225C	1970/225
69153F	2069/153
00001F	2000/001
99321F	1999/321
12054C	2012/054

COMUDAS (COMmon Uithoorn DAte Services)

COMUDAS (program product # 5788-HBB) is a common date routine that has been developed to replace all existing date routines, used by the IBM Uithoorn Lab, The Netherlands. COMUDAS offers all necessary functions for validation, conversion, and calculation of dates in any format. A standard interface must be used to call the date routine′s load module dynamically.

The package also offers the possibility to use separate functions by means of NCAL′s, that can be linked statically. This might be useful when converting large files or databases with validated data into other formats (for example, when reformatting year-date notation in an application).

Functions are available for updating the Calendar Tables by means of the Online Facility, that also can be used to test the Date Routines, and to print calendars.

A CICS version, as well as an MVS version, of this package is available.

Features

COMUDAS provides features including:

- Date validation
- Date calculation
- Date conversion
- Free date-format definitions
- Supports country-dependent data
  - Weekend definitions
  - Closing dates (for both manufacturing and fiscal purposes)
  - Holidays
- Can be used by MVS applications with PL/1 runtime environment or a Language Environment, for:
  - CICS (PL/1)
  - PL/1
  - COBOL
- Supports terms as:
  - Country codes
  - Shopdates (numbering of working days)
  - Production months
  - Closing dates (logistics/manufacturing & finance)
- Is able to print:
  - Common calendars for years in range 1760 through 9999
  - Production calendars (per country/year)
- Contains functions to support a calendar owner
- Can be used as a tool when changing applications to handle the year 2000.

Components

COMUDAS consists of two main components. First is the date routine functions. An interface is supplied that can be filled with data, and returned to the calling program, containing the requested information, including a return code and return message. The interface is available in both PL/1 and COBOL format. The date routine functions can be called by applications:

- CICS (PL/1), for example, HPS applications
- PL/1
- COBOL

The second component, the online facility, used in a TSO/ISPF environment, can be used to:

- Get familiar with the date routines.

  An online presentation of the interface can be used to test the date routine functions.
- Update the calendar tables.

  All users (with read access) may view the contents of the calendar tables, and the calendar owner can use this function to:
  - Update the calendar tables
  - Generate a new table to be used by the date routine functions
  - Generate production calendars
- Print a calendar.

  All users will be able to print a common calendar for years in range 1760 through 9999.
- Print a Production Calendar.

  All users will be able to print a production calendar for a country and year, that must have been created by the calendar owner previously.

Using COMUDAS

When using COMUDAS, the following terms can be handled by the package:

- Day of the week (number, name, 3-character abbreviation)
- Day of the month (number)
- Day of the year (Julian notation)
- Week of the year (number)
- Production Month (number)
- Month of the year (number, name, 3-character abbreviation)
- Year (two or four digits)
- Year (related to a weeknumber; this value may differ from the earlier mentioned year-field, because weeks do not always start on January 1)
- Shopdates
- SYSDATE
- European format of a date
- ISO (International Standards Organization) format of a date
- USA format of a date
- JIS (Japanese Industrial Standard) format of a date
- Remainder days of a subtraction of two dates
- Day number suffix (for example, 10th, 23rd)
- First shopdate of a week
- Last shopdate of a week
- Startdate of a production month
- Closing date of a production month
- Closing date finance
- Addition / Subtraction fields:
  - +/- days
  - +/- weeks
  - +/- weeks and days
  - +/- months
  - +/- months and days
  - +/- years, months and days
  - +/- shopdays

Special Ordering Information

To order COMUDAS, your IBM marketing representative must order the package through a formal note to:

Vnet: COMUDAS at UITVM1

IBM Tools for VM

The IBM COBOL Family for MVS & VM

For application code written in COBOL for S/370 and S/390 (for MVS and VM) platforms, IBM has developed COBOL compilers. These products target the host application development environment.

IBM′s COBOL mainframe products for S/370 and S/390 are:

- **COBOL for MVS & VM** (new name for IBM COBOL/370) – the compiler
- **Language Environment for MVS & VM** (new name for Language Environment/370) – the run-time library
- **CoOperative Development Environment** (CODE/370) – edit/compile/debug tool

See "The IBM COBOL Family for MVS & VM" on page 7-7 for detailed information about these COBOL tools.

The IBM PL/I Family for MVS & VM

For application code written in PL/I for S/370 and S/390 (for MVS and VM) platforms, IBM has developed PL/I compilers. These products target the host application development environment.

IBM′s PL/I mainframe products for S/370 and S/390 are:

- **PL/I for MVS & VM** – the compiler
- **Language Environment for MVS & VM** (new name for Language Environment/370) – the run-time library
- **CoOperative Development Environment** (CODE/370) – edit/compile/debug tool

See "The IBM PL/I Family for MVS & VM" on page 7-18 for detailed information about these PL/I tools.

REXX/EXEC Migration Tool for VM/ESA

The REXX/EXEC Migration Tool for VM/ESA (VM/ESA MIGR) is a tool that helps migrate REXX, EXEC2, ASSEMBLE, and other source files to the current release of VM/ESA.

VM/ESA MIGR does not make changes, but can assist the user in the following areas:

- Estimating the migration effort that is necessary.
- Identifying changes that have to be made.
- Finding commands, options, etc. which are incompatible or have been changed and presenting information about them through Help panels.

VM/ESA MIGR uses a keyword file (ESAMIGR SAMPLIST) to flag items which may cause incompatibilities. ESAMIGR SAMPLIST may be customized by the user to search for additional items or variations (such as command abbreviations) of items.

VM/ESA MIGR is shipped on a separate tape with the VM/ESA product. It is also available on MKTTOOLS, and through PUBORDER. Complete documentation is available in *REXX/EXEC Migration Tool for VM/ESA*, (GC24-5607).

IBM Tools for VSE

The IBM COBOL Family for VSE

For application code written in COBOL for S/370 and S/390 for VSE platform, IBM has developed COBOL compilers. These products target the host application development environment.

IBM's COBOL mainframe products for S/370 and S/390 VSE are:

- **COBOL for VSE/ESA** – the compiler
- **Language Environment for VSE/ESA** – the run-time library

Features

This COBOL compiler, COBOL for VSE/ESA, is available today and provides full 4-digit year support with features including:

- Intrinsic functions
- Sliding century window

COBOL for VSE/ESA provides full Year2000 support. It provides ANSI COBOL Standard Intrinsic Functions which give full date manipulation capability with 4-digit-year support. Language Environment for VSE/ESA provides additional date manipulation with the Language Environment Callable Services. COBOL for VSE/ESA will return a 4-digit year when the COBOL application program queries the system for the current date. DOS/VS COBOL and VS COBOL II return the system current date with a 2-digit year.

Companies need productive application development environments. A language compiler is only a portion of what application programmers need today to develop and maintain code. Tools that generate statistics from code; modularize, migrate, or restructure code; or search class libraries; are becoming more and more important.

These tools assist in migrating your existing COBOL applications to COBOL for VSE/ESA:

- **CCCA for VSE** – converts source code
- **COBOL Report Writer Precompiler** – supports Report Writer code

IBM COBOL for VSE/ESA

IBM COBOL for VSE/ESA (program product # 5686-068) is the high-performance compiler with a supporting runtime environment (Language Environment for VSE/ESA (program product # 5686-067), which facilitates multiple-language interaction. This is different from the single packaging of compiler and runtime environment for DOS/VS COBOL and VS COBOL II.

Applications written using COBOL for VSE/ESA can interface with a variety of IBM products, such as SQL/DS and CICS. The consistent interlanguage communication support, common protocols, and suite of callable services provided by Language Environment for VSE/ESA is designed to allow easier access to in-house applications or vendor packages written in COBOL for VSE/ESA.

IBM COBOL for VSE/ESA provides:

- Intrinsic functions, which reduce the need for extensive algorithms

- Programmer access to all of the elements in a table at once, reducing the need for explicit loops
- Consistent interlanguage communications, common services, and common functions, which helps extend the useful life of existing applications
- Improved dynamic calls
- Support for Year2000
- Capabilities to help application programmers incrementally enhance applications
- Help in maintaining and enhancing the investment in existing programmer skills.

IBM COBOL for VSE/ESA provides facilities to acquire and integrate packaged software, consistent with the vendor′s terms, into existing applications irrespective of the language used; use existing code in new applications (code reuse) regardless of the source code language used; and invoke functionality between applications with improved interlanguage communication (ILC).

IBM COBOL for VSE/ESA includes VS COBOL II language features, enhanced compiler options, virtual storage constraint relief, structured programming language for improved programmer productivity, enhanced double-byte character set (DBCS) support, streamlined system interfaces, expanded code optimization, flexible run-time options, and support for the VSE/ESA 31-bit addressing feature.

Most applications were coded with 2-digit-year data. COBOL for VSE/ESA provides intrinsic functions not available in our earlier COBOL compilers. Calendar functions include:

- CURRENT-DATE
- DATE-OF-INTEGER
- DAY-OF-INTEGER
- INTEGER-OF-DATE
- INTEGER-OF-DAY
- WHEN-COMPILED

Example of obtaining the 4-digit year with the COBOL Intrinsic Functions:

- DATE-OF-INTEGER gives YYYYMMDD
- DAY-OF-INTEGER gives YYYYDDD

IBM Language Environment for VSE/ESA

IBM Language Environment for VSE/ESA (program product # 5686-067) is IBM′s common runtime environment for enterprise applications written in COBOL and PL/I. Language Environment for VSE/ESA is designed to provide defined calling conventions, enhanced interlanguage communication, callable services, language-specific services (for example, common facilities messages), common math functions, application utilities, system services, and subsystem support for Customer Information Control System (CICS).

With Language Environment for VSE/ESA, application programmers can extend and integrate their applications and packages, as well as reuse code with greater flexibility, due to interlanguage communication and native language restrictions. Application packages may be extended with the language of choice. Language Environment for VSE/ESA enables existing applications to function as before with little, if any, changes required, thus helping preserve company investment in those applications. Language Environment for VSE/ESA condition handler permits programs to handle errors in a predictable, logical manner without highly-specialized routines.

Language Environment for VSE/ESA replaces the existing language-specific run-time libraries and provides a common runtime environment for all languages that conform to the Language Environment architecture. In a single product, Language Environment for VSE/ESA combines essential run-time services, such as routines for message handling, condition handling, and storage management. All these services are available through a set of interfaces that are consistent across programming languages. With Language Environment for VSE/ESA, application programmers can use one runtime environment for their applications, regardless of the programming language or system resource needs.

Today, IBM Language Environment for VSE/ESA is the run-time library for the Language Environment-enabled compilers IBM COBOL for VSE/ESA and PL/I for VSE/ESA. IBM realizes the importance our customers place on complete, open, high-quality solutions. Therefore, IBM has established an *IBM Language Environment Partner Program* at the IBM Santa Teresa Laboratory. This program encourages and assists tool and application package vendors to support and take advantage of Language Environment services. Companies with existing 3GL applications will benefit from the wider choice of tools and application packages.

Language Environment for VSE/ESA provides a number of advantages over other packages; its capabilities include:

- Ability to parse dates in an infinite number of formats by using a picture string feature as a parsing guide
- Provide NLS support by using a built-in table of defaults based on a country code
- A sliding window feature

  IBM Language Environment for VSE/ESA provides a valuable short term solution with the century window feature. If you are unable to change all of your applications and data at the same time, the century window feature allows 2-digit years to be interpreted in a 100-year window. Any 100-year window can be selected. You pass a 2-digit year to Language Environment and Language Environment returns a 4-digit year based on the 100-year window.

  The advantage to the century window is that you need to change only the application code and not the databases or files with 2-digit years. This allows you to change the application programs one at a time or groups at a time without effecting your data files. Note, this only works for dates that range less than 100 years. For example, dates of birth may not be appropriate for this solution; a person born in ‘94’ could be over 100 years of age.

  The disadvantage to the century window is that it doesn′t last forever. If your application has a long life expectancy, you many need to go back and replace the century window with full 4-digit year support. Some of your applications will be replaced prior to this replacement.

Other Host COBOL Compilers for VSE

IBM has developed three COBOL products for the VSE environment: DOS/VS COBOL, VS COBOL II, and COBOL for VSE/ESA. This section discusses issues involved with migrating from DOS/VS COBOL.

Migrating from DOS/VS COBOL: We are encouraging companies to upgrade their COBOL technology to COBOL for VSE/ESA (compiler) with Language Environment for VSE (run-time library). There are three possible migration paths that you can follow.

Migration Paths: The paths are:

1. DOS/VS COBOL------------------ COBOL for VSE/ESA

   Companies with DOS/VS COBOL are encouraged to migrate directly to IBM COBOL for VSE/ESA (compiler) and IBM Language Environment for VSE (run-time library) if the prerequisites are satisfied.

2. DOS/VS COBOL---- VS COBOL II---- COBOL for VSE/ESA

   Companies with DOS/VS COBOL who do not yet satisfy the IBM Language Environment for VSE/ESA prerequisites should migrate to VS COBOL II first. When the IBM Language Environment for VSE/ESA prerequisites are satisfied, then a company can move to IBM COBOL for VSE/ESA.

Each successive COBOL product contains more capabilities than the previous product.

VS COBOL II: VS COBOL II (program product # 5668-958) builds on the functions of DOS/VS COBOL but has a variety of features that give companies many advantages over earlier IBM COBOL products. VS COBOL II includes additional language features, enhanced compiler options, virtual storage constraint relief, structured programming language for improved programmer productivity, enhanced double-byte character set (DBCS) support, streamlined system interfaces, expanded code optimization, flexible run-time options, and support for VSE/ESA′s 31-bit addressing feature.

Migration Path:

VS COBOL II---- COBOL for VSE/ESA

Companies with VS COBOL II can migrate to IBM COBOL for VSE/ESA once the IBM Language Environment for VSE/ESA prerequisites are satisfied.

Customers who have VSE/ESA Version 2.1 or 1.4 installed and meet the prerequisites should migrate to IBM COBOL for VSE/ESA and Language Environment for VSE/ESA.

Benefits of COBOL Migration: DOS/VS COBOL is the ANSI 74 compiler. ANSI 85 introduced many significant functions which are provided to customers in either VS COBOL II or COBOL for VSE/ESA. Customers who have migrated to the newer COBOL Standard have additional functionality, increased developer productivity and exploitation of S/390 hardware capabilities. Some benefits of upgrading COBOL technology are:

- Improved Interlanguage Communication (ILC)
- Condition management features of Language Environment, which bring PL/I-like condition handling to COBOL

- Language Environment callable services, including a date/time service routine that interprets a 2-digit year to a 4-digit year to accommodate the year 2000
- Improved application performance of COBOL for VSE/ESA compared to VS COBOL II

For more information on the value of migrating from DOS/VS COBOL or VS COBOL II to COBOL for VSE/ESA, obtain a copy of ***Why Migrate to COBOL/370 and LE/370?***, which is available from your IBM representative (COBMGVAL PACKAGE on MKTTOOLS). This document contains some examples of customer benefits of migrating to COBOL/370 that also applies to the VSE environment.

COBOL and CICS/VS Command Level Conversion Aid (CCCA) for VSE

COBOL and CICS/VS Command Level Conversion Aid (CCCA) for VSE (program product # 5785-CCC) is an effective tool designed to make it easier to convert old COBOL source code and copy modules to the new COBOL Standard. CCCA converts DOS/VS COBOL and COBOL 74 Standard VS COBOL II (Release 3 and 4 (CMPR2)) source code to COBOL 85 Standard VS COBOL II Release 3 or 4 (NOCMPR2) or to IBM COBOL for VSE/ESA.

In cases where a statement is no longer supported and has no equivalent statement in the target COBOL, CCCA flags the statement. CCCA can be used to convert from DOS/VS COBOL to COBOL for VSE/ESA, just as it is used to convert from DOS/VS COBOL to VS COBOL II. The source file output for compiling under VS COBOL II can also be used for compiling under COBOL for VSE/ESA.

CCCA is designed to identify and convert source code incompatibility, to reduce the effort required to convert programs, and to minimize conversion errors. The conversion process can be customized by users to meet unique conversion requirements. Installation and usage are easy, fast, and straightforward.

CCCA key benefits are:

- Identification and conversion of source code
- Reduction of the effort required to convert programs
- Minimization of conversion errors
- Enhanced programmer productivity during migration.

CCCA provides facilities to:

- Convert most syntax differences between OS/VS COBOL, DOS/VS COBOL, or VS COBOL II Release 1 or 2 and the current release of VS COBOL II and COBOL for MVS & VM and COBOL for VSE/ESA programs
- Convert EXEC CICS commands
- Remove and/or convert the base locator for linkage (BLL) section mechanism and references
- Eliminate conflicts between user-defined names and words reserved for VS COBOL II
- Convert both source programs and copy modules
- Create conversion management reports
- Produce a statement-by-statement diagnostic listing showing the result of the conversion process for each program
- Change and/or create COBOL conversion modules
- Allow foreground conversion of CICS programs

- Perform conversion from various levels of COBOL into other COBOL levels through an open converter design
- Read from PDSEs, not just PDSs.

COBOL Report Writer Precompiler

The COBOL Report Writer Precompiler (program offering # 5798-DYR) has two functions. It can permanently convert Report Writer statements to valid COBOL statements that can be compiled in IBM COBOL for MVS & VM or IBM COBOL for VSE/ESA. Or, it can be used to precompile applications containing Report Writer statements so the code will be acceptable to the IBM COBOL for MVS & VM or IBM COBOL for VSE/ESA compiler.

When used to precompile, the Precompiler automatically invokes the IBM COBOL compiler – as though Report Writer statements in the source program are being processed by the IBM COBOL for MVS & VM or IBM COBOL for VSE/ESA compiler itself. The fact that two separate processes are involved is transparent to users.

The IBM PL/I Family for VSE

For application code written in PL/I for S/370 and S/390 for VSE platform, IBM has developed PL/I compilers. These products target the host application development environment.

IBM′s PL/I mainframe products for S/370 and S/390 VSE are:

- **PL/I for VSE/ESA** – the compiler
- **Language Environment for VSE/ESA** – the run-time library

Features

This PL/I compiler, PL/I for VSE/ESA, is available today and provides full 4-digit year support with features including:

- Built-in functions
- Sliding century window

PL/I for VSE/ESA provides full Year2000 support. It provides Built-in Function which provides 4-digit-year support. Language Environment for VSE/ESA provides additional date manipulation with the Language Environment Callable Services. PL/I for VSE/ESA will return a 4-digit year when the PL/I application program queries the system for the current date.

IBM PL/I for VSE/ESA

IBM PL/I for VSE (program product # 5686-069) is an implementation of IBM PL/I for MVS & VM and succeeds DOS PL/I for VSE - providing source code compatibility for most DOS PL/I for VSE R6 programs. PL/I for VSE requires and takes advantage of Language Environment for VSE capabilities to support use in mixed-language development environments and the use of reusable components in building applications. IBM PL/I for VSE provides:

- 31-bit virtual addressing
- Support for Language Environment for VSE functions
- Enhanced InterLanguage Communication (ILC) with COBOL for VSE

IBM Language Environment for VSE/ESA

IBM Language Environment for VSE/ESA (program product # 5686-067) is IBM's common runtime environment for enterprise applications written in COBOL and PL/I. Language Environment for VSE/ESA is designed to provide defined calling conventions, enhanced interlanguage communication, callable services, language-specific services (for example, common facilities messages), common math functions, application utilities, system services, and subsystem support for Customer Information Control System (CICS).

With Language Environment for VSE/ESA, application programmers can extend and integrate their applications and packages, as well as reuse code with greater flexibility, due to interlanguage communication and native language restrictions. Application packages may be extended with the language of choice. Language Environment for VSE/ESA enables existing applications to function as before with little, if any, changes required, thus helping preserve company investment in those applications. Language Environment for VSE/ESA condition handler permits programs to handle errors in a predictable, logical manner without highly-specialized routines.

Language Environment for VSE/ESA replaces the existing language-specific run-time libraries and provides a common runtime environment for all languages that conform to the Language Environment architecture. In a single product, Language Environment for VSE/ESA combines essential run-time services, such as routines for message handling, condition handling, and storage management. All these services are available through a set of interfaces that are consistent across programming languages. With Language Environment for VSE/ESA, application programmers can use one runtime environment for their applications, regardless of the programming language or system resource needs.

Today, IBM Language Environment for VSE/ESA is the run-time library for the Language Environment-enabled compilers IBM COBOL for VSE/ESA and PL/I for VSE/ESA. IBM realizes the importance our customers place on complete, open, high-quality solutions. Therefore, IBM has established an *IBM Language Environment Partner Program* at the IBM Santa Teresa Laboratory. This program encourages and assists tool and application package vendors to support and take advantage of Language Environment services. Companies with existing 3GL applications will benefit from the wider choice of tools and application packages.

Language Environment for VSE/ESA provides a number of advantages over other packages; its capabilities include:

- Ability to parse dates in an infinite number of formats by using a picture string feature as a parsing guide
- Provide NLS support by using a built-in table of defaults based on a country code
- A sliding window feature

  IBM Language Environment for VSE/ESA provides a valuable short term solution with the century window feature. If you are unable to change all of your applications and data at the same time, the century window feature allows 2-digit years to be interpreted in a 100-year window. Any 100-year window can be selected. You pass a 2-digit year to Language Environment and Language Environment returns a 4-digit year based on the 100-year window.

The advantage to the century window is that you need to change only the application code and not the databases or files with 2-digit years. This allows you to change the application programs one at a time or groups at a time without effecting your data files. Note, this only works for dates that range less than 100 years. For example, dates of birth may not be appropriate for this solution; a person born in '94' could be over 100 years of age.

The disadvantage to the century window is that it doesn't last forever. If your application has a long life expectancy, you many need to go back and replace the century window with full 4-digit year support. Some of your applications will be replaced prior to this replacement.

IBM Tools for AS/400

You can use the IBM products and tools listed below to help you address the changes required as you proceed with your Year2000 transition. The tools listed, however, represent only a subset of the AS/400 oriented offerings available to you for application development. A more complete list of available tools can be found in the *AS/400 Application Development Handbook*, (G325-6249).

The IBM RPG Family

For application code written in RPG, IBM has developed several RPG language compilers. These compilers target the host application development environment.

IBM′s RPG compilers for AS/400 are:

- Integrated Language Environment (ILE) RPG IV
- Original Program Model (OPM) RPG/400
- System/36 compatible compiler

All three RPG language compilers are available with either the Integrated Language Environment RPG for OS/400 Version 3 product (5716-RG1) or the IBM Integrated Language Environment RPG/400 Version 3 product (5763-RG1).

Using ILE RPG IV Date Support: The ILE RPG IV compiler has the following support for dates:

- Retrieval of the job date through special words UDATE and UYEAR. This level of support provides for a 2-digit year.
- Retrieval of the job date through special words *DATE and *YEAR. This level of support provides for a 4-digit year.
- Retrieval of the system date through operation TIME. This level of support provides for both 2-digit and 4-digit years.
- Date, Time, and Timestamp data type support. This level of support provides for date operations such as subtraction, addition, extraction, testing, comparision, and move.
- Calling OPM System APIs. This level of support provides for both 2-digit and 4-digit years.
- Calling ILE Date APIs. This level of support provides for both 2-digit and 4-digit years.
- SQL date, time, and timestamp data type support. This level of support provides for 4-digit years.

This section provides information on how to use RPG date, time, and timestamp fields. Specifically, it covers:

- The date, time, and timestamp data types and their formats
- User date special words
- How to edit date-time fields
- Date-time keywords
- Date-time operations and how to use them

Date-Time Data Types and Formats

Date, time and timestamp fields have an internal format that is independent of the external format. The **internal format** is the way the data is stored in the program. The **external format** is the way the data is stored in files.

You need to be aware of the internal format when:

- Passing parameters
- Overlaying subfields in data structures

There is a default internal format for date, time, and timestamp fields. In general, to change the internal format for a specific field, you must define the field and specify its internal format on a Definition specification. Similarly, to change (or specify) the external format for program-described fields, you specify the format on the corresponding Input or Output specificiation.

For fields in an externally described file, the external data format is specified in the data description specifications in position 35. You cannot change the external format of externally described date, time, and timestamp fields.

For subfields in externally described data structures, the data formats specified in the external description are used as the internal formats of the subfields by the compiler. The reason for this difference, is that a data structure, even if externally described, exists only when a program is running.

Internal Format: The default format for date, time, and timestamp fields is *ISO. In general, it is recommended that you use the default ISO internal format, especially if you have a mixture of external format types.

For date, time, and timestamp fields, you can change the default format in two ways. You can use the DATFMT and TIMFMT keywords on the Control specification to change the default internal format, if desired, for *all* date-time fields in the program. In addition, you can use the Definition specification to:

- Override the default internal format by using the DATFMT and TIMFMT keywords
- Specify an initial value for a date, time, or timestamp field that is different than the default, by using the INZ keyword

Specifying an External Format for a Date-Time Field: If you have date, time, and timestamp fields in program-described files, then you *must* specify their external format. You can specify a default external format for all date, time, and timestamp fields in a program-described file by using the DATFMT and TIMFMT keywords on a File-Description specification. You can specify an external format for a particular field as well. Specify the desired format in positions 31-34 on an Input specification. Specify the appropriate keyword and format in positions 53-80 on an Output specification.

For more information on each format type, see the appropriate section in the remainder of this chapter.

Date Data Type: Date fields have a predetermined size and format. They can be defined on the definition specification. Leading and trailing zeros are required for all date data.

Date constants or variables used in comparisons or assignments do not have to be in the same format or use the same separators. Also, dates used for I/O operations such as input fields, output fields or key fields are also converted (if required) to the necessary format for the operation.

The default internal format for date variables is *ISO. This default internal format can be overridden globally by the control specification keyword DATFMT and individually by the definition specification keyword DATFMT.

The hierarchy used when determining the internal date format and separator for a date field is

1. From the DATFMT keyword specified on the definition specification
2. From the DATFMT keyword specified on the control specification
3. *ISO

There are two kinds of date data formats: 2-digit and 4-digit-year formats. For 2-digit-year formats, years in the range 1940 to 2039 can be represented. This leads to the possibility of a date overflow condition occurring when converting from a 4-digit-year format to a 2-digit-year format.

The following table lists the formats for date data.

Figure 7-1. Date Formats for Date Data Type

Format name	Description	Format	Length	Example
*MDY	Month/Day/Year	mm/dd/yy	8	01/15/91
*DMY	Day/Month/Year	dd/mm/yy	8	15/01/91
*YMD	Year/Month/Day	yy/mm/dd	8	91/01/15
*JUL	Julian	yy/ddd	6	91/015
*ISO	International Standards Organization	yyyy-mm-dd	10	1991-01-15
*USA	IBM USA Standard	mm/dd/yyyy	10	01/15/1991
*EUR	IBM European Standard	dd.mm.yyyy	10	15.01.1991
*JIS	Japanese Industrial Standard Christian Era	yyyy-mm-dd	10	1991-01-15

The following table lists the *LOVAL, *HIVAL, and default values for all the date formats.

Figure 7-2. Date Values

Format name	Description	*LOVAL	*HIVAL	Default Value
*MDY	Month/Day/Year	01/01/40	12/31/39	01/01/01
*DMY	Day/Month/Year	01/01/40	31/12/39	01/01/01
*YMD	Year/Month/Day	40/01/01	39/12/31	01/01/01
*JUL	Julian	40/001	39/365	01/001
*ISO	International Standards Organization	0001-01-01	9999-12-31	0001-01-01
*USA	IBM USA Standard	01/01/0001	12/31/9999	01/01/0001
*EUR	IBM European Standard	01.01.0001	31.12.9999	01.01.0001
*JIS	Japanese Industrial Standard Christian Era	0001-01-01	9999-12-31	0001-01-01

Time Data Type: Time fields have a predetermined size and format. They can be defined on the definition specification. Leading and trailing zeros are required for all time data.

Time constants or variables used in comparisons or assignments do not have to be in the same format or use the same separators. Also, times used for I/O operations such as input fields, output fields or key fields are also converted (if required) to the necessary format for the operation.

The default internal format for time variables is *ISO. This default internal format can be overridden globally by the control specification keyword TIMFMT and individually by the definition specification keyword TIMFMT.

The hierarchy used when determining the internal time format and separator for a time field is

1. From the TIMFMT keyword specified on the definition specification
2. From the TIMFMT keyword specified on the control specification
3. *ISO

The following table lists the formats for time data.

Figure 7-3. Time formats for Time data type

Format name	Description	Format	Length	Example
*HMS	Hours:Minutes:Seconds	hh:mm:ss	8	14:00:00
*ISO	International Standards Organization	hh.mm.ss	8	14.00.00
*USA	IBM USA Standard. AM and PM can be any mix of upper and lower case.	hh:mm AM or hh:mm PM	8	02:00 PM
*EUR	IBM European Standard	hh.mm.ss	8	14.00.00
*JIS	Japanese Industrial Standard Christian Era	hh:mm:ss	8	14:00:00

The following table lists the *LOVAL, *HIVAL, and default values for all the time formats.

Figure 7-4. Time Values

Format name	Description	*LOVAL	*HIVAL	Default Value
*HMS	Hours:Minutes:Seconds	00:00:00	24:00:00	00:00:00
*ISO	International Standards Organization	00.00.00	24.00.00	00.00.00
*USA	IBM USA Standard. AM and PM can be any mix of upper and lower case.	00:00 AM	12:00 AM	00:00 AM
*EUR	IBM European Standard	00.00.00	24.00.00	00.00.00
*JIS	Japanese Industrial Standard Christian Era	00:00:00	24:00:00	00:00:00

Timestamp Data Type: Timestamp fields have a predetermined size and format. They can be defined on the definition specification. Timestamp data must be in the format

yyyy-mm-dd-hh.mm.ss.mmmmmm (length 26).

Microseconds (.mmmmmm) are optional for timestamp literals and if not provided will be padded on the right with zeros. Leading zeros are required for all timestamp data.

The default initialization value for a timestamp is midnight of January 1, 0001 (0001-01-01-00.00.00.000000). The *HIVAL value for a timestamp is 9999-12-31-24.00.00.000000. Similarly, the *LOVAL value for timestamp is 0001-01-01-00.00.00.000000.

User Date Special Words

The user date special words (UDATE, *DATE, UMONTH, *MONTH, UDAY, *DAY, UYEAR, *YEAR) allow the programmer to supply a date for the program at run time. The user date special words access the job date that is specified in the job description. The user dates can be written out at output time; UDATE and *DATE can be written out using the Y edit code in the format specified by the control specification. (For a description of the job date, see the *Work Management* manual, SC41-4306.)

Rules for User Date: Remember the following rules when using the user date:

- UDATE, when specified in positions 30 through 43 of the output specifications, prints a 6-character numeric date field. *DATE, when similarly specified, prints an 8-character (4-digit year portion) numeric date field. These special words can be used in three different date formats:

  Month/day/year
  Year/month/day
  Day/month/year

  Use the DATEDIT keyword on the control specification to specify the editing to be done. If this keyword is not specified, the default is *MDY.

- For an interactive program, the user date special words are set when the job starts running. For a batch program, they are set when the job is sent to the job queue. In neither case are they updated when the program runs over midnight or when the job date changes. Use the TIME operation code to obtain the time and date while the program is running.

- UMONTH, *MONTH, UDAY, *DAY, and UYEAR when specified in positions 30 through 43 of the output specifications, print a 2-position numeric date field. *YEAR can be used to print a 4-position numeric date field. Use UMONTH or *MONTH to print the month only, UDAY or *DAY to print the day only, and UYEAR or *YEAR to print the year only.

- UDATE and *DATE can be edited when they are written if the Y edit code is specified in position 44 of the output specifications. The DATEDIT(fmt{separator}) keyword on the control specification determines the format and the separator character to be inserted; for example, 12/31/88, 31.12.88., 12/31/1988.

- UMONTH, *MONTH, UDAY, *DAY, UYEAR and *YEAR cannot be edited by the Y edit code in position 44 of the output specifications.

- The user date fields cannot be modified. This means they cannot be used:

  – In the result field of calculations
  – As factor 1 of PARM operations
  – As the factor 2 index of LOOKUP operations
  – With blank after in output specifications
  – As input fields

- The user date special words can be used in factor 1 or factor 2 of the calculation specifications for operation codes that use numeric fields.

- User date fields are not date data type fields but are numeric fields.

Editing Date Fields

The Y edit code is normally used to edit a 3- to 9-digit date field. It suppresses the leftmost zeros of date fields, up to but not including the digit preceding the first separator. Slashes are inserted to separate the day, month, and year. The DATEDIT(fmt{separator})and (′value′) keywords on the control specification can be used to alter edit formats.

Note: The Y edit code is not valid for *YEAR, *MONTH, and *DAY.

The Z edit code removes the sign (plus or minus) from and suppresses the leading zeros of a numeric field. The decimal point is not placed in the field and is not printed.

The Y edit code suppresses the leftmost zeros of date fields, up to but not including the digit preceding the first separator. The Y edit code also inserts slashes (/) between the month, day, and year according to the following pattern:

nn/n	
nn/nn	
nn/nn/n	
nn/nn/nn	
nnn/nn/nn	
nn/nn/nnnn	Format used with M, D or blank in position 19
nnn/nn/nnnn	Format used with M, D or blank in position 19
nnnn/nn/nn	Format used with Y in position 19
nnnnn/nn/nn	Format used with Y in position 19

Date Operations

Date operations allow you to perform date and time arithmetic, extract portions of a date, time or timestamp field; or test for valid fields. They operate on date, time, and timestamp fields, and character and numeric fields representing dates, times and timestamps. The date operations are:

- ADDDUR (Add Duration)
- EXTRCT (Extract Date/Time/Timestamp)
- SUBDUR (Subtract Duration)
- TEST (Test Date/Time/Timestamp)

With ADDDUR (Add Duration) you can add a duration to a date or time. With SUBDUR (Subtract Duration) you can subtract a duration from a date or time, or calculate the duration between 2 dates, times or timestamps. With EXTRCT (Extract Date/Time/Timestamp) you can extract part of a date, time or timestamp. With TEST (Test Date/Time/Timestamp) you can test for a valid date, time, or timestamp field. The valid duration codes (and their short forms) are:

- *YEARS for the year (*Y)
- *MONTHS for the month (*M)
- *DAYS for the day (*D)
- *HOURS for the hours (*H)
- *MINUTES for the minutes (*MN)
- *SECONDS for the seconds (*S)
- *MSECONDS for the microseconds (*MS).

Adding or Subtracting Dates: When adding (or subtracting) a duration in months to (or from) a date, the general rule is that the month portion is increased (or decreased) by the number of months in the duration, and the day portion is unchanged. The exception to this is when the resulting day portion would exceed the actual number of days in the resulting month. In this case, the resulting day portion is adjusted to the actual month end date.

For example, adding one month to '95/05/30' (*YMD format) results in '95/06/30', as expected. The resulting month portion has been increased by 1; the day portion is unchanged. On the other hand, adding one month to '95/05/31' results in '95/06/30'. The resulting month portion has been increased by 1 and the resulting day portion has been adjusted because June has only 30 days.

Subtracting one month from '95/03/30' yields '95/02/28'. In this case, the resulting month portion is decreased by 1 and the resulting day portion adjusted because February has only 28 days (in non-leap years).

Similar results occur when adding or subtracting a year duration. For example, adding one year to '92/02/29' results in '93/02/28', an adjusted value since the resulting year is not a leap year.

Calculating Durations between Dates: The SUBDUR operation can be used to calculate a duration by subtracting two dates, times, or timestamps. The result of the calculation is a complete units; any rounding which is done is downwards. The calculation of durations includes microseconds.

For example, if the actual duration is 384 days, and the result is requested in years, the result will be 1 complete year because there are 1.05 years in 384 days. A duration of 59 minutes requested in hours will result in 0 hours. Here are some additional examples.

Duration in	between	and	is
Months	1994-02-28	1994-03-28	1 month
	1994-03-15	1995-03-14	11 months
	1994-03-15	1995-03-15	12 months
Years	1994-03-15	1995-03-14	0 years
	1994-03-31	1995-03-31	1 year
	1970-03-14-23.00.00.000000	1970-03-14-22.00.00.000001	0 years
Hours	1990-03-14-12.34.45.123456	1989-03-14-12.34.45.123457	0 years

Unexpected Results: If adjustment takes place on a date-time addition or subtraction, then a subsequent duration calculation will most likely result in a different duration than the one originally added or subtracted. This is because the calculated duration will no longer contain a complete unit, and so, rounding down, will yield a one unit less than expected. This is shown in examples 1 and 2 below.

A second unexpected result can be seen in examples 3 and 4. Different initial dates give the same result after adding 1 month. When subtracting 1 month from the result, it is impossible to arrive at both initial dates.

1. ′95/05/31′ ADDDUR 1:*MONTH gives ′95/06/30′
   ′95/06/30′ SUBDUR ′95/05/31′ gives 0 months

   You might expect the result of the SUBDUR to be 1 month.

2. ′95/06/30′ ADDDUR 1:*MONTH gives ′95/07/30′
   ′95/07/30′ SUBDUR ′95/06/30′ gives 1 month

   This is the ″expected″ result.

3. ′95/01/31′ ADDDUR 1:*MONTH gives ′95/02/28′
   ′95/01/28′ ADDDUR 1:*MONTH gives ′95/02/28′

   Two different dates yield the same date due to adjustment.

   ′95/02/28′ SUBDUR 1:*MONTH gives ′95/01/28′

   ′Reversing′ the addition does not result in the original dates.

Using OPM RPG/400 Date Support

The OPM RPG 400 compiler has the following support for dates:

- Retrieval of the job date via special words UDATE and UYEAR. This level of support provides for a 2-digit year.
- Retrieval of the job date via special words *DATE and *YEAR. This level of support provides for a 4-digit year.
- Retrieval of the system date via operation TIME. This level of support provides for both 2-digit and 4-digit years.
- Calling OPM System APIs. This level of support provides for both 2-digit and 4-digit years.
- SQL date, time, and timestamp data type support. This level of support provides for 4-digit years.

Using System/36 Compatible Date Support

The System/36 compatible compiler has the following support for dates:

- Retrieval of the job date via special words UDATE and UYEAR. This level of support provides for a 2-digit year.
- Retrieval of the system date via operation TIME. This level of support provides for a 2-digit year.
- Calling OPM System APIs. This level of support provides for both 2-digit and 4-digit years.

The IBM COBOL Family for AS/400

For application code written in COBOL, IBM has developed several COBOL language compilers. These compilers target the host application development environment.

IBM′s COBOL compilers for AS/400 are:

- Integrated Language Environment (ILE) COBOL/400
- Original Program Model (OPM) COBOL/400
- System/36 compatible COBOL compiler

All three COBOL language compilers are available with either the Integrated Language Environment COBOL for OS/400 Version 3 product (5716-CB1) or the IBM Integrated Language Environment COBOL/400 Version 3 product (5763-CB1).

Using COBOL Date Support

The COBOL ACCEPT statement is supported by the three compilers listed above. This level of support provides for a 2-digit year when referencing DATE or DAY.

Calling OPM System APIs is supported by the three compilers listed above. This level of support provides for both 2-digit and 4-digit years.

The COBOL WHEN-COMPILED special register is supported in both the OPM and ILE COBOL/400 compilers. This level of support provides for a 2-digit year.

Calling ILE Date APIs is supported by the ILE COBOL/400 compiler. This level of support provides for both 2-digit and 4-digit years.

SQL is supported by both the OPM and ILE COBOL compilers. This level of support provides for 4-digit years.

The IBM C Family for AS/400

For application code based on C, IBM has developed language compilers for both C and C++.

IBM′s C based compilers for AS/400 are:

- Integrated Language Environment (ILE) C for OS/400
- VisualAge C++ for OS/400

Using C and C++ Date Support

The C library, available to both compilers, provides many functions related to date and time retrieval, manipulation, and formatting. This level of support provides for 4-digit years.

Calling OPM System APIs is supported by both compilers. This level of support provides for both 2-digit and 4-digit years.

Calling ILE Date APIs is supported by both compilers. This level of support provides for both 2-digit and 4-digit years.

SQL is supported by the ILE C for OS/400 compiler. This level of support provides for 4-digit years.

Integrated Language Environment for OS/400

Integrated Language Environment (ILE) for OS/400 is IBM′s common runtime environment for enterprise applications written in the ILE languages of RPG, COBOL, C, and CL. ILE is designed to provide defined calling conventions, enhanced interlanguage communication, and callable services in areas such as date and time, math, storage management, and exception handling. ILE services are standard in OS/400 starting with Version 2 Release 3.

ILE provides a number of date functions which include:

- Ability to parse dates in an infinite number of formats by using a picture string as a parsing guide
- Retrieve current date
- Convert a date character string to a Lilian (integer) format thereby enabling easy date arithmetic operations
- Convert a Lilian date to a date character string
- Century sliding window

The ILE century sliding window technique may provide a short term solution for some applications. If you are unable to change all of your applications and data at the same time, the century window allows 2-digit years to be interpreted in a 100-year window. Any 100-year window can be selected. You pass a 2-digit year to ILE and ILE returns a 4-digit year based on the 100-year window.

The advantage to the century window is that you need to change only the application code and not the databases with 2-digit years. This allows you to change the application programs one at a time or groups at a time without affecting your databases. Note, this only works for dates that range less than 100 years. For example, dates of birth may not be appropriate for this solution.

DB2/400 SQL

DB2/400, a standard part of Operating System/400, provides the run time support for SQL on the AS/400. SQL provides support for the database datatypes of date, time, and timestamp; and operations such as add, subtract, assignment, and compare. The DB2 Query Manager and SQL Development Kit product (5716-ST1 or 5763-ST1) provides SQL precompilers for AS/400 programming languages such as RPG, C, and COBOL. SQL support provides for 4-digit year support.

Using Date, Time, and Timestamp Support

Date, time, and timestamp are data types represented in an internal form not seen by the SQL user. Date, time, and timestamp can be represented by character string values and assigned to character string variables. The database manager recognizes the following as date, time, and timestamp values:

- A value returned by the DATE, TIME, or TIMESTAMP scalar functions.
- A value returned by the CURRENT DATE, CURRENT TIME, or CURRENT TIMESTAMP special registers.
- A character string when it is an operand of an arithmetic expression or a comparison *and* the other operand is a date, time, or timestamp. For example, in the predicate:

  ```
  ... WHERE HIREDATE < '1950-01-01'
  ```

if HIREDATE is a date column, the character string '1950-01-01' is interpreted as a date.

- A character string variable or constant used to set a date, time, or timestamp column in either the SET clause of an UPDATE statement, or the VALUES clause of an INSERT statement.

For more information on character string formats of date, time, and timestamp values, refer to *DB2/400 SQL Reference* (SC41-3612).

Specifying Current Date and Time Values: You can specify a current date, time, or timestamp in an expression by specifying one of three special registers: CURRENT DATE, CURRENT TIME, or CURRENT TIMESTAMP. The value of each is based on a time-of-day clock reading obtained during the running of the statement. Multiple references to CURRENT DATE, CURRENT TIME, or CURRENT TIMESTAMP within the same SQL statement use the same value. The following statement returns the age (in years) of each employee in the EMPLOYEE table when the statement is run:

```
SELECT YEAR(CURRENT DATE - BIRTHDATE)
FROM CORPDATA.EMPLOYEE
```

The CURRENT TIMEZONE special register allows a local time to be converted to Universal Coordinated Time (UTC). For example, if you have a table named DATETIME, containing a time column type with a name of STARTT, and you want to convert STARTT to UTC, you can use the following statement:

```
SELECT STARTT - CURRENT TIMEZONE
FROM DATETIME
```

Datetime Operands and Durations: Datetime values can be incremented, decremented, and subtracted. These operations may involve decimal numbers called *durations*. A *duration* is a positive or negative number representing an interval of time. There are four types of durations:

Labeled Durations

A *labeled duration* represents a specific unit of time as expressed by a number (which can be the result of an expression) followed by one of the seven duration keywords: YEARS, MONTHS, DAYS, HOURS, MINUTES, SECONDS, or MICROSECONDS.[1] The number specified is converted as if it were assigned to a DECIMAL(15,0) number. A labeled duration can only be used as an operand of an arithmetic operator in which the other operand is a value of data type DATE, TIME, or TIMESTAMP. Thus, the expression HIREDATE + 2 MONTHS + 14 DAYS is valid whereas the expression HIREDATE + (2 MONTHS + 14 DAYS) is not. In both of these expressions, the labeled durations are 2 MONTHS and 14 DAYS.

Date Duration

A *date duration* represents a number of years, months, and days, expressed as a DECIMAL(8,0) number. To be properly interpreted, the number must have the format *yyyymmdd*, where *yyyy* represents the number of years, *mm* the number of months, and *dd* the number of days. The result of subtracting one date value from another, as in the expression HIREDATE - BRTHDATE, is a date duration.

---

[1] Note that the singular form of these keywords is also acceptable: YEAR, MONTH, DAY, HOUR, MINUTE, SECOND, and MICROSECOND.

Time Duration

A *time duration* represents a number of hours, minutes, and seconds, expressed as a DECIMAL(6,0) number. To be properly interpreted, the number must have the format *hhmmss* where *hh* represents the number of hours, *mm* the number of minutes, and *ss* the number of seconds. The result of subtracting one time value from another is a time duration.

Timestamp duration

A *timestamp duration* represents a number of years, months, days, hours, minutes, seconds, and microseconds, expressed as a DECIMAL(20,6) number. To be properly interpreted, the number must have the format *yyyymmddhhmmsszzzzzz,* where *yyyy, mm, dd, hh, mm, ss,* and *zzzzzz* represent, respectively, the number of years, months, days, hours, minutes, seconds, and microseconds. The result of subtracting one timestamp value from another is a timestamp duration.

Datetime Arithmetic in SQL: The only arithmetic operations that can be performed on datetime values are addition and subtraction. If a datetime value is the operand of addition, the other operand must be a duration. The specific rules governing the use of the addition operator with datetime values follow:

- If one operand is a date, the other operand must be a date duration or labeled duration of years, months, or days.
- If one operand is a time, the other operand must be a time duration or a labeled duration of hours, minutes, or seconds.
- If one operand is a timestamp, the other operand must be a duration. Any type of duration is valid.
- Neither operand of the addition operator can be a parameter marker.

The rules for the use of the subtraction operator on datetime values are not the same as those for addition because a datetime value cannot be subtracted from a duration, and because the operation of subtracting two datetime values is not the same as the operation of subtracting a duration from a datetime value. The specific rules governing the use of the subtraction operator with datetime values follow:

- If the first operand is a date, the second operand must be a date, a date duration, a string representation of a date, or a labeled duration of years, months, or days.
- If the second operand is a date, the first operand must be a date, or a string representation of a date.
- If the first operand is a time, the second operand must be a time, a time duration, a string representation of a time, or a labeled duration of hours, minutes, or seconds.
- If the second operand is a time, the first operand must be a time, or string representation of a time.
- If the first operand is a timestamp, the second operand must be a timestamp, a string representation of a timestamp, or a duration.
- If the second operand is a timestamp, the first operand must be a timestamp or a string representation of a timestamp.
- Neither operand of the subtraction operator can be a parameter marker.

Date Arithmetic: Dates can be subtracted, incremented, or decremented.

Subtracting Dates: The result of subtracting one date (DATE2) from another (DATE1) is a date duration that specifies the number of years, months, and days between the two dates. The data type of the result is DECIMAL(8,0). If DATE1 is greater than or equal to DATE2, DATE2 is subtracted from DATE1. If DATE1 is less than DATE2, however, DATE1 is subtracted from DATE2, and the sign of the result is made negative. The following procedural description clarifies the steps involved in the operation RESULT = DATE1 - DATE2.

```
If DAY(DATE2) <= DAY(DATE1)
    then DAY(RESULT) = DAY(DATE1) - DAY(DATE2).

If DAY(DATE2) > DAY(DATE1)
    then DAY(RESULT) = N + DAY(DATE1) - DAY(DATE2)
      where N = the last day of MONTH(DATE2).
    MONTH(DATE2) is then incremented by 1.

If MONTH(DATE2) <= MONTH(DATE1)
    then MONTH(RESULT) = MONTH(DATE1) - MONTH(DATE2).

If MONTH(DATE2) > MONTH(DATE1)
    then MONTH(RESULT) = 12 + MONTH(DATE1) - MONTH(DATE2).
    YEAR(DATE2) is then incremented by 1.

YEAR(RESULT) = YEAR(DATE1) - YEAR(DATE2).
```

For example, the result of DATE('3/15/2000') - '12/31/1999' is 215 (or, a duration of 0 years, 2 months, and 15 days).

Incrementing and Decrementing Dates: The result of adding a duration to a date, or of subtracting a duration from a date, is itself a date. (For the purposes of this operation, a month denotes the equivalent of a calendar page. Adding months to a date, then, is like turning the pages of a calendar, starting with the page on which the date appears.) The result must fall between the dates January 1, 0001 and December 31, 9999 inclusive. If a duration of years is added or subtracted, only the year portion of the date is affected. The month is unchanged, as is the day unless the result would be February 29 of a non-leap-year. In this case, the day is changed to 28, and SQLWARN6 in the SQLCA is set to 'W' to indicate the end-of-month adjustment.

Similarly, if a duration of months is added or subtracted, only months and, if necessary, years are affected. The day portion of the date is unchanged unless the result would be invalid (September 31, for example). In this case, the day is set to the last day of the month, and SQLWARN6 in the SQLCA is set to 'W' to indicate the end-of-month adjustment.

Adding or subtracting a duration of days will, of course, affect the day portion of the date, and potentially the month and year. Adding a labeled duration of DAYS will not cause an end-of-month adjustment.

Date durations, whether positive or negative, may also be added to and subtracted from dates. As with labeled durations, the result is a valid date, and a warning indicator is set in the SQLCA whenever an end-of-month adjustment is necessary.

When a positive date duration is added to a date, or a negative date duration is subtracted from a date, the date is incremented by the specified number of years, months, and days, in that order. Thus DATE1 + X, where X is a positive DECIMAL(8,0) number, is equivalent to the expression:

DATE1 + YEAR(X) YEARS + MONTH(X) MONTHS + DAY(X) DAYS

When a positive date duration is subtracted from a date, or a negative date duration is added to a date, the date is decremented by the specified number of days, months, and years, in that order. Thus, DATE1 - X, where X is a positive DECIMAL(8,0) number, is equivalent to the expression:

DATE1 - DAY(X) DAYS - MONTH(X) MONTHS - YEAR(X) YEARS

When adding durations to dates, adding one month to a given date gives the same date one month later *unless* that date does not exist in the later month. In that case, the date is set to that of the last day of the later month. For example, January 28 plus one month gives February 28; and one month added to January 29, 30, or 31 results in either February 28 or, for a leap year, February 29.

Note: If one or more months is added to a given date and then the same number of months is subtracted from the result, the final date is not necessarily the same as the original date.

Time Arithmetic: Times can be subtracted, incremented, or decremented.

Subtracting Times: The result of subtracting one time (TIME2) from another (TIME1) is a time duration that specifies the number of hours, minutes, and seconds between the two times. The data type of the result is DECIMAL(6,0). If TIME1 is greater than or equal to TIME2, TIME2 is subtracted from TIME1. If TIME1 is less than TIME2, however, TIME1 is subtracted from TIME2, and the sign of the result is made negative. The following procedural description clarifies the steps involved in the operation RESULT = TIME1 - TIME2.

If SECOND(TIME2) <= SECOND(TIME1)
    then SECOND(RESULT) = SECOND(TIME1) - SECOND(TIME2).

If SECOND(TIME2) > SECOND(TIME1)
    then SECOND(RESULT) = 60 + SECOND(TIME1) - SECOND(TIME2).
    MINUTE(TIME2) is then incremented by 1.

If MINUTE(TIME2) <= MINUTE(TIME1)
    then MINUTE(RESULT) = MINUTE(TIME1) - MINUTE(TIME2).

If MINUTE(TIME2) > MINUTE(TIME1)
    then MINUTE(RESULT) = 60 + MINUTE(TIME1) - MINUTE(TIME2).
    HOUR(TIME2) is then incremented by 1.

HOUR(RESULT) = HOUR(TIME1) - HOUR(TIME2).

For example, the result of TIME(′11:02:26′) - ′00:32:56′ is 102930 (a duration of 10 hours, 29 minutes, and 30 seconds).

Incrementing and Decrementing Times: The result of adding a duration to a time, or of subtracting a duration from a time, is itself a time. Any overflow or underflow of hours is discarded, thereby ensuring that the result is always a time. If a duration of hours is added or subtracted, only the hours portion of the time is affected. The minutes and seconds are unchanged.

Similarly, if a duration of minutes is added or subtracted, only minutes and, if necessary, hours are affected. The seconds portion of the time is unchanged.

Adding or subtracting a duration of seconds will, of course, affect the seconds portion of the time, and potentially the minutes and hours.

Time durations, whether positive or negative, also can be added to and subtracted from times. The result is a time that has been incremented or decremented by the specified number of hours, minutes, and seconds, in that order. TIME1 + X, where "X" is a DECIMAL(6,0) number, is equivalent to the expression:

TIME1 + HOUR(X) HOURS + MINUTE(X) MINUTES + SECOND(X) SECONDS

Timestamp Arithmetic: Timestamps can be subtracted, incremented, or decremented.

Subtracting Timestamps: The result of subtracting one timestamp (TS2) from another (TS1) is a timestamp duration that specifies the number of years, months, days, hours, minutes, seconds, and microseconds between the two timestamps. The data type of the result is DECIMAL(20,6). If TS1 is greater than or equal to TS2, TS2 is subtracted from TS1. If TS1 is less than TS2, however, TS1 is subtracted from TS2 and the sign of the result is made negative. The following procedural description clarifies the steps involved in the operation RESULT = TS1 - TS2.

If MICROSECOND(TS2) <= MICROSECOND(TS1)
then MICROSECOND(RESULT) = MICROSECOND(TS1) - MICROSECOND(TS2).

If MICROSECOND(TS2) > MICROSECOND(TS1)
then MICROSECOND(RESULT) = 1000000 + MICROSECOND(TS1) - MICROSECOND(TS2)
and SECOND(TS2) is incremented by 1.

The seconds and minutes part of the timestamps are subtracted as specified in the rules for subtracting times.

If HOUR(TS2) <= HOUR(TS1)
then HOUR(RESULT) = HOUR(TS1) - HOUR(TS2).

If HOUR(TS2) > HOUR(TS1)
then HOUR(RESULT) = 24 + HOUR(TS1) - HOUR(TS2)
and DAY(TS2) is incremented by 1.

The date part of the timestamps is subtracted as specified in the rules for subtracting dates.

Incrementing and Decrementing Timestamps: The result of adding a duration to a timestamp, or of subtracting a duration from a timestamp, is itself a timestamp. Date and time arithmetic is performed as previously defined, except that an overflow or underflow of hours is carried into the date part of the result, which must be within the range of valid dates. Microseconds overflow into seconds.

OS/400 CL

CVTDAT (Convert Date)

The Convert Date (CVTDAT) command converts a date value from one format to another, without changing its value. CVTDAT provides for both 2-digit and 4-digit-year formats.

OPNQRYF (Open Query File)

The Open Query File (OPNQRYF) command opens a file that contains a set of database records that satisfies a database query request. OPNQRYF provides for both 2-digit and 4-digit years.

Date, Time, and Timestamp Comparisons Using the OPNQRYF Command: A date, time, or timestamp value can be compared either with another value of the same data type or with a string representation of that data type. All comparisons are chronological, which means the farther a time is from January 1, 0001, the *greater* the value of that time.

Comparisons involving time values and string representations of time values always include seconds. If the string representation omits seconds, zero seconds are implied.

Comparisons involving timestamp values are chronological without regard to representations that might be considered equivalent. Thus, the following predicate is true:

```
TIMESTAMP('1990-02-23-00.00.00') > '1990-02-22-24.00.00'
```

When a character, DBCS-open, or DBCS-either field or constant is represented as a date, time, or timestamp, the following rules apply:

Date: The length of the field or literal must be at least 8 if the date format is *ISO, *USA, *EUR, *JIS, *YMD, *MDY, or *DMY. If the date format is *JUL (yyddd), the length of the variable must be at least 6 (includes the separator between yy and ddd). The field or literal may be padded with blanks.

Time: For all of the time formats (*USA, *ISO, *EUR, *JIS, *HMS), the length of the field or literal must be at least 4. The field or literal may be padded with blanks.

Timestamp: For the timestamp format (yyyy-mm-dd-hh.mm.ss.uuuuuu), the length of the field or literal must be at least 16. The field or literal may be padded with blanks.

Date, Time, and Timestamp Arithmetic Using OPNQRYF CL Command: Date, time, and timestamp values can be incremented, decremented, and subtracted. These operations may involve decimal numbers called *durations*. Following is a definition of durations and a specification of the rules for performing arithmetic operations on date, time, and timestamp values.

Durations: A **duration** is a number representing an interval of time. The four types of durations are:

Labeled Duration

A **labeled duration** represents a specific unit of time as expressed by a number (which can be the result of an expression) used as an operand for one of the seven duration built-in functions: %DURYEAR, %DURMONTH, %DURDAY, %DURHOUR, %DURMINUTE, %DURSEC, or %DURMICSEC. The functions are for the duration of year, month, day, hour, minute, second, and microsecond, respectively. The number specified is converted as if it was assigned to a DECIMAL(15,0) number. A labeled duration can only be used as an operand of an arithmetic operator when the other operand is a value of data type *DATE, *TIME, or *TIMESTP. Thus, the expression HIREDATE + %DURMONTH(2) + %DURDAY(14) is valid, whereas the expression HIREDATE + (%DURMONTH(2) + %DURDAY(14)) is not. In both of these expressions, the labeled durations are %DURMONTH(2) and %DURDAY(14).

Date Duration

A **date duration** represents a number of years, months, and days, expressed as a DECIMAL(8,0) number. To be properly interpreted, the number must have the format *yyyymmdd*, where *yyyy* represents the number of years, *mm* the number of months, and *dd* the number of days. The result of subtracting one date value from another, as in the expression HIREDATE - BRTHDATE, is a date duration.

Time Duration

A **time duration** represents a number of hours, minutes, and seconds, expressed as a DECIMAL(6,0) number. To be properly interpreted, the number must have the format *hhmmss*, where *hh* represents the number of hours, *mm* the number of minutes, and *ss* the number of seconds. The result of subtracting one time value from another is a time duration.

Timestamp Duration

A **timestamp duration** represents a number of years, months, days, hours, minutes, seconds, and microseconds, expressed as a DECIMAL(20,6) number. To be properly interpreted, the number must have the format *yyyymmddhhmmsszzzzzz,* where *yyyy, mm, dd, hh, mm, ss,* and *zzzzzz* represent, respectively, the number of years, months, days, hours, minutes, seconds, and microseconds. The result of subtracting one timestamp value from another is a timestamp duration.

Rules for Date, Time, and Timestamp Arithmetic: The only arithmetic operations that can be performed on date and time values are addition and subtraction. If a date or time value is the operand of addition, the other operand must be a duration. The specific rules governing the use of the addition operator with date and time values follow:

- If one operand is a date, the other operand must be a date duration or a labeled duration of years, months, or days.
- If one operand is a time, the other operand must be a time duration or a labeled duration of hours, minutes, or seconds.

- If one operand is a timestamp, the other operand must be a duration. Any type of duration is valid.

The rules for the use of the subtraction operator on date and time values are not the same as those for addition because a date or time value cannot be subtracted from a duration, and because the operation of subtracting two date and time values is not the same as the operation of subtracting a duration from a date or time value. The specific rules governing the use of the subtraction operator with date and time values follow:

- If the first operand is a date, the second operand must be a date, a date duration, a string representation of a date, or a labeled duration of years, months, or days.
- If the second operand is a date, the first operand must be a date or a string representation of a date.
- If the first operand is a time, the second operand must be a time, a time duration, a string representation of a time, or a labeled duration of hours, minutes, or seconds.
- If the second operand is a time, the first operand must be a time or string representation of a time.
- If the first operand is a timestamp, the second operand must be a timestamp, a string representation of a timestamp, or a duration.
- If the second operand is a timestamp, the first operand must be a timestamp or a string representation of a timestamp.

Date Arithmetic: Dates can be subtracted, incremented, or decremented.

Subtracting Dates: The result of subtracting one date (DATE2) from another (DATE1) is a date duration that specifies the number of years, months, and days between the two dates. The data type of the result is DECIMAL(8,0). If DATE1 is greater than or equal to DATE2, DATE2 is subtracted from DATE1. If DATE1 is less than DATE2, however, DATE1 is subtracted from DATE2, and the sign of the result is made negative. The following procedural description clarifies the steps involved in the operation RESULT = DATE1 - DATE2.

```
If %DAY(DATE2) <= %DAY(DATE1)
     then %DAY(RESULT) = %DAY(DATE1) - %DAY(DATE2).

If %DAY(DATE2) > %DAY(DATE1)
    then %DAY(RESULT) = N + %DAY(DATE1) - %DAY(DATE2)
     where N = the last day of %MONTH(DATE2).
    %MONTH(DATE2) is then incremented by 1.

If %MONTH(DATE2) <= %MONTH(DATE1)
    then %MONTH(RESULT) = %MONTH(DATE1) - %MONTH(DATE2).

If %MONTH(DATE2) > %MONTH(DATE1)
    then %MONTH(RESULT) = 12 + %MONTH(DATE1) - %MONTH(DATE2).
    %YEAR(DATE2) is then incremented by 1.

%YEAR(RESULT) = %YEAR(DATE1) - %YEAR(DATE2).
```

For example, the result of %DATE('3/15/2000') - '12/31/1999' is 215 (or, a duration of 0 years, 2 months, and 15 days).

Incrementing and Decrementing Dates: The result of adding a duration to a date, or of subtracting a duration from a date, is itself a date. (For the purposes of this operation, a month denotes the equivalent of a calendar page. Adding months to a date, then, is like turning the pages of a calendar, starting with the page on which the date appears.) The result must fall between the dates January 1, 0001, and December 31, 9999, inclusive. If a duration of years is added or subtracted, only the year portion of the date is affected. The month is unchanged, as is the day unless the result would be February 29 of a year that is not a leap year. In this case, the day is changed to 28.

Similarly, if a duration of months is added or subtracted, only months and, if necessary, years are affected. The day portion of the date is unchanged unless the result would not be valid (September 31, for example). In this case, the day is set to the last day of the month.

Adding or subtracting a duration of days will, of course, affect the day portion of the date, and potentially the month and year.

Date durations, whether positive or negative, may also be added to and subtracted from dates. As with labeled durations, the result is a valid date.

When a positive date duration is added to a date, or a negative date duration is subtracted from a date, the date is incremented by the specified number of years, months, and days, in that order. Thus, DATE1 + X, where X is a positive DECIMAL(8,0) number, is equivalent to the expression: DATE1 + %DURYEAR(%YEAR(X)) + %DURMONTH(%MONTH(X)) + %DURDAY(%DAY(X))

When a positive date duration is subtracted from a date, or a negative date duration is added to a date, the date is decremented by the specified number of days, months, and years, in that order. Thus, DATE1 - X, where X is a positive DECIMAL(8,0) number, is equivalent to the expression: DATE1 - %DURDAY(%DAY(X)) - %DURMONTH(%MONTH(X)) - %DURYEAR(%YEAR(X))

When adding durations to dates, adding one month to a given date gives the same date one month later *unless* that date does not exist in the later month. In that case, the date is set to that of the last day of the later month. For example, January 28 plus one month gives February 28; and one month added to January 29, 30, or 31 results in either February 28 or, for a leap year, February 29.

Note: If one or more months are added to a given date and then the same number of months is subtracted from the result, the final date is not necessarily the same as the original date.

Time Arithmetic: Times can be subtracted, incremented, or decremented.

Subtracting Times: The result of subtracting one time (TIME2) from another (TIME1) is a time duration that specifies the number of hours, minutes, and seconds between the two times. The data type of the result is DECIMAL(6,0). If TIME1 is greater than or equal to TIME2, TIME2 is subtracted from TIME1. If TIME1 is less than TIME2, however, TIME1 is subtracted from TIME2, and the sign of the result is made negative. The following procedural description clarifies the steps involved in the operation RESULT = TIME1 - TIME2.

If %SECOND(TIME2) <= %SECOND(TIME1)
then %SECOND(RESULT) = %SECOND(TIME1) - %SECOND(TIME2).

If %SECOND(TIME2) > %SECOND(TIME1)
then %SECOND(RESULT) = 60 + %SECOND(TIME1) - %SECOND(TIME2).
%MINUTE(TIME2) is then incremented by 1.

If %MINUTE(TIME2) <= %MINUTE(TIME1)
then %MINUTE(RESULT) = %MINUTE(TIME1) - %MINUTE(TIME2).

If %MINUTE(TIME2) > %MINUTE(TIME1)
then %MINUTE(RESULT) = 60 + %MINUTE(TIME1) - %MINUTE(TIME2).
%HOUR(TIME2) is then incremented by 1.

%HOUR(RESULT) = %HOUR(TIME1) - %HOUR(TIME2).

For example, the result of %TIME('11:02:26') - '00:32:56' is 102930 (a duration of 10 hours, 29 minutes, and 30 seconds).

Incrementing and Decrementing Times: The result of adding a duration to a time, or of subtracting a duration from a time, is itself a time. Any overflow or underflow of hours is discarded, thereby ensuring that the result is always a time. If a duration of hours is added or subtracted, only the hours portion of the time is affected. The minutes and seconds are unchanged.

Similarly, if a duration of minutes is added or subtracted, only minutes and, if necessary, hours are affected. The seconds portion of the time is unchanged.

Adding or subtracting a duration of seconds will, of course, affect the seconds portion of the time, and potentially the minutes and hours.

Time durations, whether positive or negative, also can be added to and subtracted from times. The result is a time that has been incremented or decremented by the specified number of hours, minutes, and seconds, in that order. TIME1 + X, where X is a DECIMAL(6,0) number, is equivalent to the expression: TIME1 + %DURHOUR(%HOUR(X)) + %DURMINUTE(%MINUTE(X)) + %DURSEC(%SECOND(X))

Timestamp Arithmetic: Timestamps can be subtracted, incremented, or decremented.

Subtracting Timestamps: The result of subtracting one timestamp (TS2) from another (TS1) is a timestamp duration that specifies the number of years, months, days, hours, minutes, seconds, and microseconds between the two timestamps. The data type of the result is DECIMAL(20,6). If TS1 is greater than or equal to TS2, TS2 is subtracted from TS1. If TS1 is less than TS2, however, TS1 is subtracted from TS2 and the sign of the result is made negative. The following procedural description clarifies the steps involved in the operation RESULT = TS1 - TS2:

If %MICSEC(TS2) <= %MICSEC(TS1)
then %MICSEC(RESULT) = %MICSEC(TS1) -
%MICSEC(TS2).

If %MICSEC(TS2) > %MICSEC(TS1)
then %MICSEC(RESULT) = 1000000 +
%MICSEC(TS1) - %MICSEC(TS2)
and %SECOND(TS2) is incremented by 1.

The seconds and minutes part of the timestamps are subtracted as specified in the rules for subtracting times:

If %HOUR(TS2) <= %HOUR(TS1)
then %HOUR(RESULT) = %HOUR(TS1) - %HOUR(TS2).

If %HOUR(TS2) > %HOUR(TS1)
then %HOUR(RESULT) = 24 + %HOUR(TS1) - %HOUR(TS2)
and %DAY(TS2) is incremented by 1.

The date part of the timestamp is subtracted as specified in the rules for subtracting dates.

Incrementing and Decrementing Timestamps: The result of adding a duration to a timestamp, or of subtracting a duration from a timestamp, is itself a timestamp. Date and time arithmetic is performed as previously defined, except that an overflow or underflow of hours is carried into the date part of the result, which must be within the range of valid dates. Microseconds overflow into seconds.

Application Dictionary Services/400

IBM Application Dictionary Services/400 is a host-based, integrated impact analysis tool to improve programmer productivity and application quality during application development and maintenance.

The product is a feature of the Application Development ToolSet/400 and is integrated with such tools as Source Entry Utility (SEU), Screen Design Aid (SDA), and Data File Utility (DFU). It is fully menu-driven, very simple to learn and has virtually no learning curve for experienced AS/400 programmers.

Application Dictionary Services provides the ability to place a "dictionary" or a cross-referencing index over data and programs. When a programmer wants to make a change to an application program, he or she uses the dictionary to determine the effect that change will have on other programs, files, and data. The programmer is saved from manually performing the tedious and error-prone work of looking for these relationships. This, coupled with the Application Dictionary Services mass compile feature, can improve the quality of applications and decrease the amount of time programmers spend maintaining existing code.

With the pervasiveness of date information in many applications, this impact analysis and mass compile capability can simplify your transition to a Year2000-ready application base.

Application Development Manager/400

IBM Application Development Manager/400 is a host-based change management tool that provides application developers a development environment to effectively and efficiently manage application development and maintenance.

The product is integrated with the AS/400 Product Development Manager (PDM) and is a feature of Application Development ToolSet/400. It can be used either from the PDM menu, or by typing the Control Language (CL) commands on the command line.

Application Development Manager provides a discipline to better control the development and maintenance environment. It forces the project leader to define the application environment, outlining the different stages of application (production, test, or fix) and the roles of each developer. The developers work in a well-organized environment, which in turn, leads to increased productivity. Through its version control facility, it allows the creating and managing of several versions of the same applications with less disk utilization and the audit trail feature keeps a log of all changes made to an application, allowing better monitoring of the application.

With the pervasiveness of date information in many applications, this change management tool can simplify your transition to a Year2000 safe application base.

IBM Tools for Personal Computers

The IBM COBOL Family for the Workstations

For application code written in COBOL for the workstation or developed for the MVS, VM, VSE, or AS/400 platforms, IBM has developed COBOL compilers for the workstation. These products target the workstation and host application development environments.

IBM′s COBOL workstation products are:

- **IBM VisualAge for COBOL for OS/2**
- **IBM COBOL Set for AIX**

Features

These COBOL compilers, IBM VisualAge for COBOL for OS/2 and IBM COBOL Set for AIX, are available today and provide full Year2000 support. They provide ANSI COBOL Standard Intrinsic Functions which give full date manipulation capability with 4-digit-year support.

Companies need productive application development environments. A language compiler is only a portion of what application programmers need today to develop and maintain code. Tools that generate statistics from code; modularize, migrate, or restructure code; or search class libraries; are becoming more and more important.

IBM VisualAge for COBOL for OS/2

IBM VisualAge for COBOL for OS/2 provides the COBOL programmer with 32-bit Direct-to-SOM based, object-oriented support on the OS/2 operating system. In addition, a COBOL application development environment is provided that is designed specifically to handle client/server, mission-critical, line-of-business applications through visual development. IBM VisualAge for COBOL for OS/2 also gives the COBOL programmer a set of high-productivity, OS/2-based power tools for the development of applications targeting OS/2 execution systems.

Local and Remote Data Access: IBM VisualAge for COBOL for OS/2 provides local and remote data access to data including:

- IBM SMARTdata UTILITIES (SdU) which provides:
  - Record oriented file access through standard COBOL I/O statements to
    - local OS/2 VSAM files
    - remote MVS VSAM, SAM, PDS, and PDSE files
    - remote OS/400 Record Files
    - remote CICS managed VSAM files on MVS using CICS/DDM
  - A full set of data conversion API′s for converting single, double and mixed byte character strings, numerics and complex structured records
  - A full set of SMARTsort API′s for sorting, copying, and merging record and byte files located locally or remotely
- Direct access to data managed by BTRIEVE.
- Support for local and remote DB2 data access using DB2 for OS/2
- Support for local and remote CICS data access using CICS for OS/2 or CICS Client for OS/2

Visual GUI Designer: The visual GUI designer provides capabilities to allow the building of complex CUA-compliant screens. The visual interface (GUI work screen) that creates the GUI code, is an easy-to-use, intuitive tool for creating graphical interfaces, eliminating the need for in-depth GUI development knowledge. Programmers can create applications by selecting controls from the control palette, moving them onto the design editor, thereby providing an integrated "what you see is what you get" (WYSIWYG) user interface. Either during or after this brief development process, developers may build the application by coding in COBOL logic with COBOL sensitive edit/compile/debug tools.

WorkFrame/2: IBM WorkFrame/2 provides seamless integration of all the components included in the IBM VisualAge for COBOL for OS/2 product. WorkFrame/2 is a highly configurable, project-oriented application development environment for use on OS/2 and is specifically designed to take full advantage of the features offered by OS/2. When used as the integration medium for application development tools, the fully configurable IBM WorkFrame/2 increases the effectiveness of these tools as agents for enhancing user productivity. IBM WorkFrame/2 organizes the programmer's workplace by grouping files into logical units or projects. As an organizer, IBM WorkFrame/2

- Adapts to the user's project organization environment instead of the project organization having to fit into the WorkFrame defined environment.
- Sets up projects to consist of source and object files spanning multiple directories, and one target directory, (such as, .EXE or .DLL).
- Associates each project with multiple actions, including compiling, debugging, making, linking, browsing, profiling/analyzing, and preprocessing.
- As a tools integrator, multiple developers can now work concurrently on a single project by plugging in their own source control system.

Context Sensitive Editor: The LPEX Editor is a language-sensitive, color editor which supports COBOL. The LPEX Editor can be used to create and edit many types of text files, including program source and documentation. By automatically parsing COBOL source code, LPEX distinguishes between language constructs. For instance, language keywords, comments, string literals, and numbers are displayed using distinctive fonts and colors. Developers can quickly find items they are looking for in their source code. Using LPEX, developers can:

- Be made aware of some syntax errors when the source code is created
- Use multiple windows to display several documents or to display more than one view of the same document
- Dynamically configure LPEX to be a multiple-window or single-window tool
- Select a block of text and move or copy it between documents
- Cut and paste to a shell or another application
- Undo previous changes to a document

Developers can customize and extend virtually every aspect of this programmable editor. LPEX is designed to be extended through dynamic link libraries; there is no proprietary extension language to learn. With the LPEX application programming interface (API), developers can write powerful extensions to the editor using C and C++. In addition, LPEX provides a rich command language that developers can use to create or modify editor functions. Developers can:

- Define their own fonts and colors
- Modify the editor action key layout
- Add menus to perform frequently used commands (menu definitions can be applied on a filename extension basis)

- Write their own editor commands

Interactive Debug Tool for OS/2 (IDbug): The debug tool supplied with IBM VisualAge for COBOL for OS/2 provides source level debugging built around a set of core functions designed to let users quickly and efficiently control execution, and analyze data. Users can:

- Display and change variables
- Display and change storage
- Display and change the processor registers
- Display the call stack
- Add and delete simple and complex breakpoints
- Control the execution of multiple threads
- View source code as a listing, disassembly or mixed

CICS for OS/2 Version 3.0 transactions built with IBM VisualAge for COBOL for OS/2 can be debugged interactively.

For PM application programming support synchronous and asynchronous modes gives users two ways to debug PM applications. The application windows can be managed concurrently with the debug tool windows.

Performance Tuning: Execution trace analysis and performance tuning is provided through the IBM Performance Analyzer. It is designed to help users tune and understand their programs by monitoring program execution and generating a function-by-function trace of the run. This trace can subsequently be examined by utility programs that graphically display the execution trace. Not only does the analyzer trace procedures in the . EXE file, but it traces the entry points to system calls and application DLLs.

Data Assistant and Transaction Assistant: Data Assistant simplifies the process of constructing syntactically correct, embedded SQL statements. It gives you a graphical view of your relational database, allows you to map COBOL data structures to the database and generate SQL statements into your source file. Transaction Assistant enables non-CICS COBOL applications to access CICS transactions.

Product Positioning: IBM′s object-oriented COBOL family of products combine the technical and practical advantages of the COBOL language with the benefits of object-oriented programming. The fact that these products extend the COBOL language, the world′s most popular conventional programming language, gives them four distinct advantages over other object-oriented programming languages:

1. The ability to readily interface to, and extend, existing COBOL applications

2. An abundance of developers with COBOL programming skills (which increases the likelihood that object-oriented COBOL will become the most widely-used object-oriented programming language in the industry)

3. The natural synergy of COBOL, transaction processing and database access via object-oriented programming, provides an compelling reason to use IBM′s object-oriented COBOL offerings.

4. The object-oriented features are a natural extension of COBOL. It now will be easy for COBOL developers to get started with object-oriented COBOL while using the same robust COBOL development environment they have grown to depend on.

Object-oriented COBOL applications can work with existing COBOL applications, as well as SOM objects written in other languages.

IBM VisualAge for COBOL for OS/2 works with TeamConnection to store COBOL source files, build COBOL applications, provides version control and change management for applications in production.

Year2000 Support: Most applications were coded with 2-digit-year data. IBM VisualAge for COBOL for OS/2 provides intrinsic functions not available in our earlier COBOL compilers. Calendar functions include:

- CURRENT-DATE
- DATE-OF-INTEGER
- DAY-OF-INTEGER
- INTEGER-OF-DATE
- INTEGER-OF-DAY
- WHEN-COMPILED

Example of obtaining the 4-digit year with the COBOL Intrinsic Functions:

- DATE-OF-INTEGER gives YYYYMMDD
- DAY-OF-INTEGER gives YYYYDDD

IBM COBOL Set for AIX

IBM COBOL Set for AIX provides the COBOL programmer with Direct-to-SOM-based, object-oriented support on the AIX operating system. IBM COBOL Set for AIX provides a COBOL application development environment that is designed specifically to create client/server, mission critical, line-of-business applications. IBM COBOL Set for AIX also gives the COBOL programmer a set of high-productivity, AIX-based tools for the development of applications targeting AIX execution systems.

IBM COBOL Set for AIX includes local and remote data access, a context sensitive editor (LPEX), Program Builder, Program Debug Tool, Common Desktop Integration of these tools, and COBOL online documentation.

Local and Remote Data Access: IBM COBOL Set for AIX provides local and remote data access to data including:

- IBM SMARTdata UTILITIES (SdU) which provides:
  - Record oriented file access through standard COBOL I/O statements to
    - local AIX VSAM files
    - remote MVS VSAM, SAM, PDS, and PDSE files
    - remote OS/400 files
    - remote CICS managed VSAM files on MVS using CICS/DDM
    - data managed by IBM ENCINA for AIX Shared File System.
- Support for local and remote DB2 data access using DB2 for AIX.
- Support for local and remote CICS data access using CICS for AIX or CICS Client for AIX

Context Sensitive Editor: The LPEX Editor is a language-sensitive, color editor which supports COBOL. The LPEX Editor can be used to create and edit many types of text files, including program source and documentation. By automatically parsing COBOL source code, LPEX distinguishes between language constructs. For instance, language keywords, comments, string literals, and numbers are displayed using distinctive fonts and colors.

Developers can quickly find items they are looking for in their source code. Using LPEX, developers can:

- Be made aware of some syntax errors when the source code is created
- Use multiple windows to display several documents or to display more than one view of the same document
- Dynamically configure LPEX to be a multiple-window or single-window tool
- Select a block of text and move or copy it between documents
- Cut and paste to a shell or another application
- Undo previous changes to a document

Developers can customize and extend virtually every aspect of this programmable editor. LPEX is designed to be extended through shared libraries; there is no proprietary extension language to learn. With the LPEX application programming interface (API), developers can write powerful extensions to the editor using C and C++. In addition, LPEX provides a rich command language that developers can use to create or modify editor functions. Developers can:

- Define their own fonts and colors
- Modify the editor action key layout
- Add menus to perform frequently used commands (menu definitions can be applied on a filename extension basis)
- Write their own editor commands

Program Builder: The Program Builder manages the repetitive tasks of compiling, linking, and correcting errors in program source code. The Program Builder:

- Provides a graphical user interface to simplify the process of setting and saving compile and linker options.
- Lists build errors in a window. Selecting a compile error in the list will position you at the error in the source code in the LPEX Editor.
- Creates a makefile that is used by the AIX make command to construct and maintain programs and libraries. The Program Builder also determines build dependencies by scanning the source code files for dependency information.

Common Desktop Environment (CDE): End-user productivity is enhanced on AIX Version 4 with a new user interface for the AIXwindows (R) Desktop which is based on the Common Desktop Environment. This new graphical user interface is included on both the AIX for Clients and the AIX for Servers packages.

The Common Desktop Environment (CDE) integration for COBOL Set for AIX consists of a COBOL application folder which is integrated within the CDE Application Manager. The COBOL Set for AIX application folder contains icons representing the COBOL tools and applications. The COBOL application folder will contain icons for the LPEX editor, the Program Builder, the Program Debug Tool, and COBOL Online Documentation.

CDE Integration of the COBOL tools will allow the user to invoke the tools in a simple and consistent manner. The CDE desktop recognizes different types of files using a data type database. A data type identifies the files of a particular format and associates them with the appropriate applications. These associations mean that users don′t have to remember command line invocations of tools. In most cases when a user double-clicks on a file, the CDE desktop will automatically launch the correct application that understands that file′s data.

Program Debug Tool: The COBOL Set for AIX debug tool helps you detect and diagnose errors in code developed using the COBOL Set for AIX compiler. This intuitive graphical user interface allows you to control execution of the program, examine and modify data (variables, storage, and registers), and perform many other useful functions. Additionally, you can debug C functions that your applications may be using.

The debug tool provides source-level debugging and is built around a set of core functions designed to let developers quickly and efficiently control execution and analyze data. With these core functions, developers can:

- Display and change variables
- Display and change storage
- Display and change the processor registers
- Display the call stack
- Add and delete simple and complex breakpoints
- Control the execution of multiple threads
- View source code as listing, disassembly or mixed

Product Positioning: IBM′s object-oriented COBOL family of products combine the technical and practical advantages of the COBOL language with the benefits of object-oriented programming. The fact that these products extend the COBOL language, the world′s most popular conventional programming language, gives them four distinct advantages over other object-oriented programming languages:

1. The ability to readily interface to, and extend, existing COBOL applications
2. An abundance of developers with COBOL programming skills (which increases the likelihood that object-oriented COBOL will become the most widely-used object-oriented programming language in the industry)
3. The natural synergy of COBOL, transaction processing and database access via object-oriented programming, and provides a compelling reason to use IBM′s object-oriented COBOL offerings
4. The object-oriented features are a natural extension of COBOL. It now will be easy for COBOL developers to get started with object-oriented COBOL while using the same robust COBOL development environment they have grown to depend on.

Object-oriented COBOL applications can work with existing COBOL applications, as well as SOM objects written in other languages.

Year2000 Support: Most applications were coded with 2-digit-year data. IBM COBOL Set for AIX provides intrinsic functions not available in our earlier COBOL compilers. Calendar functions include:

- CURRENT-DATE
- DATE-OF-INTEGER
- DAY-OF-INTEGER
- INTEGER-OF-DATE
- INTEGER-OF-DAY
- WHEN-COMPILED

Example of obtaining the 4-digit year with the COBOL Intrinsic Functions:

- DATE-OF-INTEGER gives YYYYMMDD
- DAY-OF-INTEGER gives YYYYDDD

The IBM PL/I Family for the Workstations

For application code written in PL/I for the workstation or developed for the MVS, VM, VSE, or AS/400 platforms, IBM has developed PL/I compilers for the workstation. These products target the workstation and host application development environments.

IBM′s PL/I workstation products are:

- **IBM PL/I for OS/2**
- **IBM PL/I Set for AIX**

Features

These PL/I compilers, IBM PL/I for OS/2 and IBM PL/I Set for AIX, are available today and provide full Year2000 support. They provide a Built-in Function which provides 4-digit-year support.

Companies need productive application development environments. A language compiler is only a portion of what application programmers need today to develop and maintain code. Tools that generate statistics from code; modularize, migrate, or restructure code; or search class libraries; are becoming more and more important.

IBM PL/I for OS/2

IBM PL/I for OS/2 (10H7848, 10H7819, 1322966, 5622-088) provides the PL/I programmer with 32-bit support on the OS/2 operating system. In addition, a PL/I application development environment is provided that is designed especially to handle mission critical, line-of-business applications through visual programming and construction-from-components technologies. PL/I for OS/2 is offered in three options - a Personal Edition, a Professional Edition, and a Toolkit.

IBM PL/I for OS/2 Personal Edition: IBM PL/I for OS/2 Personal Edition was designed for small software development companies, consultants, and students. The Personal Edition contains a full 32-bit compiler, run-time library, and graphical, interactive debugging facility that supports new PL/I application development on stand-alone PCs or small LANs.

IBM PL/I for OS/2 Professional Edition: IBM PL/I for OS/2 Professional Edition includes all of the features of the Personal Edition, plus additional function that enhances compatibility with the mainframe compiler. Developers can build and test host-based VSAM, DB2 (R), CICS, and IMS applications, or they can take advantage of PL/I for OS/2 (in combination with DB2/2, CICS OS/2, and IMS Client Server/2) to create client/server applications that integrate with existing mainframe programs.

IBM PL/I for OS/2 Toolkit: IBM PL/I for OS/2 Toolkit helps you streamline the programming process, offering a development environment for PL/I for OS/2 presentation manager (PM) applications. The Toolkit is a collection of tools designed to complement either the Professional or Personal Edition of PL/I for OS/2. The Toolkit includes: a visual prototyping tool and code generator that can help you develop graphical user interfaces, a programming aid designed to help you convert C header files to PL/I header files, and the OS/2 Developer′s Toolkit.

IBM PL/I Set for AIX

IBM PL/I Set for AIX (33H1858, 33H5425, 5765-549) provides a PL/I application development environment designed to allow you to create mission critical, line-of-business applications that can run on host systems, workstations, or client/server systems with access to DB2(R), CICS, VSAM/SAM, and other data systems. IBM PL/I Set for AIX provides the PL/I programmer with an optimizing compiler and a set of high-productivity, AIX-based tools integrated with the AIX Common Desktop Environment, for the development of applications.

PL/I Compiler: IBM PL/I Set for AIX provides an optimizing compiler that contains a rich implementation of the PL/I language as well as support to improve compatibility with mainframe PL/I and enhancements that can allow new AIX-based applications to take advantage of features of the AIX platform.

The PL/I compiler also includes powerful, integrated preprocessors. You can select from one or more of the preprocessors as required for use in your program. The preprocessors included are:

- the macro facility
- the include preprocessor that allows you to incorporate external source files
- the SQL preprocessor that translates embedded SQL statements into PL/I statements, providing support for local and remote DB2 data access when used with DB2 for AIX
- the CICS preprocessor that translates embedded CICS statements into PL/I statements, providing support for local and remote CICS data access when used with CICS for AIX or CICS Client for AIX

A choice of linkages and parameter-passing mechanisms is also provided to facilitate interlanguage communications (ILC) between your PL/I routines and C/C++, Fortran, Pascal, and COBOL routines on AIX.

PL/I Set for AIX supports a single-byte character set (SBCS) and a double-byte character set (DBCS).

Common Desktop Environment: IBM PL/I Set for AIX utilizes the new graphical user interface (GUI), based on the Common Desktop Environment (CDE), in IBM AIX Version 4.1. The CDE integration consists of a PL/I application folder which is integrated within the CDE Application Manager. The PL/I Set for AIX application folder contains icons representing the PL/I tools and applications. CDE Integration of the PL/I tools will allow the user to invoke the tools in a simple and consistent manner. The CDE desktop recognizes different types of files using a data type database. A data type identifies the files of a particular format and associates them with the appropriate applications. These associations mean that users don′t have to remember command line invocations of tools. In most cases when a user double-clicks on a file, the CDE desktop will automatically launch the correct application that understands that file′s data.

The PL/I application folder contains:

- Live Parsing Extensible (LPEX) editor
- Program Builder
- Debug Tool
- PL/I online documentation

LPEX Editor: The LPEX editor is a language-sensitive editor which supports PL/I. The LPEX editor can be used to create and edit many types of text files, including program source and documentation. Using LPEX, developers can:

- Use multiple windows to display several documents or to display more than one view of the same document
- Dynamically configure LPEX to be a multiple-window or single-window tool
- Select a block of text and move or copy it between documents
- Cut and paste to a shell or another application
- Undo previous changes to a document

Developers can customize and extend virtually every aspect of this programmable editor. LPEX is designed to be extended through dynamic link libraries. There is no proprietary extension language to learn. With the LPEX application programming interface (API), developers can write powerful extensions to the editor. In addition, LPEX provides a rich command language that developers can use to create or modify editor functions. Developers can:

- Define their own fonts and colors
- Modify the editor action key layout
- Add menus to perform frequently used commands (menu definitions can be applied on a filename extension basis)
- Write their own editor commands

Program Builder: The Program Builder manages the repetitive tasks of compiling, linking, and correcting errors in program source code. The Program Builder is designed to:

- Provide a graphical user interface to simplify the process of setting and saving compile and linker options.
- Lists build errors in a window. Selecting a compile error in the list will position you at the error in the source code in the LPEX Editor.
- Creates a makefile that is used by the AIX make command to construct and maintain programs and libraries. The Program Builder also determines build dependencies by scanning the source code files for dependency information.

Debug Tool: The debug tool helps you detect and diagnose errors in code developed using the PL/I Set for AIX compiler. The intuitive graphical user interface allows you to control execution of the program, examine and modify data (variables, storage, and registers), and perform many other useful functions.

The debug tool provides machine-level and source-level debugging. It is built around a set of core functions designed to let developers quickly and efficiently control execution, and analyze data. With these core functions, developers can:

- Display and change variables
- Display and change storage
- Display and change the processor registers
- Display the call stack
- Add and delete simple and complex breakpoints
- Control the execution of multiple threads
- View source code as listing, disassembly or mixed

CICS for AIX Version 2.1 transactions built with IBM PL/I Set for AIX can be debugged interactively.

Local and Remote Data Access: IBM PL/I Set for AIX provides the ability to write applications that support local and remote access to data including:

- The IBM SMARTdata UTILITIES (SdU) which are designed to provide record oriented file access through standard PL/I I/O statements to
  - local AIX VSAM files

  - remote MVS VSAM, SAM, PDS, and PDSE files
  - remote OS/400 files
  - remote CICS managed VSAM files on MVS through CICS/DDM
- Support for local and remote DB2 data access using DB2 for AIX
- Support for local and remote CICS data access using CICS for AIX or CICS Client for AIX

Solution Developer Tools

Figure 7-5 lists Solution Developer analysis/re-engineering/development tools that aid in reformatting the year-date notation.

Disclaimer

The following is a partial listing of some Solution Developers who represented to us that they provide tools to assist in achieving Year2000 readiness. IBM does not make any representations, endorsement, or warranties of any of the following products. The following information was provided by the Solution Developer and no effort has been made by IBM to independently verify the accuracy of the information relating to the Year2000 date transition. The reader is responsible for checking with the individual Solution Developer to confirm the specific implementation of the Year2000 date transition.

If your company markets a tool, and you would like IBM to consider listing that tool here, please complete and return (by mail or FAX) the "Year2000-Ready Solution Developer Product and Tool Authorization" form provided at the back of this book.

Solution Developer Contacts and Product Name

Figure 7-5 (Page 1 of 4). Solution Developer Tools

Company Name	Contact	Product Name	Supported Platform(s)
ADPAC Corp.	425 Market Street, 4th Floor San Francisco, CA 94105 Tel: 415-777-5400 Fax: 415-546-7130	SystemVision YEAR 2000	OS/390 all MVS
COGNICASE	425 Viger Ouest Suite 303 Montreal, Quebec H2Z 1X2 Canada Tel: 800-322-3386 Tel: 514-866-6161 Fax: 514-866-6260	Cogni-2000	DOS OS/2 UNIX
Computer Software Corp.	19100 Detroit Road Cleveland, OH 44116 Tel: 800-908-2000 Tel: 216-333-4420 Fax: 216-333-8288	DateServer 2000	OS/390 MVS VSE
Compuware Corp.	31440 Northwestern Highway Farmington Hills, Michigan 48334 Tel: 800-535-8707 Tel: 810-737-7300 Fax: 810-737-2718	PATHVU RETROFIT XPEDITER/TSO XPEDITER/CICS XPEDITER+ XPEDITER/Xchange	OS/390 all MVS/ESA XPEDITER+ (also OS/2)
DATE/2000	P.O. Box 600084 San Diego, CA 92160-0084 Tel: 619-282-6792 Fax: 619-454-1643 email: date2000@aol.com	DATE/2000	System 34 System 36 AS/400 Advanced 36 AS/400 System 36

Figure 7-5 (Page 2 of 4). Solution Developer Tools			
Company Name	**Contact**	**Product Name**	**Supported Platform(s)**
EDGE Information Group	2250 East Devon Des Plaines, IL 60018-4510 Tel: 513-948-8906	Edge Portfolio Analyzer	
Global Software, Inc.	P.O. Box 2813 15 Depot Street Duxbury, MA 02331-2813 Tel: 617-934-0949 Fax: 617-934-2001 e-mail: GILES@globsoft.com	GILES	OS/390 MVS/ESA MVS/XA
HAL Informatica srl	Via Spoleto 4 20125 Milano Italia Tel: +39(2)28.015.1 Fax: +39(2)26.140.165 e-mail: HALINFO@HALINFO.IT	BYPASS2000	OS/390 MVS/ESA AS/400
Information Builders, Inc.	1250 Broadway New York, NY 10001 Tel: 212-736-4433 Fax: 212-629-3612 e-mail: PMSPAK@IBI.COM	FOCUS	OS/390 MVS
Informission Group, Inc.	7811 L. H. LaFontaine Blvd., Suite 200 Anjou, Quebec H1K 4E4 Canada Tel: 514-351-7755 Fax: 514-351-5845 e-mail: RECY2000@INFORMISSION.CA	RECY2000	AIX OS/390 MVS OS/2 OS/400 VM VSE
Ironsoft, Inc.	4323 Winnequah Rd. Monona, WI 53716 Tel: 800-236-0141 Tel: 608-695-1896 Fax: 608-221-8018	Analyzer 2000	OS/390 MVS OS/2
ISOGON Corp.	330 Seventh Avenue New York, New York 10001 Tel: 800-568-8828 Tel: 212-376-3200 Fax: 212-376-3280	TICTOC SOFTAUDIT	OS/390 MVS/ESA MVS/SP V3
Izar Associates, Inc.	4 Emery Avenue Randolph, NJ 07869 Tel: 201-442-0577 Fax: 201-442-0633	Fieldex 2000	OS/390 MVS VSE
Mainware, Inc.	7176 Pioneer Creek Road Maple Plain, MN 55359 Tel: 612-932-9154 Fax: 612-932-9155	HourGlass 2000	OS/390 MVS/ESA MVS/XA
Micro Focus	2465 East Bayshore Road Palo Alto, CA 94303 Tel: 800-318-4259 Tel: 415-856-4161 Fax: 415-856-6134	Challenge 2000	OS/2 WINDOWS

Figure 7-5 (Page 3 of 4). Solution Developer Tools			
Company Name	**Contact**	**Product Name**	**Supported Platform(s)**
Namtrig Incorporated	P.O. Box 160757 Altamonte Springs, Florida 32716 Tel: 407-884-6771 Fax: 407-884-6755 e-mail: namtrig@iag.net	ADAMS/400	AS/400 V2R3+
PRINCE Software, Inc.	1000C Lake Street Ramsey, NJ 07446 Tel: 800-934-2022 FAX: 201-934-0220 e-mail: PORTAL2000@ PRINCESOFTWARE.com	PORTAL 2000 / SIMULATE 2000 PORTAL 2000 / SURVEY 2000 PORTAL 2000 / TRANSLATE 2000	OS/390 MVS
Princeton Softech	1060 State Road Princeton, NJ 08540 Tel: 609-497-0205 FAX: 609-497-0302 e-mail: INFO@PrincetonSoftech.com	Version Merger Relational Tools Suite	MVS/ISPF 2.3+
Quintic Systems, Inc.	3166 Des Plaines Avenue, Suite 36 Des Plaines, Illinois 60018 Tel: 800-699-1169 Tel: 708-699-1169 Fax: 708-699-1214	Century File Conversion Century Source Conversion	OS/390 MVS VSE/ESA
Reasoning Systems	3260 Hillview Avenue Palo Alto, CA 94304 Tel: 415-494-6201 Fax: 415-494-8053	Software Refinery REFINE/COBOL	AIX
SEEC, Inc.	5001 Baum Blvd. Pittsburgh, PA 15213 Tel: 412-682-4991 Fax: 412-682-4958	COBOL Analyst	OS/2 WARP
Software Eclectics, Inc.	Suite 131 10955 Jones Bridge Rd. Alpharetta, GA 30202-7343 Tel: 800-457-3113 Tel: 770-667-9117 Fax: 770-667-9417	SE/One	
Software Migrations Limited	Unit 1C Mountjoy Research Center Stockton Road Durham, DH1 3SW United Kingdom Tel: 44+(0)191 386 0420 Fax: 44+(0)191 386 1243	FermaT 2000	AIX
The Source Recovery Co., Inc.	992 E. Freeway Drive, Suite A Conyers, GA 30207-5916 Tel: 770-785-9801 FAX: 770-760-7316 e-mail: 72604,2340@Compuserve.com	Source Recovery Services	OS/390 MVS VM VSE
Tech-Beamers	21 Davis Ave. Poughkeepsie, NY 12603 Tel: 914-473-4444 FAX: 914-473-4478	Year 2000 S/390 Workstation	OS/390 MVS/ESA 5.2.2 VM/ESA VSE/ESA

Figure 7-5 (Page 4 of 4). Solution Developer Tools

Company Name	Contact	Product Name	Supported Platform(s)
TechForce BV	Saturnusstraat 60 P.O. Box 3108 2132 HB Hoofddorp The Netherlands Tel: +31.23.56.22929 Fax: +31.23.56.27052 e-mail: sales@techforce.nl	COSMOS Year 2000 Workbench	AIX OS/390 MVS OS/2 OS/400 VM VSE
TransCentury Data Systems	111 Pine Street, Suite 715 San Francisco, CA 94111 Tel: 415-255-7082 Fax: 415-255-4584	TransCentury Calendar Routines	
VIASOFT	3033 N. 44th Street Phoenix, AZ 85018 Tel: 800-525-7775 Tel: 602-952-0050 Fax: 602-840-4068	Enterprise 2000	OS/390 MVS
Visionet Systems Inc.	P.O. Box 316 1103 Pleasant Rd. Monmouth Jct., NJ 08852 Tel: 908-329-8090 Fax: 908-329-1712 e-mail: visionet@superlink.net	Millennium/400	OS/400

Chapter 8. IBM Consulting and Services

This section includes descriptions of two IBM consulting and service offerings:

- TRANSFORMATION 2000 (a comprehensive set of software and hardware offerings specific to assist your YEAR 2000 efforts)
- Availability Services (a suite of migration services).

TRANSFORMATION 2000: IBM′s Century Date Change Solutions

Note: The following is extracted directly from IBM announcement letter 25744 dated 31 October 1995. See that IBM document for complete announcement information.

IBM recognizes that the Year 2000 change poses a significant challenge for the Information Technology industry. To help with this change IBM′s Information Systems Solution Corporation (ISSC) has developed a comprehensive set of solutions that takes into account applications, systems software and hardware in both centralized and distributed environments. This comprehensive solution set is called TRANSFORMATION 2000.

The TRANSFORMATION 2000 approach seeks to balance Year 2000 investment activities with current and planned strategic initiatives (for example, new architectures and new application development). ISSC brings together state-of-the-art techniques and technologies developed and proven through both internal and external projects for the Year 2000 and other data field expansions. This experience enables ISSC to help reduce both the cost and the complexity of implementing the Year 2000 change.

Assessment and Strategy

The Assessment and Strategy solution is an 8- to 12-week engagement that will identify the magnitude of the Year 2000 problem. This solution covers a comprehensive range (5 to 150+ million lines of code) of portfolios in a mixed language environment. ISSC′s approach allows for analysis across the entire client portfolio thereby identifying logical work partitions from which the implementation plan will be formulated, helping to reduce the overall risk of the project.

ISSC uses a optimal combination of commercially available and internally developed tools depending on the client environment.

At the completion of the project, the client will have a documented strategy, general cost estimates, timeframes, and resource estimates required to implement the Year 2000 change.

Detailed Analysis and Planning

The Detailed Analysis and Planning solution provides an in-depth analysis of each of the Year 2000-affected areas and identifies detailed work partitions needed to maximize the potential for automated change. This review includes not only the application code, but file structures and other environmental objects.

The output from this engagement includes detailed work partitions, detailed work and resource plans, and the customized specifications needed to implement the Year 2000 changes.

Test Support Services

TRANSFORMATION 2000 test support services include the following:

- On-site end-to-end testing services
- Alternative remote test facilities
- Value-add assistance to your staff
- Testing activities performed for you, from your TRANSFORMATION 2000 analysis.

Integrated Systems Solutions Corporation (ISSC) offers a comprehensive testing service for all your application and system testing needs. This includes the resources you need for planning and implementing your test environments and easy-to-use tools for your immediate productivity to manage test cases, test libraries, and test runs. ISSC offers a comprehensive testing package, with which you can be confident, provides a thorough Year 2000 solution test.

ISSC will work with you during all phases of the TRANSFORMATION 2000 project. ISSC will:

- Develop a detailed test plan with test objectives for each of your test levels:
  - Integration tests
  - System tests
  - User acceptance tests.
- Understand you total TRANSFORMATION 2000 conversion environment and advise you how to create a test environment for it
- Design and execute test scripts and scenarios that accurately represent your application′s use both before and after the Year2000 modifications
- Capture and playback applications, as well as use a high-level scripting language to build "robust" scripts.

Platform and Operational Environment

ISSC has expertise on multiple platforms and environments. ISSC can help you define and meet your TRANSFORMATION 2000 conversion objectives relating to function, compatibility, and capacity from your host application needs to those of your entire distributed network.

Benefits to Your Organization

You can:

- Continue to use your Year2000 modifications, allowing ISSC to focus on the testing process, tools, and test execution.
- Use the test scripts, tools, and techniques for further regression and stress testing.
- Validate the performance of converted applications under different levels of user load.
- Test application functions to the level required.

Other Testing Services

ISSC provides a full range of services to support the testing environment including:

- Test process evaluation
- Test management consulting
- Stress testing
- Usability testing.

Contacts

For more information on TRANSFORMATION 2000 and other ISSC testing services, contact your ISSC service representative or ISSC at:

ISSC Testing Services
1510 Page Mill Road
Palo Alto, CA 94304

Phone: 1-800-690-4772
FAX: (415) 855-3215
Internet: ISSCTEST@VNET.IBM.COM

Implementation

The Implementation solution is unique and consists of automating the required changes to source code and data, testing, and production turnover. In addition, TRANSFORMATION 2000 includes an optional conditioning activity which will improve the overall quality of the resultant application as well as reduce the total time required for the Year 2000 changes. Conditioning includes but is not limited to the following:

- standardization/rationalization of data definitions
- segmenting procedure logic
- encapsulating date handling/developing common routines
- externalizing hard-coded data
- restructuring COBOL procedure logic
- removing system redundancy

Also included in the TRANSFORMATION 2000 solution are sophisticated testing activities:

1. After making the Year 2000 changes, the total environment must be tested to ensure it can handle 19xx data correctly.
2. While running in 199x, the system must be able to handle dates after 1999.
3. When running in the Year 2000 and after, the system needs to be tested to ensure it can handle both 19xx and 2xxx data correctly.

Year 2000 Clean Management

During the Year 2000 change process it is critical for the client to protect the investment of the application modifications. When enhancements are incorporated into the application and/or new releases of software products are implemented, the Year 2000-clean status must be maintained. For this reason, ISSC has developed the Year 2000-Clean Management solution which focuses on instituting the appropriate processes, disciplines, techniques, and technologies in order to preserve the Year 2000 investment.

Summary

Recognizing that the Year 2000 is a significant client challenge, ISSC has created a Year 2000 Center-of-Competence (CoC) function, with the initial site in Atlanta. The CoC supports ISSC and IBM client services teams in providing complete Year 2000 solutions to our clients.

Further information on the TRANSFORMATION 2000 solution set is available from your IBM or ISSC representative.

IBM Availability Services

As you address your Year2000 challenge, you will need to consider a number of options when reviewing your existing software portfolio. You can retire outdated and unused applications, update those applications that can be made Year2000-ready, or upgrade to Year2000-ready software. When reviewing your system software, you might find only one option is realistic, because, for example, the vendor has announced Year2000 support for only currently maintained or yet-to-be available software releases. If your current operating system is down-level and that program product cannot provide all mission-critical function(s) in a Year2000 environment, you will be further required to upgrade or move to a different platform.

To assist you in these migration tasks, IBM provides technical support through a suite of service offerings referred to as **availability services**. These services are available to assist your migration to more current and Year2000-ready software, and most can be customized to meet your enterprise's specific needs. Figure 8-1 on page 8-4 lists several priced, platform-specific availability services and the platform(s) they serve. All are available from *IBM Direct* with a phone call to 1-800-IBM-CALL (1-800-426-2255). These services are available in the United States only.

Figure 8-1. IBM Availability Services and Their Supported Platform(s)

Platform	**Availability Service**	**Page**
AIX	IBM AIX Upgrade Services	8-5
AS/400	IBM AS/400 SoftInstall	8-6
AS/400	IBM AS/400 SysMigration	8-7
CICS	IBM CICS Application Rehosting Services	8-8
OS/390 MVS MVS/ESA VM/ESA VSE/ESA	SoftwareXcel Installation Express	8-9
AIX Lotus Notes OS/2 RISC ⋮	IBM SmoothStart & Migration	8-10
General Support	IBM House Call	8-12

IBM AIX Upgrade Services

IBM AIX Upgrade Services provides a wide range of AIX program support to meet you needs, including an on-site specialist to assist you with the following migrations:

- AIX 3.1.5 to 3.2.5
- AIX 3.2.x to 3.2.5
- AIX 3.2.x to 4.1.x (AIX 4.1.3 is Year2000 compliant)

Platform

AIX on RS/6000 (RISC System/6000)

Benefits

- IBM specialists to help determine your AIX upgrade requirements
- Expert assistance with installation and configuration considerations
- Multiple upgrade options to meet your unique needs

Contacts

- IBM Direct: 1-800-IBM-CALL (1-800-426-2255)
- USA IBMLink users can receive more information by contacting their client representative or calling: 1-800-IBM-4YOU

IBM AS/400 SoftInstall

The AS/400 SoftInstall program provides skilled IBM representatives that can make your software release upgrade simple. This service will perform installation of a new release of the AS/400 Operating System, HIPER (high impact pervasive) PTFs, and cumulative PTFs on your system with minimal impact to your business.

An on-site IBM specialist will:

- Research code and PTFs for stability
- Review publications to make sure they are correct for your new software release
- Verify sufficient disk capacity for the release upgrade
- Install the new release and associated fixes
- Provide planning and guidance on performing a full-system backup
- Provide your system administrator with information in the new system enhancement

Platform

AS/400

Benefits

- Reduced resource spent on performing installation
- Increased productivity by rapid installation
- Assurance of quality installation

Contacts

- IBM Direct: 1-800-IBM-CALL (1-800-426-2255)
- USA IBMLink users can receive more information by contacting their client representative or calling: 1-800-IBM-4YOU
- For FAX Doc: call 800-426-4329 - Fax document # 7613697

IBM AS/400 SysMigration

IBM AS/400 SysMigration helps you migrate your data and applications from an existing system to another system using the ‘tape to restore’ methodology. The new system can be migrated to the same release or to a different release. AS/400 SysMigration helps ensure minimum downtime on the production system, reorganizes your files, and allows both systems to be available upon completion. The price is tailored based on the customized services you choose.

Platform

AS/400

Benefits

- Your data moved to your new AS/400 system
- User profiles restored
- Libraries restored
- Document library objects (DLO) restored
- Authority restored
- Synchronization of the two systems to put the new system into production
- Reduces down time on your production system
- Reorganizes and cleans up your data files
- New and old systems run in parallel while new system is evaluated
- Reduces risk involved with normal data migration

Contacts

- IBM Direct: 1-800-IBM-CALL (1-800-426-2255)
- USA IBMLink users can receive more information by contacting their client representative or calling: 1-800-IBM-4YOU

IBM CICS Application Rehosting Services

The CICS Application Rehosting service offering delivers a CICS any-to-any application migration and rehosting service. This service provides migration services for existing mainframe applications across several IBM platforms.

Platform

- AIX
- OS/400
- OS/2
- Windows

Benefits

- Expert CICS consulting skills to assist implementing CICS
- Turnkey application migration tailored to your CICS environment
- Minimal migration downtime with a proven methodology

Contacts

- IBM Direct: 1-800-IBM-CALL (1-800-426-2255)
- To receive more information contact your IBM marketing representative or business partner or call: 1-800-IBM-4YOU

SoftwareXcel Installation Express

SoftwareXcel Installation Express (SIE) is an IBM offering designed to help you install new levels of software technology more efficiently and quickly in the large system operating environment.

SoftwareXcel Installation Express provides pre-built systems tailored to your hardware and software configuration and includes on-site planning, installation, and testing of the package. Post-installation support services are provided for 30 days following installation completion.

IBM offers SoftwareXcel Installation Express to address many of the concerns and inhibitors facing enterprises such as yours. It can be used to install or upgrade to new levels of IBM′s strategic computer operating systems.

Platform

- OS/390 MVS
- MVS/ESA
- VM/ESA
- VSE/ESA

Benefits

- Significantly reduce your system programming staffs′ time spent in software upgrading tasks, and relieves your programming staff of the time spent researching and applying service recommended with an upgrade.
- Reduces dedicated computer time and system disruption because a significant amount of the upgrade work is performed by IBM.
- Permit your programming staffs to dedicate their skills to maintaining and enhancing the business functions of the I/S systems.
- Offload installation integration of IBM and non-IBM software products and services into a single functioning system. IBM will notify you of vendor version, release, or service level requirements necessary for installation.
- Provide complete information for licensed software planning, customization, and installation activities performed as part of SoftwareXcel Installation Express.

Contacts

- IBM Direct: 1-800-IBM-CALL (1-800-426-2255)
- For FAX Doc: call 800-426-4329 - Fax document # 7615063

IBM SmoothStart & Migration

IBM SmoothStart services are installation services designed to accelerate the productive use of your IBM solutions by delivering:

- Project management
- Planning
- Pre-installation software configuration (if applicable)
- Software installation
- Software configuration
- Operational customization (optional)
- SmoothStart installation record
- Training

Platform

- IBM hardware and/or software
- non-IBM hardware and/or software

SmoothStart covers a wide range of software servers:

- AIX
  - Communications
  - Connections
  - Database
  - Directory/Security
  - Internet connection
  - Transactions
  - SystemView
- IBM SmoothStart Architecture
- Lotus Notes
- OS/2
  - Communications
  - Database
  - Internet connections
  - Transaction
  - Warp

IBM SmoothStart Services for:

- AS/400
  - Internet Connection
  - V3R6 Transition Services
- Gopher Client Internet
- MVS - Internet Connection
- Novell - Integration Services of OS/400
- Nways ATM Campus

IBM NetView SmoothStart for:

- AIX
- NetWare
- OS/2

IBM CICS SmoothStart for:

- AIX
- CICSPlex
- OS/400
- Windows NT

IBM SmoothStart for:

- AS/400
- AS/400 Advanced 36
- MQSeries
- OS/2 - VisualAge for COBOL
- RISC

Benefits

- Gets your solution installed and running quickly
- Reduces your need for acquiring new skills and resources
- Provides on-site education and training

Contacts

- IBM Marketing Representative or IBM Availability Services Specialists
- For FAX Doc: call 800-426-4329 - Fax document # 7612551 & 7612552

IBM House Call

The IBM House Call service can help you supplement critical skills needed to perform on-site hardware and software support. In addition to answering questions, a support specialist can perform specific services pertaining to the installation, usage and performance tuning of supported products.

With IBM House Call, you receive:

- Access to an on-site support specialist who can assist in hardware and software installation activities, software maintenance and upgrade activities, operational services, problem assistance, and any other mutually agreed upon tasks.
- Coverage at a mutually agreed upon time (8:00 a.m. to 5:00 p.m. in your time zone, Monday through Friday, except U.S. national holidays).
- Assistance with basic operational tasks such as PTF application, release application, and performance tuning.

Platform

- PS/2
- RS/6000 (RISC/6000)
- AS/400
- OS/390
- Networking

Benefits

- Minimize customer resource needed
- Maintain lower customer skill level by leveraging with IBM
- Extra resource/skill not required for “once in a lifetime” projects
- Less expensive access to distributed customer sites

Contacts

- USA IBMLink users can receive more information by contacting their client representative or calling: 1-800-IBM-4YOU

Appendix A. IBM Year2000-Ready Key Program Products and Hardware

This section lists key IBM products and offerings and their supported level or levels that are **currently available or planned to be available** as Year2000 ready by year-end 1996. Incorporation in the lists does not imply that a product or version of a product will necessarily be current or available in 1999 or beyond.

These lists are not meant to be exhaustive or exclusive, but are provided to answer the most common questions. If a product is not listed, it does not imply it will not be Year2000 ready. IBM intends for at least the latest level of the latest version of its products to be Year2000 ready by year-end 1996. When an earlier level is also Year2000 ready, it will be indicated specifically in the lists. These lists will be updated periodically as more information becomes available.

Specific program products are listed under the IBM platform they support as follows:

- MVS
- TPF
- VSE/ESA
- VM
- OS/400
- AIX
- OS/2
- Lotus Products
- Tivoli Products

If a product is listed with a version/release number, it indicates that level of the product (with PTF if applicable) and all subsequent versions/releases will include Year2000 support. Keep in mind that new versions will have new product numbers. If just a version is listed or if no version/release number is listed, you can assume that the most recent version/release as of year-end 1996 will include the Year2000 support. Be aware that if the most current level at the end of 1996 is a different version from the one listed here, it will have a different product number.

Some products will be Year2000 ready through the service stream. The keyword YR2000 is added to the APAR and/or PTF cover letter text to assist searching. In addition to any product specific preventive service planning (PSP) buckets, there are specific Year2000 (PSP) buckets for MVS, VM, and VSE. The upgrade names are:

- YR2000MVS
- YR2000VM
- YR2000VSE

Each bucket is primed with the names of products that will have Year2000 fixes. These buckets will be updated periodically with pointers to APARs and/or PTFs as they become available (but no later than 31 December 1996). We recommend you monitor the buckets to ensure you have the correct level of Year2000 service.

For additional product information, contact an IBM representative.

Exception Considerations and Program Products

There will be a few exceptions to the general rule, typically where a product will be superseded or replaced in the relevant time period, and incorporation in the lists does not imply that a product or version of a product will necessarily be current or available in 1999 or beyond. Where a product is superseded or replaced by a new version/release by the end of 1996, the later level might not be listed here, rather only the older version. When it is appropriate to publish the later level, the list will indicate the level it has replaced.

Environmental Report Editing and Printing (EREP)

When using Environmental Report Editing and Printing (EREP) (5654-260, EREP for VM; 5656-260, EREP for VSE; and 5658-260, EREP for MVS), to produce reports where the records have been created in both 1999 and 2000, special handling is necessary to prevent problems and minimize customer impact. Because EREP data on a single LOGREC tape spans a relatively short period of time, be aware of the following limitations when processing EREP reports which include both 31 December 1999 and 1 January 2000.

- Input of the DATE parameter is restricted to a 2-digit-year specification. If you do not use the DATE parameter, and the data on a given LOGREC tape includes from both 31 December 1999 and 1 January 2000, the output is sorted with data from the year 1999 after the data from the year 2000.
- EREP does not properly process date selection if the range includes both 31 December 1999 and 1 January 2000. Therefore, if you desire to select report entries for such a range, you must create two reports. On the first report specify DATE=(yy.ddd,99.365) and on the second report specify DATE=(00.001,yy.ddd).

3 Certain tables (MVS - Figure A-1 on page A-3; TPF - Figure A-2 on page A-9;
3 VSE - Figure A-3 on page A-10; and OS/400 - Figure A-5 on page A-14) have
3 been modified to include information on whether or not a product is Year2000
3 ready today. A checkmark (√) in the Year2000-Ready column indicates that the
3 product level is already Year2000 ready. A check in the By Year-End 1996 column
3 indicates the product will be Year2000 ready by year-end 1996. An asterisk (*) in
3 the Year2000 Ready field indicates support is available through the service
3 stream.

MVS

Figure A-1 (Page 1 of 6). MVS Platform Products			
Program Product Number	**Product Name (Version/Release)**	**Year2000 Ready**	**By Year End 1996**
5668-854	ACF/NCP V4 R3 M1	√	
5665-338	ACF/SSP V3 R6	√	
5695-117	ACF/VTAM 3.3	√	
5688-216	AD/Cycle C/370 V1 R2	√	
5688-194	AD/Cycle CODE/370 V1 R2	√	
5648-020	ADSTAR Distributed Storage Manager/MVS V1 R1	√	
5655-119	ADSTAR Distributed Storage Manager/MVS V2 R1	√	
5688-021	AFP Bar Code V1 R1	√	
5648-113	AFP Font Collection V1 R1	√	
5695-068	Airline Control System (ALCS) V2 R1	√	
5685-151	AOC/MVS V1 R2 (see Note 1)	√ *	
5688-228 5688-229	APL2 V2 R2	√	
5665-348	Application Development Facility (ADF) II V2 R2	√	
5655-002	Application Support Facility V3 R1	√	
5648-018	Application System V3 R2	√	
5740-M53	Automatically Programmed Tool--Advanced Contouring V1.5.1	√	
5655-065	Batchpipes/MVS V1 R2		√
5695-045	BookManager Build V1 R3	√	
5695-046	BookManager Read V1 R3	√	
5688-224	Browsemaster V2	√	
5665-279	BTAM V1 R1	√	
5665-264	Bulk Data Transfer (BDT) V2 R1	√ *	
5635-001	CADAM Interactive Design System (and associated products)		√
5648-003	CAMkit/370 V1.1.1	√	
5627-COM	CATIA Object Manager V4.1.6 (and associated products)	√	
5734-F11	Check Processing Control System (CPCS) V1 R11 M0		√
5655-018	CICS for MVS/ESA V4 R1	√	
5695-061	CICS Application Migration Aid V1 R1	√	
5665-463	CICS/Distributed Data Management(DDM) V1 R1 M0	√	
5696-582	CICS Transaction Affinities Utility V1 R1	√	
5695-081	CICSPlex System Mgr V1 R2	√	
5695-010	CICSVR V2		√
5688-197	COBOL for MVS & VM V1 R2	√	
5688-218	Cross System Product(CSP)/370AD V4 R1	√	
5688-206	Cross System Product(CSP)/370RS V4 R1	√	
5655-121	C/C++ for MVS/ESA V3 R2	√	

MVS

Figure A-1 (Page 2 of 6). MVS Platform Products

Program Product Number	Product Name (Version/Release)	Year2000 Ready	By Year End 1996
5688-187	C/370 Compiler V2 R1	√	
5688-188	C/370 Library V2 R2	√	
5798-DLQ	Data Base Edit Facility (DBEDIT) V1 R3	√	
5787-LAR	Data Base Integrity Control Facility (DBICF) V4 R1	√	
5685-036	DataInterchange/MVS V1 R5		√
5695-076	DataInterchange/MVS CICS V1 R5		√
5655-076	Data Propogator (DPROPR)/CAPTURE V1 R2 M0	√	
5622-267	Data Propogator (DPROPR)/APPLY V1 R2 M0	√	
5696-703	Data Refresher V1 R1	√	
5685-DB2	DB2 V3 R1		√
5688-515	DB2 Administration Tool V1 R1	√	
5695-077	DB2 Automated Utility Generator V1 R2		√
5655-102	DB2 Performance Monitor(PM) V4 R1		√
5740-XC5	Development Mgmt System/CICS V1 R5	√	
5655-257	Device Support Facility (ICKDSF) V1 R16		√
5747-DS1	Device Support Facility (ICKDSF) Standalone V1 R16		√
5695-DF1	DFSMS/MVS V1 R2 (DFSMSdfp, DFSMSdss, DFSMShsm, DFSMSrmm, NFS feature, DFSMSopt feature)		√
5740-SM1	DFSORT V1 R13	√	
5655-068 5655-069	DFS/MVS V1 R1	√	
5665-290	DISOSS V3 R4		√
5685-101	Displaywrite/370 V2 R1 under MVS/TSO	√	
5685-101	Displaywrite/370 V2 R1 under MVS/CICS	√	
5648-106	Distributed Security Manager for MVS V1 R1		√
5655-103	DITTO/ESA for MVS V1 R1		√
5748-XX9	Document Composition Facility (DCF) V1 R4	√ *	
5748-XXE	Document Library Facility (DLF) V1 R3		√
5688-168	dpAccounting Mgr (DPAM) for MVS V1 R2	√	
5735-XXB	Emulation Program V1 R6 M1	√	
5695-101	Enterprise Performance Data Manager (EPDM) V1 R1	√	
5688-008	ES Connection Manager V1 R3	√	
5655-073	FASTService for MVS Runtime Library V1 R2	√	
5685-088	FASTService for MVS V1 R2	√	
5695-044	First Failure Support Technology (FFST) V1 R2	√	
5655-123	FFST Customized Services V1 R2	√	
5655-129	Flowmark for MVS V1 R1	√	
5668-890	Font Library Service Facility (FLSF) V1 R1	√	
5668-806	FORTRAN V2.5 Compiler/Library/Interactive Debug	√	

Figure A-1 (Page 3 of 6). MVS Platform Products

Program Product Number	Product Name (Version/Release)	Year2000 Ready	By Year End 1996
5688-087	FORTRAN V2.6 Compiler/Library	√	
5668-805	FORTRAN V2.6 Library only	√	
5668-989	4700 HOST SUPPORT	√	
5622-187	4700 PC Application	√	
5668-723	GDDM/IVU V1 R1	√	
5668-801	GDDM/IMD V2 R1	√	
5668-812	GDDM/PGF V2 R1	√	
5688-113	GDDM/OS2 Link V1 R0	√	
5665-356	GDDM/MVS V2.3	√	
5695-167	GDDM/MVS V3 R1 M1	√	
5740-XX7	Generalized Information System (GIS) V1 R1 M1		√
5685-105	geoManager V1 R1 M5	√	
5764-023	geoManager API V1 R1 M0	√	
5688-169	Graphical Display and Query Facility V2 R2 M1		√
5799-AXX	Graphics Attachment Support Program(GASP) V1 R3 M10	√	
5688-050	Graphics Program Generator V2		√
5697-119	Hardware Configuration Manager (HCM) V1 R1	√	
5655-068 5655-069	Hardware Configuration Definition (HCD) V5 R1	√ *	
5696-234	HLASM for MVS & VM & VSE V1 R2	√	
5685-037	Host Support Product (HSP)		√
5648-001	IGES Processor V3 R1 M2	√	
5695-041	ImagePlus for MVS/Folder Application Facility V2 R2		√
5695-042	ImagePlus for MVS/Object Distribution Manager V2 R2		√
5655-071	ImagePlus Visualinfo Library Server for MVS/ESA	√	
5655-072	ImagePlus Visualinfo Object Server for MVS/ESA	√	
5695-176	IMS/ESA V5 R1	√	
5798-CHJ	IMSASAP V1 R1	√	
5655-085	IMS Compress V1 R2	√	
5685-093	IMS Data Base Tools (DBT) V2 R1	√	
5655-109	IMS Data Entry Databases (DEDB) Fast Recovery (DFR) V1 R1	√	
5740-XXR	IMS DB Analyser V1 R1	√	
5655-038	IMS Message Requeuer		√
5798-CQP	IMS Performance Analysis and Reporting System (IMSPARS) V1 R1		√
5668-948	IMS/VS Batch Terminal Simulator (BTS) V2 R2	√	
5695-171	Info Management V6 R2	√	
5665-393	Inforem Base		√
5665-394	Inforem Allocation		√
5685-051	Integrated Cryptographic Service Facility (ICSF)/MVS V1 R1	√	

MVS

Figure A-1 (Page 4 of 6). MVS Platform Products

Program Product Number	Product Name (Version/Release)	Year2000 Ready	By Year End 1996
5685-054	ISPF V3 R5	√	
5655-042	ISPF V4 R1 and up		√
5695-080	Item Access Fac (IAFC) V1 R3	√	
5785-BAC	JES/328X Print Facility V2 R2	√	
5655-068	LAN Server for MVS	√	
5688-198	Language Environment for MVS & VM V1 R4	√	
5695-123	LANRES/MVS	√	
5655-039	MERVA/ESA V3 R2 (supports the definitions in the SWIFT User Handbook)	√	
5655-040	MERVA/MVS V3 R2 (supports the definitions in the SWIFT User Handbook)	√	
5695-137	MQSeries for MVS/ESA V1 R1 M3	√	
5655-068	MVS/ESA SP JES2 V5 R1		√
5655-069	MVS/ESA SP JES3 V5 R1 M1		√
5655-ACS	NaviQuest V1 R1	√	
5688-029	NC Toolkit V1.1.1	√	
5665-365	NetView Access Services V1 R3 M2	√	
5695-036	NetView Access Services V2 for MVS	√	
5695-169	NetView AutoBridge	√	
5685-016	NetView Distribution Manager (DM) V1 R6		√
5685-108	NetView FTP for MVS V2 R2		√
5655-044	NetView MultiSystem Manager V1 R1	√	
5655-126	NetView MultiSystem Manager V2 R1	√	
5655-007	NetView V3 R1	√	
5685-111	NetView V2 R4	√	
5665-333	NetView Performance Monitor V1 R6	√	
5655-043	NetView Performance Monitor V2 R1, V2 R2	√	
5696-583	NetView Remote Ops Manager	√	
5685-045	Network Design & Analysis(NetDA) V1 R2	√	
5735-XX7	Network Terminal Option (NTO) V1 R5 M1	√	
5648-077	NtuneMON V1 R1	√	
5648-089	NTuneNCP V1 R1	√	
5668-757	Numerical Control PostProcessor Generator V1.2.3	√	
5688-030	Numerical Control PostProcessor Generator/Execution Library V1.1.3	√	
5685-106	OfficeVision/MVS V1 R3		√
5695-007	Operations Planning and Control (OPC)/ESA V1 R3 M1	√	
5655-064	OpenEdition DCE Appl Support for MVS/ESA V1 R1	√	

Figure A-1 (Page 5 of 6). MVS Platform Products

Program Product Number	Product Name (Version/Release)	Year2000 Ready	By Year End 1996
5668-909 5668-910 5668-911	OS/VS PL/I V2 R3	√	
5740-RG1	OS/VS RPGII V1 R1		√
5645-001	OS/390 V1 R2		√
5655-104	Open Systems Adapter(OSA) Support Facility V1 R1	√	
5688-191	Overlay Generation Language (OGL/370) V1 R1 M0		√
5688-190	Page Printer Formatting Aid (PPFA) V1 R1	√	
5668-767	PASCAL Compiler and Library V1 R2		√
5668-717	PASCAL Library V1 R2		√
5665-311	PC/File Transfer V1 R1 M2	√	
8641-MYF 8641-MYC 8641-MYS 8641-NYV	PC Server 500 System/390	√ (Hardware)	√ (Software)
5695-101	Performance Reporter for MVS V1 R2	√	
5771-ABC	Pi and Specials V1 R1	√	
5688-235	PL/I for MVS & VM V1 R1	√	
5665-307	Print Management Facility (PMF) V1 R1	√	
5665-340	Print Services Access Facility (PSAF) V1 R1	√	
5655-010	ProductManager Application Svcs Mgr (and associated products) V3 R1	√	
5740-XY5	Programmed Cryptographic Facility(PCF) V1 R1	√	
5695-040	PSF/MVS V2 R2	√	
5688-015	Publishing Systems BookMaster V1 R4	√ *	
5668-721	QMF/MVS V2 R4		√
5695-039	RACF V2 R2	√	
5655-067	ReDiscovery/MVS V1 R1	√	
5648-048	Report Management and Distribution System (RMDS) V2 R2	√	
5695-013	REXX Compiler for 370 V1 R3 M0	√	
5695-014	REXX Runtime Library V1 R3 M0	√	
5655-086	REXX Development System for CICS/ESA V1 R1 M0	√	
5655-087	REXX Runtime Facility for CICS/ESA V1 R1 M0	√	
5655-084	RMF V5 R1		√
5665-366	Screen Definition Facility (SDF) II MVS V1 R4		√
5665-488	SDSF V1 R6	√	
5695-070	SearchManager/370 under CICS/MVS V1 R2	√	
5665-397	Service Level Reporter V3 R3 M1		√
5668-949	SMP/E V1 R8 M0	√	
5668-949	SMP/E V1 R8 M1	√	

Figure A-1 (Page 6 of 6). MVS Platform Products

Program Product Number	Product Name (Version/Release)	Year2000 Ready	By Year End 1996
5665-345	SNA Application Monitor (SAMON) V1 R0	√	
5696-822	SOMobjects for MVS V1 R1	√	
5655-146	SystemView for MVS Base V1 R1	√	
5695-178	SystemView Automated Operations Network (AON) V1 R1		√
5735-RC3	TCAM V2 R4	√	
5665-314	TCAM V3 R1	√	
5655-HAL	TCP/IP V3 R1		√
5798-DYE	The Information Facility (TIF) V2 R1 M3	√	
5688-121	TPNS V3 R4	√	
5752-VS2	TIOC V1 R1	√	
5688-139	Target System Control Facility(TSCF) V1 R1	√	
5685-025	TSO/E V2 R4		√
5685-025	TSO/E V2 R5	√	
5734-UT1	TSO Data Utilities	√	
5648-078	VisualGen Host Services V1 R1 (referred to previously as CSP)	√	
5648-109	VisualLift for MVS, VSE & VM V1 R1 M1	√	
5748-AP1	VS APL V1 R4	√	
5748-XX1	VS BASIC	√	
5688-035	X.25 NPSI V3 R7	√	

Notes:

* Support is available through the service stream.

[1] AOC at present contains only partial Year2000 support. Full support will be available by year end 1996.

PROLOG 5706-236 was withdrawn from service 6/30/96. See announcement letter 995-133 dated November 28, 1995.

TPF

Figure A-2. TPF Platform Products

Program Product Number	Program Name (Version Release)	Year2000 Ready	By Year End 1996
5748-T14	TPF 4.1		√
5695-068	TPF/MVS 2.1.3	√	
5706-196	TPF/DF 1.1.3		√
5695-067	EOCF/2 1.0 with CSD2		√

VSE/ESA

Figure A-3 (Page 1 of 2). VSE Platform Products

Program Product Number	Program Name (Version/Release)	Year2000 Ready	By Year End 1996
5750-ACD	VSE/ESA (Package) 1.4		√
5690-VSE	VSE/ESA (Package) 2.1		√
5686-032	VSE/Advanced Functions 5.2		√
5686-066	VSE Central Functions 6.1		√
5686-034	VSE/FAST COPY 2.2		√
5686-036	VSE/ICCF 3.2		√
5656-092	VSE/OLTEP 1.1		√
5686-033	VSE/POWER 5.2		√
5686-028	VSE/SP Unique Code 5.2		√
5686-037	VSE/VSAM 2.2		√
5668-854	ACF/NCP V4 R3 M1	√	
5666-322	ACF/SSP V3 R6	√	
5666-363	ACF/VTAM for VSE/ESA 3.4	√	
5686-065	ACF/VTAM for VSE/ESA 4.2	√	
5686-073	ADSTAR Distributed Storage Manager for VSE	√	
5648-113	AFP/Font Collection V1R1	√	
5686-075	ASF/VSE V3R1.0	√	
5746-RC5	BTAM/ES 1.1	√	
5686-026	CICS for VSE/ESA V2.3		√
5688-187	C/370 Compiler V2R1	√	
5688-188	C/370 Library V2R1	√	
5785-CCC	CCCA/VSE	√	
5686-011	CICSVR/VSE V1 R1		√
5798-DYE	COBOL Report Writer Precompiler		√
5686-068	IBM COBOL for VSE/ESA (COBOL/VS) 1.1		√
5748-XX9	DCF V1 R4	√	
5748-XXE	DLF V1 R3		√
5688-052	DITTO for VSE and VM 3.2		√
5688-052	DITTO for VSE 3.2 PF		√
5648-099	DITTO/ESA for VSE V1 R1		√
5746-XC4	Development Management System(DMS)/CICS/VS V1 R5 M0	√	
5746-SM3	DFSORT/VSE V3 R2		√
5746-XX1	DL/I DOS/VS 1.10		√
5686-041	VSE/DSNX 2.1	√	
5735-XXB	Emulation Program(EP) V1 R6 M1	√	
5668-890	FLSF V1 R1	√	

Figure A-3 (Page 2 of 2). VSE Platform Products

Program Product Number	Program Name (Version/Release)	Year2000 Ready	By Year End 1996
5748-FO3	VS FORTRAN Compiler and Library for VSE		√
5746-LM3	VS FORTRAN Library for VSE		√
5686-057	GDDM/VSE V3.1.1	√	
5666-328	GDDM/VSE V2.3	√	
5696-234	HLASM for MVS & VM & VSE V1 R2	√	
5747-DS2	Device Support Facility (ICKDSF) R16	√	
5747-DS1	ICKDSF Stand-alone	√	
5665-393	Inforem Base		√
5665-394	Inforem Allocation		√
5686-067	Language Environment for VSE/ESA 1.1	√	
5686-063	MERVA V3 R2	√	
5686-055	NetView for VSE/ESA V2.3	√	
5735-XX7	Network Terminal Option(NTO) V1 R10	√	
5648-077	NTuneMON V1 R1	√	
5648-089	NTuneNCP V1 R1	√	
5746-XC5	VSE/OCCF 1.4	√	
5688-191	OGL/370 V1 R1		√
5686-069	IBM PL/I for VSE/ESA 1.1		√
5688-190	PPFA/370 V1 R1	√	
5686-040	PSF/VSE V2 R2		√
5648-061	QMF/VSE V3 R1 M1	√	
5686-058	REXX/VSE 1.1	√	
5746-RG1	DOS/VS RPG II 1.3		√
5746-XXT	Screen Definition Facility (SDF)/CICS V1 R5		√
5688-103	SQL/DS 3.5		√
5648-078	VisualGen Host Services V1 R1	√	
5648-086	VisualGen Generator Option for VSE (replaced by VisualGen V2 R0)	√	
5648-109	VisualLift for MVS, VSE & VM V1 R1 M1	√	
5668-719	X.25 NPSI V2 R1	√	

Note:

5686-072 ALERT for VSE 4.9 and 5686-079 ALERT for CICS/VSE 4.9 are owned by Computer Associates. For Year 2000 information, contact Computer Associates directly.

VM

Figure A-4 (Page 1 of 2). VM Platform Products

	Program Product Number	Product Name (Version/Release)
	5648-063	ACF/NCP 7.3.0
3	5668-854	ACF/NCP V4 R3 M1
3	5664-289	ACF/SSP V3 R8
	5654-010	ACF/VTAM 4.2.0
	5688-216	AD/Cycle C/370 1.2.0
	5688-194	AD/Cycle CODE/370 V1R2
	5648-020	ADSTAR Distributed Storage Manager/VM
	5654-033	C for VM/ESA Compiler V3R1
	5635-001	CADAM Interactive Design System (and associated products)
	5627-COM	CATIA Object Manager (and associated products)
	5688-197	COBOL for MVS & VM
	5688-187	C/370 Compiler V2R1
	5688-188	C/370 Library V2R2
	5684-042	Device Support Facility (ICKDSF) R16
	5747-DS1	ICKDSF Stand-alone
	5684-112	DFSMS/VM 2.2.1
	5684-134	DFSORT/CMS V2 R1
	5748-XE4	DIRMAINT 1.5.0
	5684-113	Display Mgt System for CMS 2.1.0
	5654-029	DITTO/ESA for VM V1 R1
	5668-806	Fortran 2.6.0
	5684-168	GDDM/VM 3.1.1
	5684-007	GDDM/VM XA V2.3
	5696-234	HLASM for MVS & VM & VSE V1 R2
	5684-157	Host Mgt Facilities 1.1.1
	5684-043	ISPF 3.2.0
	5648-039	LAN File Services (LFS) 1.1.2
	5688-198	Language Environment for MVS & VM
	5684-142	LANRES/VM 1.3.0
	5684-028	NetView Access Services VM V1.3.2
	5684-001	NetView Performance Monitor/VM V1.6
	5684-011	NetView Performance Monitor/VM V2
	5756-051	NetView for VM/ESA 2.3
3	5648-077	NtuneMON V1 R2
3	5648-087	NTuneNCP V1 R2
	5684-084	OV/VM 1.3.0

Figure A-4 (Page 2 of 2). VM Platform Products

	Program Product Number	Product Name (Version/Release)
	5668-909	OS PL/I Version 2
	5688-235	PL/I for MVS and VM
	5664-310	PMF/VM Version 1.1
	5664-312	PSAF/VM Version 1.1
	5684-141	PSF/VM Version 2.1.1
	5684-100	PVM 2.1.1
	5706-255	QMF V3 R1 M1
3	5740-XXH	RACF V1 R1 M0
	5695-013	REXX Compiler V1 R3 M0
	5684-096	RSCS 3.2.0
	5664-307	Screen Definition Facility (SDF) II VM V1 R3
	5684-143	SearchManager/370 for VM V1 R2
	5688-103	SQL/DS 3.5
	5735-FAL	TCP/IP 2.3.0
	5688-121	TPNS
	5648-109	VisualLift for MVS, VSE & VM V1 R1 M1
	5654-030	VM/ESA Version 2
	5686-037	VSE/VSAM for VM 6.1.0
3	5668-719	X.25 NPSI V2 R1

OS/400

Figure A-5 (Page 1 of 3). OS/400 Platform Products

Program Product Number	Product Name (Version/Release)	Year2000 Ready	By Year End 1996
5716-SV1	ADSTAR Distributed Storage Manager for OS/400 V3 R6	√	
5763-SV1	ADSTAR Distributed Storage Manager for OS/400 V3 R1	√	
5733-197	ADSTAR Distributed Storage Manager for OS/400 V2 R3	√	
5716-AP1	Adv DBCS Printer Support for OS/400 V3 R6	√	
5763-AP1	Adv DBCS Printer Support/400 V3 R1	√	
5716-AF1	Adv Function Printing Utils for OS/400 V3 R6	√	
5716-FNT	Adv Function Printing Fonts for OS/400 V3 R6	√	
5716-FN1	Adv Function Printing DBCS Fonts for OS/400 V3 R6	√	
5763-AF1	Adv Function Printing Utils/400 V3 R1	√	
5763-FNT	Adv Function Printing Fonts/400 V3 R1	√	
5763-FN1	Adv Function Printing DBCS Fonts/400 V3 R1	√	
5716-PW1	Application Development ToolSet for OS/400 V3		√
5763-PW1	Application Development ToolSet/400 V3 R2	√	
5716-CL1	Appl Dev ToolSet Client Svr for OS/400 V3 R6	√	
5763-CL1	Appl Dev ToolSet Client Svr/400 V3 R1	√	
5716-PD1	Application Program Driver for OS/400 V3 R6	√	
5763-PD1	Application Program Driver/400 V3 R1	√	
5763-DS1	AS/400 Business Graphics Util V3 R2	√	
5763-DB1	AS/400 System/38 Utilities V3 R1	√	
5763-VR1	AS/400 VRPG Client/2 V3 R1	√	
5716-BR1	Backup Recovery & Media Svcs for OS/400 V3 R6	√	
5763-BR1	Backup Recovery and Media Services/400 V3 R1	√	
5716-DS1	Business Graphics Utility for OS/400 V3		√
5716-CP2	CallPath for OS/400 V3 R6	√	
5763-CP2	CallPath/400 V3 R1	√	
5716-DFH	CICS for OS/400 V3 R6	√	
5716-XA1	Client Access for OS/400 Version 3 R6	√	
5716-XC1	Client Access Windows+ Client for OS/400 Version 3 R6	√	
5716-XL1	Client Access DOS Client for OS/400 Version 3 R6	√	
5716-XM1	Client Access ToolKit for OS/400 Version 3 R6	√	
5716-XB1	Client Access DOS with ExtMem Client for OS/400 V3 R6	√	
5716-XF1	Client Access OS/2 Client for OS/400 V3 R6	√	
5716-XG1	Client Access Opt OS/2 Client for OS/400 V3 R6	√	
5716-US1	Client Access Ultimedia Tools for OS/400 V3 R6	√	
5763-US1	Client Access/400 Ultimedia Tools V3 R1	√	
5763-XA1	Client Access/400 Family Version 3 R1	√	

Figure A-5 (Page 2 of 3). OS/400 Platform Products

Program Product Number	Product Name (Version/Release)	Year2000 Ready	By Year End 1996
5763-XC1	Client Access/400 for Windows 3.1 Version 3 R1	√	
5763-XL1	Client Access/400 for DOS Version 3 R1	√	
5763-XM1	Client Access/400 ToolKit Version 3 R1	√	
5763-XB1	Client Access/400 for DOS with Ext Mem V3 R1	√	
5763-XF1	Client Access/400 for OS/2 Client V3 R1	√	
5763-XG1	Client Access/400 Optimized for OS/2 V3 R1	√	
5716-CM1	Communications Utils for OS/400 V3		√
5763-CM1	Communications Utils/400 V3 R2	√	
5763-CD1	CoOperative Development Environment/400 V3 R1	√	
5716-CR1	Cryptographic Support for OS/400 V3 R6	√	
5763-CR1	Cryptographic Support/400 V3 R1	√	
5716-DP1	DataPropagator Rel Capture&Apply for OS/400 V3 R6	√	
5763-DP1	DataPropagator Rel Capture&Apply/400 V3 R1	√	
5716-ST1	DB2 Query Manager and SQL Development Kit for OS/400 V3 R6	√	
5763-ST1	DB2/400 Query Manager and SQL Development Kit V3 R1	√	
5798-TAY	Facsimile Support for OS/400 V3 R6	√	
5798-RZT	Facsimile Support/400 V3 R1	√	
5716-CX2	Integrated Language Environment C for OS/400 V3 R6	√	
5763-CX2	Integrated Language Environment C/400 V3 R1	√	
5716-CB1	Integrated Language Env COBOL for OS/400 V3 R6	√	
5716-RG1	Integrated Language Env RPG for OS/400 V3 R6	√	
5763-CB1	Integrated Language Environment COBOL/400 V3 R1	√	
5763-RG1	Integrated Language Env RPG/400 V3 R1	√	
5716-JS1	Job Scheduler for OS/400 V3 R6	√	
5716-DCT	Language Dictionaries for OS/400 V3 R6	√	
5763-DCT	Language Dictionaries/400 V3 R1	√	
5763-MW1	ManageWare/400 V3 R6	√	
5763-MQ1	MQSeries for OS/400 V3 R1	√	
5798-TBC	Mobile Network Access PagerPac for OS/400 V3 R1	√	
5798-TBD	Mobile Network Access RadioPac for OS/400 V3 R1	√	
5733-196	NetView FTP for OS/400 V3 R1	√	
5798-TBA	Neural Network Util for OS/400 V3 R6	√	
5798-RZK	Neural Network Util/400 V3 R1	√	
5798-TAQ	OfficeVision JustMail for OS/400 V3 R6	√	
5798-RZJ	OfficeVision JustMail/400 V3 R2	√	
5716-WP1	OfficeVision for OS/400 V3 R6	√	
5763-WP1	OfficeVision/400 V3 R2	√	
5716-SS1	Operating System/400 V3		√

Figure A-5 (Page 3 of 3). OS/400 Platform Products

Program Product Number	Product Name (Version/Release)	Year2000 Ready	By Year End 1996
5763-SS1	Operating System/400 V3 R2	√	
5716-OS1	OSI Comm Subsystem for OS/400 V3 R6	√	
5763-OS1	OSI Comm Subsystem/400 V3 R1	√	
5716-FS1	OSI File Svcs for OS/400 V3 R6	√	
5763-FS1	OSI File Svcs/400 V3 R1	√	
5716-MS1	OSI Message Svcs for OS/400 V3 R6	√	
5763-MS1	OSI Message Svcs/400 V3 R1	√	
5763-PS1	Pascal V3 R1Svcs/400 V3 R1	√	
5798-RYZ	PagerPac for the AS/400 V3 R1	√	
5716-PM1	Performance Management for OS/400 V3		√
5716-PT1	Performance Tools for OS/400 V3		√
5763-PT1	Performance Tools/400 V3 R2	√	
5763-PL1	PL/I V3 R1	√	
5716-CF1	Point-Of-Sale Comm Util for OS/400 V3 R6	√	
5763-CF1	Point-Of-Sale Comm Util/400 V3 R1	√	
5716-QU1	Query for OS/400 V3 R1	√	
5763-QU1	Query/400 V3 R1	√	
5798-RYY	RadioPac for the AS/400 V3 R1	√	
5733-218	Report/Data Archive&Retrieval Sys for OS/40V3 R1	√	
5716-STK	SOMobjects Developer Toolkit for OS/400 V3 R6	√	
5716-DB1	System/38 Utilities for OS/400 V3 R6	√	
5716-MG1	SystemView Managed Sys Serv for OS/400 V3 R6	√	
5763-MG1	SystemView Managed System Services/400 V3 R1	√	
5763-MW1	SystemView ManageWare/400 V3 R1	√	
5716-ES1	SystemView OMEGAMON Svcs for OS/400 V3 R6	√	
5763-ES1	SystemView OMEGAMON Svcs/400 V3 R1	√	
5716-SM1	SystemView System Manager for OS/400 V3 R6	√	
5763-SM1	SystemView System Manager/400 V3 R1	√	
5716-TC1	TCP/IP Connectivity Utils for OS/400 V3 R6	√	
5763-TC1	TCP/IP Connectivity Utils/400 V3 R2	√	
5716-UB1	Ultimedia Business Conf'g for OS/400 V3 R6	√	
5763-UB1	Ultimedia Business Conf'g/400 V3 R1	√	
5716-VG1	VisualGen Host Svcs for OS/400 V3 R6	√	
5763-VG1	VisualGen Host Svcs/400 V3 R1	√	
5716-CX4	VisualAge C++ for OS/400 V3		√

AIX

Figure A-6 (Page 1 of 4). AIX Platform Products

Program Product Number	Product Name (Version/Release)
5765-564	ADSTAR Distributed Storage Manager V2 R1
5697-078	ADSTAR Distributed Storage Manager V1 R2.1
5765-268	AIX Asynch.Term.Svr.Accelerator/6000
5765-266	AIX CallPath Server/6000
5765-117	AIX DCE Base Services/6000 V1
5765-119	AIX DCE Cell Dir Server/6000 V1
5765-121	AIX DCE Enhanced Dist.File Sys./6000 V1
5765-120	AIX DCE Global Dir.Server/6000 V1
5765-259	AIX DCE Global Dir.Client/6000 V1
5765-118	AIX DCE Security Server/6000 V1
5765-232	AIX DCE Threads/6000 V1
5765-001	AIX DirectTalk/6000
5696-902	AIX Distributed SMIT V2.2 for AIX
5765-042	AIX EngSci Subroutine Lib./6000 V2 R2
5696-708	AIX File Storage Facility (FSF)
5756-030	AIX for RISC System/6000 Version 3.2.5
5696-923	AIX HACMP/6000 V3.1
5696-933	AIX HACMP/6000 V4.1 for AIX V4
5765-551	AIX HIPPI/6000 V3 R2 M3
5696-108	AIX InfoCrafter/6000 V1.1
5696-893	AIX InfoCrafter V2.1
5621-107	AIX NetView Service Point 1.2
5621-013	AIX OSL/6000 Version 1.2
5765-296	AIX Parallel System Support Programs V1.2
5765-529	AIX Parallel System Support Programs for AIX V2.1
5696-899	AIX Performance Aide/6000 V2.1
5696-900	AIX Performance Toolbox/6000 V2.1
5765-349	AIX /SMARTsort for Workstation
5765-261	AIX SNA Gateway/6000 V2.2
5696-906	AIX Ultimedia Services/6000 V2.1.1
5765-393	AIX Version 4.1.3
5696-868	AIX X.25 V1.1
5765-011	AIX X-Windows 3270 Emulator/6000
5601-248	AIX XL FORTRAN/6000 Version 2.3
5765-018	AIX XL Fortran Compiler/6000
5765-019	AIX XL Fortran Runtime Env/6000

Figure A-6 (Page 2 of 4). AIX Platform Products

Program Product Number	Product Name (Version/Release)
5765-176	AIX XL Fortran for AIX V3.2
5601-251	AIX XL Pascal Runtime Env./6000 (service withdrawn April 1995)
5601-254	AIX XL Pascal Compiler/6000 Version 1.1.2 (service withdrawn April 1995)
5765-245	AIX XL Pascal Compiler/6000 Version 2 (replaces 5601-251 and 5601-254 above)
5601-260	AIX 3270 Host Conn/6000 Version 1.3
5765-249	AIX 5080 Emulation Program/6000
5765-398	AIXlink V2.1.1
5696-926	AIXlink/X.25 V1.1
5696-904	AIXwindows Display Postscript V1.1
5601-257	AIXwindows Environment/6000 V1.2.5
5756-027	AIXwindows Interface Composer/6000 (AIC) V1.2
5765-423	C for AIX
5765-421	C Set ++ for AIX
5626-COM	CATIA Object Manager V4 R1 M6 (and associated products)
5765-148	CICS for AIX V1 R2
5765-152	CICS Client for AIX V1 R2
5765-427	CICS System Manager V1.1 for AIX
5801-AAR	COBOL Set V1.1.0 for AIX
5765-561	CommonPoint Application System for AIX V1.1
5765-562	CommonPoint Application Development Toolkit V1.1
5765-652	Communications Server for AIX V4
5765-022	Consumer Transaction Definition/6000
5767-023	Consumer Transaction Runtime/6000
5765-418	Data Encryption Std Lib Routine V1.1
5765-642	Database Server V4
5765-256	DataHub Support/6000 V3.2 (service withdrawn October 1995)
5801-AAR	DataHub for UNIX Operating Systems (replaces 5765-256 above)
5871-AAA	DB2 for AIX V2.1
5765-459	DB2 SDK/6000 V1 R2
5765-453	DB2 SDK for AIX V2 R1
5765-464	DB2/6000 V1 R2
5871-AAA	DDCS for AIX V2.3
5696-239	Encina Monitor for AIX/6000
5696-238	Encina PTP Exec for AIX/6000
5696-347	Encina PTP Gateway for AIX/6000
5696-240	Encina Server for AIX/6000
5696-237	Encina Structured File Svr for AIX/6000
5765-527	Extended Systems Administration Feature (SystemView)
5696-923	High Availability Cluster Multi-Processing V3 R1 M1

Figure A-6 (Page 3 of 4). AIX Platform Products

Program Product Number	Product Name (Version/Release)
5765-482	LANDP/6000 V2.1
5801-AAR	FlowMark for AIX V2 R2 (replaces 5765-270 FlowMark for AIX V1 R1)
5765-026	geoGPG/6000
5696-919	Hypertext Information Base Libraries V1.1
5696-898	InfoExplorer License Extension V1.1
5765-223	InterMix for AIX Version 1.2
33H4191	Internet Connection Secure Server for AIX
33H4190	Internet Connection Server for AIX
33H4290	Internet Connection Secured Network Gateway V2.1
5765-527	Job Scheduler Feature (SystemView)
5765-264	LAN Management Utilities/6000 V1.1.3
5765-145	LoadLeveler Version 1.2.1
5765-449	MERVA AIX V1 R1 (supports the definitions in the SWIFT User Handbook)
5765-115	MQSeries for AIX Version 2.1.0 (service to be withdrawn 31 Dec 1996. Announced 28 Mar 1995, letter number 295-144)
5765-550	NetBIOS & IPX/SPX Support/6000 V2
5871-BBB	NETSP Secured Network Gateway 1.2 (feature 6362)
5765-196	NetView Distribution Manager for AIX V1.1 (service to be withdrawn Sep 1996. Announced 5 Dec 1995, letter number 995-140)
5765-214	NetView Distribution Management Agent for AIX V1.1 (service to be withdrawn Sep 1996. Announced 5 Dec 1995, letter number 995-140)
5696-731	NetView for AIX 3.1
5622-242	NetView FTP Client V1.1 for AIX
5765-435	NetView FTP Server V1.1 for AIX
5696-236	Netware for AIX/6000
5696-939	OpenGL & GL 3.2 Version 4.1.3
5765-352	OpenMail for AIX B.02 (withdrawn from service Jan 1996. Announced 10 Oct 1995, letter number 995-090)
5765-543	Parallel Environment for AIX V2.1
5765-422	Parallel ESSL for AIX V1.1
5765-297	Parallel I/O File System V1.1
5765-392	Parallel OSL (OSLp) V1.1.1
5765-469	Parallel Visual Explorer
5765-527	Performance Reporter Feature (SystemView)
5696-907	Pex & PHIGS Version 4.1.3
5765-193	ProductManager Application Svcs Mgr (and associated products for DB2)
5765-440	ProductManager Application Svcs Mgr (and associated products for Oracle)
5765-605	ProductManager Version 3 Release 1 for AIX
5756-094	PROFESSIONAL CADAM Interactive Design (and associated products)
5765-505	PSF for AIX Version 2.1

Figure A-6 (Page 4 of 4). AIX Platform Products	
Program Product Number	**Product Name (Version/Release)**
5765-246	PVMe for AIX V1 R3
5765-544	PVMe for AIX V2.1
5696-038	Realtime Interface Co-Processor AIX Support V1.1.1
5765-444	Recoverable Virtual Shared Disk V1.1
5765-292	RMONitor V1.1 for AIX
5765-343	Router and Bridge Manager/6000 V2.3
5765-233	SNA Manager/6000 V1.1
5765-247	SNA Server/6000 V2.2
5765-410	Systems Monitor for AIX V2.2
5765-527	SystemView for AIX
5697-213	The Multimedia Server for AIX
5765-265	Trouble Ticket V3.2
5765-400	UIM/X V2.8
5696-398	UniTree for AIX/6000 (service withdrawn Dec 1995)
5765-315	WABI V2.0

Notes:

- 5706-294, AIX ADA Run-Time Env/6000 Composer and 5706-291, AIX ADA/6000 Version 1.2.3 were withdrawn from service Nov 1994 and have since been transferred to OC Systems, Inc.
- 5756-085, AIX OSI Messaging and Filing/6000 was withdrawn from service Dec 1994 and has been replaced by ObjectStore by Object Design, Inc.

OS/2

Figure A-7 (Page 1 of 2). OS/2 Platform Products

Program Product Number	**Product Name (Version/Release)**
83G8102	OS/2 Warp, Version 3 CD-ROM
83G8100	OS/2 Warp, Version 3 3.5-inch diskette
83G8701	OS/2 Warp with WIN-OS2, Version 3 CD-ROM
83G8700	OS/2 Warp with WIN-OS2, Version 3 3.5-inch diskette
10H9800	OS/2 Warp Connect, Version 3
10H9810	OS/2 Warp Connect with WIN-OS2, Version 3
52G8474	OS/2 LAN Server Entry Version 4 CD-ROM
52G8468	OS/2 LAN Server Entry Version 4 3.5-inch diskette
52G8476	OS/2 LAN Server Advanced Version 4 CD-ROM
52G8475	OS/2 LAN Server Advanced Version 4 3.5-inch diskette
96F8690	IBM DCE SDK for OS/2 and Windows Version 1.0
96F8691	IBM DCE Client for OS/2 Version 1.0
83G9217	IBM PC DOS Version 7 3.5-inch diskette
83G9219	IBM PC DOS Version 7 5.25-inch diskette
89G1342	ADSTAR Distributed Storage Manager/2
87G7776	AnyNet/2 (and associated products)
03H3624	CallPath CallCoordinator/2*
03H3625	CallPath CallCoordinator/2 Server
03H3626	CallPath CallCoordinator/2 Archive
17H78xx	CallPath DirectTalk/2* V2.01
5622-275	CallPath Server/2
5621-159	CallPath SwitchServer/2*
5648-036	CICS for OS/2
79G0257	Communications Manager/2 V1.1 CD-ROM
79G0258	Communications Manager/2 V1.1 3.5-inch diskette
3 5688-205	Cross System Product (CSP)/2AD V1 R1
3 5688-195	Cross System Product (CSP)/2RS V1 R1
41H2112	DB2 for OS/2 (V2.1 and later) Single-User, CD-ROM
41H2113	DB2 for OS/2 (V2.1 and later) Single-User, 3.5
41H2114	DB2 for OS/2 (V2.1 and later) Server, CD-ROM
41H2115	DB2 for OS/2 (V2.1 and later) Server, 3.5
41H2120	DDCS for OS/2 V2.3 Single-User, 3.5
41H2121	DDCS for OS/2 V2.3 Multi-User Gateway, 3.5
41H2126	DB2 SDK for OS/2 (V2.1 and later), 3.5
29H1040	Distributed Console Access Facility V1.3
33H4189	Internet Connection Server for OS/2 Warp

Figure A-7 (Page 2 of 2). OS/2 Platform Products

Program Product Number	**Product Name (Version/Release)**
33H4207	Internet Connection Secure Server for OS/2 Warp
24H3919	LANDP/2 V3
20H1677	License Use Management Application Developer's Toolkit for OS/2
41H4495	License Use Management Runtime for OS/2
76G7991	LAN NetView Management Utility for OS/2 V1.1
5622-122	MERVA/2 V3 R2 (supports the definitions in the SWIFT User Handbook)
17H7937	MQSeries for OS/2 V2.0
72G6198	Network Door/2 V1
28H3798	NetView Distribution Manager Easy Preparer for OS/2 V1
17H7720	NetView Distribution Manager/2 V2.1 Entry
17H7721	NetView Distribution Manager/2 V2.1 Extended
10H7889	NetView Distribution Manager/2 V2.1 Remote Administrator
79G9845	NetView Distribution Management Agent/2 V1
16H9589	NetView for OS/2 V2.0
16H9610	NetView for OS/2 V2.1
89G1381	NetView File Transfer Program Client/2
89G1384	NetView File Transfer Program Server/2
5622-009	Network Design and Analysis/2 V1
53G3997	Person-to-Person for OS/2
85G8697	Personal Communications/3270 V4.0
85G8661	Personal Communications AS/400 V4.0 for OS/2
85G8805	Personal Communications AS/400 and 3270 V4.0 for OS/2
5622-551	PSF for OS/2 Version 2.00
27H8163	SMARTsort
96F8379	System Performance Monitor/2 Version 2
50H0794	SystemView for OS/2 V1 R1
89H1653	SystemView Server V4 R0
5802-AAR	TCP/IP for OS/2
31H3744	TeamConnection for OS/2
17H7495	VisualAge V2 for OS/2
5622-679	VisualAge C++ for OS/2
3 28H2177	VisualAge for COBOL for OS/2 V1 R1
31H3678	VisualGen V2 for OS/2
5801-AAR	Workgroup Services for OS/2 and AIX V2

Lotus Products

Figure A-8 (Page 1 of 2). Lotus Products
Program Product Name (Version/Release)
Approach 3.0
Approach 96
cc:Mail client (all platforms) Release 6.0
Freelance Graphics (all versions and platforms)
Lotus Access Unit/cc:Mail for OS/2
Lotus Access Unit/Notes for OS/2
Lotus Directory Synchronization/Banyan
Lotus Directory Synchronization/Callup
Lotus Directory Synchronization/cc:Mail
Lotus Directory Synchronization/Central
Lotus Directory Synchronization/DEC
Lotus Directory Synchronization/EAB
Lotus Directory Synchronization/EMC2
Lotus Directory Synchronization/LMS
Lotus Directory Synchronization/Lotus Messaging Switch
Lotus Directory Synchronization/MEMO
Lotus Directory Synchronization/Microsoft Mail for PC Networks
Lotus Directory Synchronization/MS Mail for PCs
Lotus Directory Synchronization/Notes
Lotus Directory Synchronization/PROFS
Lotus Directory Synchronization/SYSM
Lotus Directory Synchronization/Toolkit
Lotus Directory Synchronization/Wang OFFICE
Lotus Gateway for CA-eMail+
Lotus Gateway for cc:Mail
Lotus Gateway for GroupWise
Lotus Gateway for Message Router
Lotus Gateway for MHS
Lotus Gateway for MS Mail for PC Networks
Lotus Gateway for MS Mail for PCs
Lotus Gateway for Notes
Lotus Gateway for VMSmail
Lotus Gateway for Wang OFFICE
Lotus Mail Monitor
Lotus mailFax
Lotus Messaging Switch
Lotus Messaging Switch Admin-By-Mail
Lotus Messaging Switch Application Toolkit/MVS

Figure A-8 (Page 2 of 2). Lotus Products
Program Product Name (Version/Release)
Lotus Messaging Switch Query-By-Mail
Lotus Messaging Switch Open Addressing Service
Lotus Pages
Lotus 1-2-3 (all current versions; OS/2 R2.1, DOS R4, Windows R5)
Notes Client Release 3 and Release 4 (all platforms)
Organizer 2.0
Remote Manager for Lotus Access Unit/cc:Mail for OS/2
Remote Manager for Lotus Access Unit/Notes for OS/2
ScreenCAM 2.1
SmartCenter for Windows
SmartCenter for OS/2
SmartCenter 96 for Win 95
Soft-Switch Central/MVS & VM
Soft-Switch Central Admin-By-Mail & Query-By-Mail
Soft-Switch Central Application Toolkit/MVS & VM
Soft-Switch Central Fax Gateway
Soft-Switch Central Open Addressing Service
Soft-Switch MAILbridge Server/DEC
Soft-Switch MAILbridge Server/HPDesk
Soft-Switch MAILbridge Server/MCI
Soft-Switch MAILbridge Server/MHS
Soft-Switch MAILbridge Server/SYSM
Soft-Switch MAILbridge Server/Wang OFFICE
Soft-Switch SNADS Gateway/Banyan
Soft-Switch SNADS Gateway/MS Mail for Appletalk Networks
Word Pro (all platforms)

3 Tivoli Products

3 All Tivoli products at all release levels are Year2000 ready today.

Hardware

The hardware timers on the IBM S/390, AS/400, RISC/6000, and PowerPC machines are not sensitive to change of century. On a personal computer, the situation is more complicated.

IBM Personal Computer (PCs) - Hardware Timer Setting

In original PCs and XTs, the hardware did not have an internal real time clock. Therefore, the time and date had to be manually set each time the machine was booted. Beginning with the AT, personal computer systems maintain two dates; one date in the hardware component, and one in the operating system software (for example, DOS or Windows). The date in the operating system software is created by converting the date in the hardware component.

All IBM machines since the AT have a century byte in the hardware; it is used by the basic input/output system (BIOS) to determine the century. However, that century byte does not typically roll over at the end of 1999. You will have to reset it manually using either the Setup program (provided either on floppy disk, hard disk system partition, or via a BIOS program) or use the DATE command found in either DOS 3.3 or later or OS/2. This is the same procedure used to re-enter the date when the battery (which supports the timer) looses its charge and is replaced.

All new models of IBM PCs shipped in 1996 will automatically update the century byte.

For current IBM PCs and earlier models, some may require you use a command or a CMOS update utility to change centuries, however, some may not require any change.

The following tables list IBM PCs and PC Servers and indicate whether each model has manual or automatic ‘century-byte-set’ support.

Desktop PC Systems

Figure A-9 (Page 1 of 7). Desktop PC Systems - Internal Clock Setting

System	Model	Date Available	Century Byte Set Manually	Century Byte Set Automatically
PC	all		See Note	
XT	all		See Note	
XT286	all		√	
AT	all		√	
PS/1	all		√	
PS/2	all		√	
PS/55	all		√	
APTIVA	all before 1996		√	
2144-14P			√	
2144-16P			√	
2144-22P			√	

Figure A-9 (Page 2 of 7). Desktop PC Systems - Internal Clock Setting				
System	**Model**	**Date Available**	**Century Byte Set Manually**	**Century Byte Set Automatically**
2144-24P			√	
2144-25P			√	
2144-27P			√	
2144-29P			√	
2144-66P			√	
2144-67P			√	
2144-82P			√	
2144-83P			√	
2144-86P			√	
2144-22F			√	
2144-24F			√	
2144-25F			√	
2144-27F			√	
2168-26P			√	
2168-62P			√	
2168-26F			√	
2168-62F			√	
2144-743			√	
2144-744			√	
2144-745			√	
2144-754			√	
2144-766			√	
2144-767			√	
2144-768			√	
2144-782			√	
2144-784			√	
2144-785			√	
2144-786			√	
2144-787			√	
2144-788			√	
2144-797			√	
2144-798			√	
2144-843			√	
2144-853			√	
2144-854			√	
2144-855			√	
2144-856			√	
2144-866			√	
2144-887			√	

Figure A-9 (Page 3 of 7). Desktop PC Systems - Internal Clock Setting				
System	**Model**	**Date Available**	**Century Byte Set Manually**	**Century Byte Set Automatically**
2144-888			√	
2144-8K2			√	
2144-S82			√	
2144-S86			√	
2144-T82			√	
2168-755			√	
2168-756			√	
2168-789			√	
2168-792			√	
2168-857			√	
2168-889			√	
2144-H55			√	
2144-L55			√	
2144-H66			√	
2144-L66			√	
2144-K66			√	
2144-H77			√	
2144-L77			√	
2144-K77			√	
2144-KB1			√	
2144-KB2			√	
2144-H78			√	
2144-L78			√	
2144-K78			√	
2168-H89			√	
2168-L89			√	
2168-K89			√	
2144-H92			√	
2144-L92			√	
2144-K92			√	
2144-KB3			√	
2144-H67			√	
2144-L67			√	
2144-K67			√	
2144-H80			√	
2144-L80			√	
2144-K80			√	
2144-H81			√	
2144-L81			√	

Figure A-9 (Page 4 of 7). Desktop PC Systems - Internal Clock Setting				
System	**Model**	**Date Available**	**Century Byte Set Manually**	**Century Byte Set Automatically**
2144-K81			√	
2144-KB4			√	
2144-MB4			√	
2144-KB5			√	
2144-MB5			√	
2144-H93			√	
2144-L93			√	
2144-K93			√	
2144-KB6			√	
2168-H90			√	
2168-L90			√	
2168-K90			√	
2144-X65			√	
2144-X70			√	
2144-X88			√	
2144-X79			√	
2144-X89			√	
2144-X90			√	
2144-Y65			√	
2144-Y70			√	
2144-Y78			√	
2144-Y84			√	
2144-Y88			√	
2144-Y79			√	
2144-Y89			√	
2144-Y90			√	
2144-26J			√	
2144-68J			√	
2144-27J			√	
2144-28J			√	
2168-62J			√	
2168-63J			√	
2168-64J			√	
2168-65J			√	
2144-P30		5/95	√	
2144-S15		5/95	√	
2144-LP0		6/95	√	
2144-LP1		6/95	√	
2144-LS1		8/95	√	

Figure A-9 (Page 5 of 7). Desktop PC Systems - Internal Clock Setting				
System	**Model**	**Date Available**	**Century Byte Set Manually**	**Century Byte Set Automatically**
2144-KP1		8/95	√	
2144-LP2		6/95	√	
2144-LS2		8/95	√	
2144-KP2		8/95	√	
2144-XP1		6/95	√	
2144-XP2		6/95	√	
2144-YP1		6/95	√	
2144-YP2		6/95	√	
2144-70J		7/95	√	
2144-71J		7/95	√	
2144-KU0		8/95	√	
2144-KU1		8/95	√	
2144-M30		8/95	√	
2144-M40		8/95	√	
2144-900		8/95	√	
2144-910		8/95	√	
2144-911		8/95	√	
2144-930		8/95	√	
2144-LB0		8/95	√	
2144-KB0		8/95	√	
APTIVA	all after 1995			√
2144-C30		2H95	√	
2144-M31		2H95	√	
2144-C31		2H95	√	
2144-M35		2H95	√	
2168-M40		2H95	√	
2168-C40		2H95	√	
2168-M41		2H95	√	
2168-C41		2H95	√	
2144-M50		2H95	√	
2144-M51		2H95	√	
2144-C51		2H95	√	
2168-M52		2H95	√	
2168-M53		2H95	√	
2144-C53		2H95	√	
2168-M54		2H95	√	
2168-M55		2H95	√	
2168-M56		2H95	√	

System	Model	Date Available	Century Byte Set Manually	Century Byte Set Automatically
2168-M57		2H95	√	
2168-M58		2H95	√	
2168-M60		2H95	√	
2168-M61		2H95	√	
2168-M62		2H95	√	
2168-M63		2H95	√	
2168-M70		2H95	√	
2168-M71		2H95	√	
2168-M72		2H95	√	
2168-M91		2H95	√	
2144-91W		2H95	√	
2144-914		2H95	√	
2144-921		2H95	√	
2144-92W		2H95	√	
2168-931		2H95	√	
2168-93W		2H95	√	
2168-932		2H95	√	
2168-9W2		2H95	√	
2168-934		2H95	√	
2168-935		2H95	√	
2144-937		2H95	√	
2144-941		2H95	√	
2144-94W		2H95	√	
2168-951		2H95	√	
2168-95W		2H95	√	
2168-964		2H95	√	
2168-96W		2H95	√	
2144-LL1		2H95	√	
2144-L10		2H95	√	
2144-K10		2H95	√	
2144-LL2		2H95	√	
2144-L20		2H95	√	
2144-K20		2H95	√	
2144-KU6		2H95	√	
2144-LL3		2H95	√	
2144-L30		2H95	√	
2144-K30		2H95	√	
2144-LL4		2H95	√	
2144-L40		2H95	√	

Figure A-9 (Page 6 of 7). Desktop PC Systems - Internal Clock Setting

Figure A-9 (Page 7 of 7). Desktop PC Systems - Internal Clock Setting				
System	**Model**	**Date Available**	**Century Byte Set Manually**	**Century Byte Set Automatically**
2144-K40		2H95	√	
2144-KU4		2H95	√	
2168-LL5		2H95	√	
2168-L50		2H95	√	
2168-K50		2H95	√	
2168-LL6		2H95	√	
2168-L60		2H95	√	
2168-K60		2H95	√	
2144-X10		2H95	√	
2144-Y10		2H95	√	
2144-Z10		2H95	√	
2144-X20		2H95	√	
2144-Y20		2H95	√	
2144-Z20		2H95	√	
2144-W30		2H95	√	
2144-X30		2H95	√	
2144-Y30		2H95	√	
2144-Z30		2H95	√	
2168-X40		2H95	√	
2168-Y40		2H95	√	
2168-Z40		2H95	√	
2144-N30		2H95	√	
2144-N40		2H95	√	
2144-N41		2H95	√	
2168-N50		2H95	√	
2168-N51		2H95	√	
2168-N60		2H95	√	
2168-N61		2H95	√	
2168-N71		2H95	√	
Note: No internal clock. You must enter the date when you re-boot the system.				

Commercial PC Desktop Systems

All models in Figure A-10 on page A-32 update the century byte automatically after you install the flash BIOS available from the Bulletin Board Service.

Figure A-10. Commercial Desktop PC Systems - Internal Clock Setting					
System	**Model/ Type**	**Update Level**	**Date Avail- able**	**Century Byte Set Manually**	**Century Byte Set Automatically**
6381		L8JT45A and higher	1996		√
6382/S		L6JT69A and higher	1996		√
6384/D		L6JT69A and higher	1996		√
6387/T		L6JT69A and higher	1996		√
6384 P60/D		1.00.06 and higher	1996		√
6472		LDJT71A and higher	1996		√
6482		LDJT71A and higher	1996		√
6484		LDJT71A and higher	1996		√
6492		LDJT71A and higher	1996		√
6494		LDJT71A and higher	1996		√
6571		LEJT62A and higher	1996		√
6573		LEJT62A and higher	1996		√
6575		N1JT63A and higher	1996		√
6581		LEJT62A and higher	1996		√
6583		LEJT62A and higher	1996		√
6585		N1JT63A and higher	1996		√
6875		N1JT63A and higher	1996		√
6876		N2JT38A and higher	1996		√
6885		N1JT63A and higher	1996		√
6886		N2JT38A and higher	1996		√

Figure A-11. Commercial Desktop PC Systems - Internal Clock Setting				
System	**Model/ Type**	**Date Available**	**Century Byte Set Manually**	**Century Byte Set Automatically**
PS/V	2405/2410	10/92	√	

PCs

Figure A-11. Commercial Desktop PC Systems - Internal Clock Setting				
System	**Model/ Type**	**Date Available**	**Century Byte Set Manually**	**Century Byte Set Automatically**
PS/V	2405	5/93	√	
PS/V	2405/2410	9/93	√	
PS/V Vision	2408	11/93	√	
PS/V Entry	2406	2/94	√	
PS/V MASTER	2411	5/94	√	
PC 750	6885	1/95	√	
PC 330	6571	3/95	√	
PC 720 ISA	6869	5/95	√	
OMNI-MC	6860-J0x/ JZG	01/95	√	
OMNI-MC	6860-J4G	01/95	√	
OMNI-MC(CR)	6860-Jxx	08/95	√	
5510	all		√	
5520	all		√	
5530	all		√	
5535	all		√	
5540	all		√	
5545	all		√	
5550	all		√	
5560	all		√	
5570	all		√	
5580	all		√	
6576		1996		√
6586		1996		√
6598	CXJ	12/95		√
6598	CXK	12/95		√
6598	CXX	12/95		√
6877				√
6887				√
8595	all		√	
9557	all		√	
9585	all		√	
9595	all		√	
550x	all		√	

Mobile PC Systems

Figure A-12 (Page 1 of 2). Mobil PC Systems - Internal Clock Setting

System	Model/ Type	Date Available	Century Byte Set Manually	Century Byte Set Automatically
700/C	9552	10/92	√	
700T	2521		√	
710T	2523	3/93	√	
720/C	9552	5/93	√	
730T	2524	5/94	√	
750C/Cs	9545	9/93	√	
750Ce	9545	11/93	√	
755C/Cs	9545	6/94	√	
755CE/CSE	9545	10/94	√	
755CD	9545	11/94	√	
755CX	9545	5/95	√	
755CV/CDV	9545	5/95	√	
760C	3546	10/95		√
701C	2630	3/95		√
500	2603		√	
510Cs	2604		√	
520CS			√	
530Cs	2605	5/95	√	
N45SL	2614		√	
300	2615		√	
340	2610	7/94	√	
340CSE	2610	2/95	√	
340CSE	2610	3/95	√	
350C	2618		√	
355C/Cs	2619	6/94	√	
360C/Cs/P	2620	5/94	√	
360CE/CSE/PE	2620C	0/94	√	
370C	9545	5/95	√	
345	2610	9/95		√
365C/CD/ CS/CSD	2625	11/95	√	
220	2432	7/93	√	
230Cs	2432	7/94	√	
PS/55	N51SX 8551	12/91	√	
PS/55	N51SLC 8551	5/92	√	
PS/55	N27SX 5527	4/92	√	

PCs

Figure A-12 (Page 2 of 2). Mobil PC Systems - Internal Clock Setting				
System	**Model/ Type**	**Date Available**	**Century Byte Set Manually**	**Century Byte Set Automatically**
PS/55	N23SX 5523	10/91	√	
PS/55	23V 5523	10/92	√	
320	5523	5/93	√	
330Cs	5523	11/93	√	
550BJ	2437	2/93	√	
555BJ	2437	4/94	√	
365	2625	2H95		√

PC Servers

All new models of IBM Servers shipped in 1996 will automatically update the century byte.

Figure A-13. PC Servers - Internal Clock Setting				
Server	**Model/ Type**	**Date Available**	**Century Byte Set Manually**	**Century Byte Set Automatically**
95	all	9/93		√
300	all	3/95	√	
320	all	6/95	√	
500	all	3/95		√

3 ## Network Hardware Products

3 Figure A-14 on page A-36 lists network hardware products that either have no
3 Year2000 impact or will be fixed by year-end 1996.

3 **Note:** The section of the table entitled MANAGEMENT PRODUCTS are related
3 software products; all others are hardware.

Figure A-14 (Page 1 of 3). Year2000-Ready Network Hardware Products			
Product Category	**NHD Product**	**No Impact**	**By Year End 1996**
MANAGEMENT PRODUCTS	LAN Support Program	√	
	LAN Client	√	
	NWays LAN Remote Monitor for Windows, AIX, and HP-UX	√	
	NWays Manager for Windows		√

Figure A-14 (Page 2 of 3). Year2000-Ready Network Hardware Products

Product Category	NHD Product	No Impact	By Year End 1996
CONTROLLERS	3172 NWays Interconnect Controller	√	
	3274 IBM Control Unit	√	
	3174 Establishment Controller	√	
	3705 IBM Communications Controller	√	
	3720 IBM Communications Controller	√	
	3725 IBM Communications Controller	√	
	3745 Communications Controller		√
	3746 NWays MultiNetwork Controller		√
	5394 Remote Control Unit	√	
	5494 Remote Control Unit *	√	
ROUTERS	2210 NWays Multiprotocol Router	√	
	6611 NWays Multiprotocol Router	√	
SWITCHES	2220 NWays Broadband Switch	√	
	8271 NWays Ethernet LAN Switch	√	
	8272 NWays Token Ring LAN Switch	√	
	8285 NWays ATM Workgroup Switch	√	
HUBS	8222 Ethernet Workgroup Hub	√	
	8224 Ethernet Stackable Hub	√	
	8230 TR Network Controlled Access Unit	√	
	8238 TR Stackable Hub	√	
	8250 NWays Multiprotocol Switching Hub	√	
	8260 NWays Multiprotocol Intelligent Hub	√	
CONCENTRATORS	2217 NWays Multiprotocol Concentrator	√	
	2218 NWays Frame Relay Access Device	√	
	8226 TR RJ45 Multistation Access Device	√	
	8240 FDDI Concentrator	√	
	8244 FDDI Workgroup Concentrator	√	
	8282 NWays ATM Workgroup Concentrator	√	
BRIDGES	8209 LAN Bridge	√	
	8229 Bridge	√	
	8281 NWays ATM LAN Bridge	√	
RLAN ACCESS SERVERS	8235 DIALS Server		√
Multiplexers	9729 Optical Wavelength Multiplexer	√	
	9741 Inverse Multiplexer	√	
IDNX	9733 IDNX		√
	9735 IDNX		√
	9736 IDNX		√
	9737 IDNX		√
	9738 IDNX		√
	9739 IDNX		√

Figure A-14 (Page 3 of 3). Year2000-Ready Network Hardware Products

Product Category	NHD Product	No Impact	By Year End 1996
ISDN/WAN	IBM 3278/79 Emulator Adapter	√	
	IBM 3270 Connection	√	
	IBM Wide Area Connecter	√	
	IBM Wave Runner Products	√	
	IBM 7845 ISDN NTE	√	
	IBM Interface Co-Processor	√	
	IBM Interface Co-Processor/2	√	
	ISDN Primary Rate Adapter	√	
LAN ADAPTERS	IBM TURBOWAYS ATM Adapters	√	
	IBM Token Ring Adapters	√	
	IBM Ethernet Adapters	√	
	Home and Away Credit Card Adapter	√	
WIRELESS PRODUCTS	Wireless Modem for Mobitex	√	
	Wireless Modem for Ardis	√	
	Wireless for Cellular/CDPD	√	
	Wireless LAN	√	
	Wireless LAN Entry	√	
MODEMS	Small Office Home Office Assistant	√	
	Multimedia Modem for ISA	√	
	28.8 Kbps Modems	√	
	14.4 Kbps Modems	√	
	Communications Adapters	√	
	58XX Modems	√	
	78XX Modems	√	

Notes:

* You must reset it at the end of 1999 (the same as all “AT-based” PCs) to time stamp error conditions. It will not affect normal operation.

Appendix B. Year2000-Ready Solution Developer Products

Figure B-1 is a list of Year 2000-ready products provided by their respective Solution Developers.

IBM intends to update this list in subsequent releases of this publication. If you have a product that is Year 2000-ready, and you would like IBM to consider including it in this list, please follow the instructions in "Year2000-Ready Solution Developer Product and Tool Authorization" in the back of this book.

Disclaimer

The following is a partial listing of some Solution Developers who represented to us that they provide products that are Year2000 ready. IBM does not make any representations, endorsement, or warranties of any of the following products. The following information was provided by the Solution Developer and no effort has been made by IBM to independently verify the accuracy of the information relating to the Year2000 date transition. The reader is responsible for checking with the individual Solution Developer to confirm the specific implementation of the Year2000 date transition.

Figure B-1 (Page 1 of 5). Year2000-Ready Solution Developer Products

Company	Solution Developer Product	Operating System	Solution Developer Product Version/Release
Advanced Software Technologies Company Ltd.	ASTUTE	MVS	3.2
ALTAI Software	ZACK	MVS	3.20A
	ZACK	VSE	2.10A and up
	ZARA	MVS	1.1.0A
	ZEBB	MVS	2.20A
	ZEKE	MVS	4.1.C and up
	ZEKE	VSE	4.1.C and up
	ZELA	MVS	1.1.0A
American Software, Inc.	AMSOFT Accounts Payable	MVS, VSE	20
	AMSOFT Accounts Receivable	MVS, VSE	20
	AMSOFT Customer Order Processing	MVS, VSE	20
	AMSOFT DRP/2000 Series 3	MVS, VSE	20
	AMSOFT Forecaster/2000	MVS, VSE	20
	AMSOFT Inventory Control & Accounting (IC&A)	MVS, VSE	20
	AMSOFT MRP II	MVS, VSE	20
	AMSOFT Purchasing	MVS, VSE	20

Figure B-1 (Page 2 of 5). Year2000-Ready Solution Developer Products			
Company	**Solution Developer Product**	**Operating System**	**Solution Developer Product Version/Release**
BlueLine Software, Inc.	CACHE/VSE	VSE	2.1.1
	MAXBACK/VSE	VSE	2.2.5
	Multiprint/VM	VM	3.0.1
	Multiterm/VM	VM	5.2.2
	Multiterm/SNA	MVS	2.5.175
	Multiterm/SNA	VM	2.5.175
	Multiterm/SNA	VSE	2.5.175
	Netware for MVS	MVS	2.0.0
	Vital Signs/VM	VM	4.1.1, 4.1.4
	Vital Signs CICS	VSE	4.0.1
	Vital Signs VTAM •	MVS	4.0
	Vital Signs VTAM	VM	3.1.3
	Vital Signs VTAM	VSE	3.1.3
	VLOCK/VM	VM	3.03
CDB	CDB/Part Roll	MVS	all
	CDB/REXX	MVS	all
	CDB/Super Copy	MVS	all
	CDB/Super Load	MVS	all
	CDB/Super Reorg	MVS	all
	CDB/Super Restore	MVS	all
	CDB/Super Unload	MVS	all
Cole Software	XDC	MVS	X3.0
Compuware Corporation	DATA-XPERT	OS/2	1.0
	File-AID/MVS	MVS	8.0
Cybermation Inc.	ESP Workload Manager	MVS	4.4.1
	ESP Workload Manager Extensions	MVS	4.4.1
	ESP Encore	MVS	2.1.3
	ESP Console Manager	MVS	1.1.1
	ESP Communication Server	MVS	1.1.1
	InfoServ	MVS	1.1.1
	ESP API	MVS	3.1.1
Data Base Architects, Inc.	The OLR System •	MVS	all
The DBT Group, Inc.	COMPRESS for DB2	MVS	any
	COMPRESS for IDMS	MVS	any
	COMPRESS for IMS	MVS	any
	DBT/Advisor for VSAM	MVS	any
	DBT/Control for VSAM	MVS	any
DTS Software, Inc.	Allocation Control Center	MVS	1.1, 1.2, 2.1
	Space Recovery System	MVS	1.1, 1.2, 2.1
	Portal 2000	MVS	1.1
Davison Associates, Inc.	MTS/CICS	MVS/CICS	1.1.7 or higher
	MTS/CICS	VSE/CICS	1.1.7 or higher

Figure B-1 (Page 3 of 5). Year2000-Ready Solution Developer Products			
Company	**Solution Developer Product**	**Operating System**	**Solution Developer Product Version/Release**
Firesign Computer Company	Outbound ClientServer	MVS	3.2L
	Outbound ClientServer	VM	3.2L
	Outbound ClientServer	VSE	3.2L
	Outbound Host	MVS	3.2.L
	Outbound Host	VM	3.2.L
	Outbound Host	VSE	3.2.L
Gimpel Software	FlexeLint	VM	all
	FlexeLint	MVS	all
Information Management Company	TUXEDO	MVS	4.2.2
	Open TransPort	MVS	2.0.16
Information Technology Systems, Inc.	SoftSpy	MVS	all
	SoftSpy	VM	all
	SoftSpy	VSE	all
InfoTel Corporation	INFOCOPY for DB2	MVS	any
	INFOCYPHER	MVS	any
	INFOLOAD for DB2	MVS	any
	INFOPAK for MVS,VSAM, DB2, IMS DATACOM, IDMS	MVS	any
	INFORECOVERY for DB2	MVS	any
	INFORECOVERY for IMS	MVS	any
	INFOREORG for DB2	MVS	any
	INFOSCOPE for DB2	MVS	any
	INFOTRACE for DB2	MVS	any
Innovative DP Designs, Inc.	IMS ToolKit		all
IntelliWare Systems, Inc.	IntelliCONSOLE	VSE	all
	IntelliRESOURCE	VSE	all
KTS Informations-Systeme GmbH	DOMESTIC	MVS	95
	DOMESTIC	OS/2	95
Mazda Computer Corp. (Action Software Int.)	Change Action	MVS/XA, MVS/ESA	5.01
Merrill Consultants	MXG	all	all
META Health Technology	CHARMS	MVS	
	CHARMS	VSE	
North Ridge Software	The Network Director	MVS	4.1.0
	The Network Director	VSE	4.1.0
	The Network Director	VM	4.1.0
	The Network Center	VM	1.7.0
	The Network Center	MVS	1.7.0
OSYS AG	QPAC-Batch	MVS	all
	QPAC-Online	VSE	all
Panorama Software Corporation	Sunrise	MVS/ESA	N/A
Real Solutions	SAF	MVS	2.0
RSDSA	EOS	MVS	1.0
	RSD FOLDERS	MVS	3.1

Figure B-1 (Page 4 of 5). Year2000-Ready Solution Developer Products			
Company	**Solution Developer Product**	**Operating System**	**Solution Developer Product Version/Release**
SAP AG, Germany	R/2	MVS	5.0
	R/2	VSE	5.0
Softworks, Inc.	Capacity Plus for VSAM(VCP)	MVS/XA, MVS/ESA	any
	Catalog Solution	MVS/XA, MVS/ESA	any
	HSM Agent	MVS/XA, MVS/ESA	any
	Performance Solution •	MVS/XA, MVS/ESA	any
	Space Recovery Facility	MVS/XA, MVS/ESA	any
	TeraSAM	MVS/XA MVS/ESA	any
	VSAM Assist	MVS/XA, MVS/ESA	any
	VSAM Quick Index	MVS/XA, MVS/ESA	any
	VSAM Space Manager	MVS/XA, MVS/ESA	any
Stonehouse & Company	MONIES	MVS/ESA	
	MONIES	VSE/ESA	
Storage Technology Corporation	CAM	MVS, UNIX	1.0
	CommonParser	MVS, VM, UNIX	1.0
	EMV/MVS	MVS	1.0
	ExLM	MVS	2.1
	ExPR	PC,MVS	1.0.3
	IXFP	MVS,VM	
	LibraryStation	MVS	1.0
	MPPM	PC, VM, MVS, VSE	1.0
	MPST	PC, VM, MVS, VSE	1.0
	MVS/CSC	MVS	2.0
	NASF	MVS	1.1
	RCE	MVS	all
	Stager	MVS,UNIX	1.0
SuperWylbur Systems, Inc	SuperWylbur	MVS	4.3
Systemware	C/QUE	MVS	3.3 and up
	MPS	MVS	3.6
	X/PTR	MVS	3.3 and up
	JHS	MVS	3.3 and up
Thomson Software Products	NOMAD	VM,MVS	all

Figure B-1 (Page 5 of 5). Year2000-Ready Solution Developer Products			
Company	**Solution Developer Product**	**Operating System**	**Solution Developer Product Version/Release**
Ton Beller GmbH	SIRON	AIX	94.1
	SIRON	MVS	94.1
	SIRON	OS/2	94.1
	SIRON	VM	94.1
	SIRON	VSE	94.1
Triangle Systems, Inc.	IOF	MVS	IOF 7C

Notes:

● - Being renamed Vital Signs Vision Net

● - OLR System includes:

- Online Help
- Online Reference
- Online Note pad
- The OLR API

● - Performance Solution includes:

- I/O Plus for VSAM
- I/O Plus for xSAM
- HiperLoad Plus for VSAM
- VSAM I/O Plus

Appendix C. Bibliography

Non-IBM Publications

By Author

1. Adam, Gerhard, *Year 2000: Starting the Assessment Process* Technical Support, Jun 1996, p. 44.
2. Alter, Allan E., *The 2,000-year-old IS manager*, Computerworld, 1 Apr 1996, p. 37.
3. Appleton, Elaine L., *Call in the Cavalry Before 2000*, Datamation, 1 Jan 1996, p. 42.
4. Arnold, Robert S., *Millennium Now: Solutions for Century Date Change Impact*, Application Development Trends, Jan 1995, pp. 60-67.
5. Arnold, Robert S., *On Assurances of Year 2000 Compliance*, Inside DPMA, Data Processing Management Association, Jan 1996.
6. Arnold, Dr. Robert S., *Resolving Year 2000 Problems in Legacy Software*, presentation at Software Quality Week, San Francisco, CA, 1 Jun 1995.
7. Arnold, Robert S., *Tips for Performing A Software Inventory to Resolve the Year 2000 Problem*, Inside DPMA, Data Processing Management Association, Jan 1996.
8. Barker, Paul, *End of Century Problem Looms As IS Remains Idle*, Computing Canada, Vol. 19, No. 23, 8 Nov 1994, pp. 1-4.
9. Barker, Paul, *Study Shows Impact of Year 2000*, Computing Canada, 94-47112, Vol. 19, No. 26, 20 Dec 1993, pp. 1-7.
10. Bartholomew, Doug, *The Year 2000 Problem: Time's Running Out*, InformationWeek, 5 Feb 1996, p. 30.
11. Baum, David, *Tool Up for 2000*, Datamation, 1 Jan 1996, p. 49.
12. Baum, David, *Union Pacific Stays on Track for 2000*, Datamation, 1 Jan 1996, p. 39.
13. Beizer, Boris, *Software Testing Techniques,*, Van Nostrand Reinhold Company, 1983.
14. Betts, Mitch, *Desktops Veer Toward Year 2000 Crisis*, Computerworld, Vol. 28, No. 28, 11 Jul 1994, pp. 1-28.
15. Betts, Mitch, *IBM Pledges Year 2000 Update in '96*, Computerworld, Vol. 29, No. 46, 13 Nov 1995, p. 79,87.
16. Betts, Mitch, *Users Slow to Face Year 2000 Conversion*, Computerworld, Vol. 29, No. 15, 10 Apr 1995, p. 73.
17. Bloom, Al, Managing System Development, Jan 1996.
18. Butler, Ken, *Can Your Computer System Handle the Change of the Century?*, Rough Notes, Vol.138, No. 9, Sep 1995, p. 28.
19. Celko, Joe, *Start Fixing Year 2000 Problems Now*, Datamation, 1 Jan 1996, p. 36.

3 20. Chandler, David L., *Century mark is system downer*, Boston Globe 23 Jun
3 1996, p. 1. The Boston Globe is available on the Internet at: *http://globe.com*

21. Cini, Al, *System Bug of the Apocalypse*, Internetwork, Jan 1995.

22. Cohn, Michael B., *Dateline 1999: IS Pros Retire in Droves*, Computerworld, Vol. 29, No. 44, 30 Oct 1995, p. 37.

23. Cohn, Michael B., *No Need to Fear the Year 2000*, Computerworld, Vol. 28, No. 47, 21 Nov 1994, p. 35.

3 24. Cohn, Mike, *Year 2000? Shut Up Already!*, Midrange Systems, 14 Jun 1996, p.
3 70.

25. Cox, Brian, *Year 2000 is Problematic for Systems*, National Underwriter (Life/Health/Financial Services), Vol. 98, No. 42, 17 Oct 1994, pp.7, 28.

26. Cox, Brian, *Year 2000 is A Big Problem for Systems*, National Underwriter (Life/Health/Financial Services), Vol. 98, No. 47, 21 Nov 1994, pp.9, 18.

27. de Jager, Peter, *Collateral Damage Will Surface*, Computer World Canada, Vol 12, No. 1, 19 Jan 1995, p. 6.

28. de Jager, Peter, *Companies come clean*, Computerworld, Vol 29, No. 47, 20 Nov 1995, pp. 97-100.

29. de Jager, Peter, *Computer D-Day: Jan 1, 2000*, Toronto Star, 16 Jun 1994.

30. de Jager, Peter, *Do your Systems have only 5 years to live?*, Best's Review (Life/Health), Vol. 96, No. 2, May 1995, pp. 88-90.

31. de Jager, Peter, *Doomsday 2000*, Computerworld Vol. 27, No. 36, 6 Sep 1993, pp. 105-109.

32. de Jager, Peter, *Embrace the Future, let go of the Past*, Information Canada, Vol. 19, No. 12, Dec 1994, pp. 5.

33. de Jager, Peter, *Feedback Highlights bad BIOS Blues*, Information Canada, Vol. 20, No. 2, Feb 1995.

34. de Jager, Peter, *If you start now ... you just might make it* Computerworld, Vol. 29, No. 47, 20 Nov. 1995, pp. 97-100.

35. de Jager, Peter, *Only you can Decide*, Information Canada, Jun 1995.

36. de Jager, Peter, *Start now or face 2000 Doom*, Datamation 96-73255, Vol. 41, No. 9, 15 May 1995, p.92, 11.

37. de Jager, Peter, *Take a reporter out to lunch*, Datamation, 1 Jan 1996, p. 76.

38. de Jager, Peter, *Time's a Wasting and The Clocks Keep on Tickin'*, Info Canada, Vol. 18, No. 5, May 1993, p. 7.

39. de Jager, Peter, *2000 or Bust*, CA Magazine, Vol. 127, No. 6, Aug 1994, pp.32-33.

40. DeVoe, Deborah, *Still No Quick Fix to Year 2000 Glitch*, InfoWorld Magazine, Vol. 17, No. 48, 27 Nov. 1995, p. 46.

41. Dunn, Robert H., *Software Defect Removal*, McGraw-Hill, Inc., 1984.

42. Edwards, John, *New Year's Evil*, CIO Magazine, 15, Dec 1995, p. 82.

43. Elms, Teresa, *Global Trends Shape Midrange for 2000*, System 3X/400, Vol. 20, No. 12, Dec 1992, pp. 44-50.

44. Farber, Arnold and Rosemary LaChance, *Time Slips Away: Year 2000 is Closer Than You Think*, Enterprise Systems Journal, Feb 1996, p. 42.

45. Fine, Doug, *Companies Brace for Millennium*, Infoworld, 10 Apr 1995.

46. Furman, Jeff, Albert Marotta, and Cliff Candiotti, *Party When It's 1999*, Software Magazine, Apr 1995, pp. 6-8.

47. Furman, Jeff and Albert Marotta, *Year 2000 Denial*, Computerworld, Vol. 28, No. 43, 24 Oct 1994, pp. 70.

48. Gartner Group Reports and Research Notes:

    a. Case, A., *System Date 2001, a Development Odyssey*, Gartner Group, Application Development & Management Strategies (ADM), Research Note, Strategic Planning SPA-210-835, 8 Feb 1993.

    b. Hall, B. and K. Schick, *Year 2000 Crisis: Estimating the Cost*, Gartner Group, Applications Development and Management Strategies (ADM), Research Note, Key Issue Analysis KA-210-1262 28 Dec 1995.

    c. Hall, B. and K. Schick, *Year 2000 Crisis: Make Applications Compliant by YE1997*, Gartner Group, Applications Development and Management Strategies (ADM), Research Note, Strategic Planning SPA-215-1251 29 Nov 1995.

    d. Jones, N., *PCs and the Year 2000*, Gartner Group, Applications Development and Management Strategies (ADM), Research Note, Strategic Planning, SPA-980-1278, 30 Jan 30, 1996.

    e. Phelps, John, *Year 2000 - Known but Not Understood?*, Gartner Group, Large Computer Research Note, Strategic Planning, SPA-800-1734, 26 Jul 1995.

    f. Schick, K., *An RFP for the Year 2000 Date Change*, Gartner Group, Applications Development and Management Strategies (ADM), Research Note, Tutorials, TU-215-1147, 12 Apr 1995.

    g. Schick, K., *INSPECT Legacy Applications for the Year 2000*, Garter Group, Application Development & Management Strategies (ADM), Research Note, Strategic Planning SPA-215-1146, 12 Apr 1995.

    h. Schick, K., *Three Certainties: Death, Taxes and the Year 2000*, Gartner Group, Application Development & Management Strategies (ADM), Research Note, Strategic Planning SPA-215-1116, 25 Jan 1995.

    i. Schick, K. and C. Germann, *Building Year 2000 Contract Protection*, Gartner Group, Applications Development and Management Strategies (ADM), Research Note, Strategic Planning SPA-215-1252, 29 Nov 1995.

    j. Schick, K. and C. Germann, *Date Change in the Year 2000: A Test of Asset Management*, Gartner Group, Applications Development and Management Strategies (ADM), Research Note, Strategic Planning, SPA-605-228, 25 July 1995.

    k. Schick, K. and C. Germann, *Date Change in the Year 2000: A Test of Asset Management*, Gartner Group, Transition Strategies (TS), Research Note, Strategic Planning, SPA-LEG-162, 7 Aug 1995.

    l. Schick, K., *The Year 2000 Date Change Crisis: Awareness*, Gartner Group Continuous Services, Applications Development & Management Strategies (ADM), Strategic Analysis Report, R-215-130, 9 Feb 1996.

49. Glass, Brett, *Turn of the Century Will Bring Out Flaws in Many Programs*, InfoWorld, Vol. 17, No. 25, 19 Jun 1995, p 42.

50. Glass, Robert L., *The Date Wars: Learning Management by Issue*, Information Systems Management, Vol. 12, No. 2, Spring 1995, pp. 67-69.

3 51. Gleick, James, *Oh-Oh*, New York Times Magazine, 2 Jun 1996, p. 19.

52. Goodwin, Bill, *Tick, Tick, Tick...*, (newsletter), 2000AD, Inc., Brooklyn, NY.

3 53. Governor, James, *UK Post Office Sorts Out The Millennium*, Computing (UK),
3 27 Jun 1996.

54. Greiner, Lynn, *Year 2000 a Time Bomb Ready to Explode*, Computing Canada, Vol. 21, No. 14, 5 Jul 1995, p. 11.

55. Hoffman, Thomas, *1,418 days and counting - CA offers year 2000 date-change products*, Computerworld, 12 Feb 1996.

56. Hayes, Brian, *Waiting for 01-01-00*, American Scientist, Volume 83, Jan-Feb 1995.

57. Jones, David C., *Solving Year 2000 Problem Long, Costly Project*, National Underwriter (Life/Health/Financial Services), Vol. 99, No. 26, 26 Jun 1995, pp.9-24.

3 58. Levy, Stephen, *The 1,000 Year Glitch*, Newsweek, 24 Jun 1996, p. 92.

59. Kilmar, and Wasserman, *Pros/Cons Procedure vs. Data*, American Programmer, Feb 1996.

60. Lips, Michael, *Six-Digit Dates And The Century Change: Complex Problem. Simple Solution?*, Enterprise System Journal, Oct 1993, pp.68-69.

61. Marcoccia, Lou, Managing System Development, Jan 1996.

62. Martin, James, *Information Engineering, Book I Introduction,*, Prentice Hall, 1990.

63. Martin, James, *Information Engineering, Book II Planning and Analysis*, Prentice Hall, 1990.

64. Martin, James, *Information Engineering, Book III Design and Construction*, Prentice Hall, 1990.

65. Martin, James and Joe Leben, *Strategic Information Planning Methodologies*, Second Edition, Prentice Hall, 1989.

66. Martin, James and Carma McClure, *Structured Techniques The Basis for CASE*, Revised Edition, Prentice Hall, 1988.

67. Martin, Larry W., *Millennium Preparation*, Software Magazine, Vol. 13, No. 10, Jul 1993, pp.6-8.

68. McCarthy, Vance, *Keep the Millennium Virus off Your Net*, Datamation, 1 Jan 1996, p. 55.

69. McColgan, Declan, *Countdown to 2000*, Irish Computer, Feb 1996.

70. McKendrick, Joseph, *A Once in a Century Crisis*, Midrange Systems, Vol. 8, No. 12, 30 Jun 1995, pp. 17-18.

71. McKendrick, Joseph, *The Year 2000: Still Way Off*, Midrange Systems, Vol. 8, No. 19, 13 Oct 1995, p. 3.

72. Meador, C. Lawrence, *Solving the Year 2000 Problem*, InformationWeek, 5 Feb 1996, p. 44.

73. Meall, Lesley., *The Century's Time Bomb*, Accountancy, Vol. 116, No 1228, Dec 1995, p. 52.

74. Miller, Edward and William E. Howden, *Tutorial: Software Testing & Validation Techniques*, Second Edition,

3 75. Murray, Jerome. T. and Marilyn. J. Murray, *The Year 2000 Computing Crisis:
3 A Millenium Date Conversion Plan* (formerly titled *Computers in Crisis*),
3 McGraw-Hill, 1996.

76. Neumann, Peter, G., *Computer Related Risks*, Addison-Wesley, ISBN 0-201-55805-X.

77. Olsen, Florence, *Govt. Urged to Perform Triage in Two-Digit Field Repairs*, Government Computer News, 18 Mar 1996.

78. Perry, William E., *A Structured Approach to Systems Testing*, QED Technical Publishing Group, 1983.

79. Perry, William E., *Quality Assurance for Information Systems: Methods Tools, and Techniques*, QED Technical Publishing Group, 1991.

80. Peterson, DuWayne, *The Year 2000: Close Enough to be Dealt With*, SIM Network (Society for Information Management), Feb 1996.

81. Peterson, Ivars, *Fatal Defect*, Times Books, ISBN 0-8129-2023-6.

82. Poland, Tom, *Year 2000 Dilemma Creating Havoc with Software*, National Underwriter (Life/Health/Financial Services), Vol. 98, No. 49, 5 Dec 1994, pp. 38-39.

83. Porter, Alan L., Frederic A. Rossini, Stanley R. Carpenter, A. T. Roper, Ronald W. Larson, and Jeffrey S. Tiller, *A Guidebook for Technology Assessment and Impact Analysis*, North Holland, 1980.

84. Ross, Noah, *The End of the Century is Nearer Than You Think*, Application Development Trends, Apr 1995.

85. Sager, Ira, *Glitch of the Millennium*, Business Week, No. 3450, No. 3450, 13 Nov 1995, p. 54.

86. Scheier, Robert L., *Face up to it*, Computerworld, 25 Mar 1996, pp. 83-84.

87. Schick, Kevin, *The Year 2000 Date Crisis*, Government Finance Review, Vol. 11, No. 4, Aug 1995, pp. 32-33.

88. Schulmeyer, Gordon G., *Zero Defect Software*, McGraw-Hill, Inc., 1990.

89. Schwartz, Susana, *Are Insurers Stuck in a Time Warp?*, Insurance & Technology, Vol. 20, No. 5, May 1995, p. 56.

90. Stedman, Craig, *Utility charges to 2000*, Computerworld, 18 Feb 1996, p. 71.

91. Steward, Donald V., *Software Engineering with System Analysis and Design*, Brooks/Cole Publishing Company, 1987, IEEE Computer Society Press.

92. Stitt, Martin, *Debugging: Creative Techniques and Tools for Software Repair*, John Wiley & Sons, Inc., 1992.

93. Sullivan, R. Lee, *Ghosts in the Machines*, Forbes, 19 Jun 1995.

94. Ulrich, William R., *10 pitfalls to avoid in year 2000 initiatives*, Application Development Trends, Feb 1996.

3 95. *What's the ROI on Objects*, Software Magazine, May 1996.

96. Vandercook, R. Gibbs, *Now Is the Time*, Systems Management, Vol. 23, No. 3, Mar. 1995, p. 66.

97. Xenakis, John J, *The Year 2000 Surprise*, CFO: The Magazine for Senior Financial Executives, Vol. 11, No. 8, Aug 1995, p. 22.

3 98. *Year 2000 Testing Task Ahead*, Software Magazine, May 1996.

99. Yourdon, Edward, *Modern Structured Analysis*, Yourdon Press, Prentice Hall, 1989.

100. Zvegintzov, Nicholas, *The Year 2000 as racket and ruse*, American Programmmer, Feb 1996.

101. Zvegintzov, Nicholas, Managing System Development, Jan 1996.

By Title

1. *Adpac Unveil Date Conversion Tool*, Computerworld, 10 Oct 1994, p. 81.

2. *ANSI Cobol Standard Modified For The Year 2000*, Information Week, 26 Feb 1990, p. 19.

3. ANSI X3.30-1985 (R1991) *Representation for Calendar Date and Ordinal Date for Information Interchange*.

4. ANSI X3.51-1994 *Information Systems -- Representations of Universal Time, Local Time Differentials, and United States Time Zone References for Information Interchange*.

3 5. *Are You Ready for the 21st Century?*, CrossTalk: The Journal of Defense
3 Software Engineering, Mar 1996.

6. *Biting the Bullet*, Computerworld, 18 Mar 1996, p. 16.

7. *CAP Gemini Seeks To Solve Year 2000 Dilemma*, News 3x/400, Jun 95, p. 30.

8. *Computer Horizons Establishes Mini-homepage on de Jager's Year 2000 Information Center Internet Home Page*, PR Newswire Association, 6 Nov 1995, p. 1106NY062.

3 9. *Defusing Millenium Time Bomb*, Jun 1996. At WWW site:
3 *http://www.us.net/signal*

10. *Firms Race Around the Clock to Avert "Millennium Bug" Bite*, Detroit News, 3 Dec 1995, p. C1.

11. *For The Unwary, The New Millennium Could Bring Headaches*, Newsbytes News Network, Seattle, WA, 30 Mar 1995.

12. *IBM Counts on Developers, VARs to Ring in Year 2000*, Computer Reseller News, CMP Publications, Inc., 13 Nov. 1995, p. 96.

13. *IBM Plans Year 2000 Upgrades*, InformationWeek, CMP Publications, Inc., 13 Nov. 1995, p. 30.

14. *IBM Prepares for Year 2000 Date*, Newsbytes News Network, 2 Nov 1995.

15. *IT Begins to Face Year 2000 PC Data Problem*, Newbytes News Network, 31 Oct. 1995.

16. ISO 8601:1988 *Data elements and interchange formats -- Information interchange -- Representation of dates and times*, Technical Committee ISO/TC 154.

17. *Keane, VIASOFT Announce Agreement to Provide Complete Year 2000 Solution*, Business Wire, 4 Nov. 1995, p11061123.

18. *Mainframe Time Bomb: 01-01-00*, Fortune Magazine, 6 Mar 1995, p. 24.

19. *Micro Focus Offers Year 2000 Fixer*, Computerworld, 31 Jul 1995, p. 8.

20. *Millennium could bring computer chaos*, Southam News organization as reported in The Daily News, Halifax, N.S., Canada, Feb 1996.

21. *The Millennium: Problems and Solutions*, Xephon, pp 173.

22. *Millennium Countdown of 1996* (subtitles *Ready or Not, Here Comes the Millennium* and *Can't You Just Push 'Reset'*), 1996 Old Farmer's Almanac, pp 46-51.

23. *New OS/390 handles year 2000*, Computerworld, 19 Feb 1996,

24. *Ottawa Confident of Escaping Year 2000 Rollover Debacle*, Technology in Government, Feb 1996.

25. *Pain or Gain in the Year 2000?*, Computer Business Review, Mar 1996.

26. *SEEC Ship COBOL Tool Upgrade*, PC Week, 21 Nov 94, p. 31.

27. *Seec Upgrades Analysis & Maintenance*, Software Magazine, Feb 1995, p. 93.

28. *Service for Year 2000 Year Date Processing of Programs Software*, New Technology Japan, Feb. 1995, p. 24.

29. *Something's Got to Give*, New York Times Magazine, 24 Mar 1996.

30. *Technology Problem Of The Century*, Bank Technology News, Faulkner & Gray, Inc., Jun. 1995, p. 10.

31. *There Is No Feb 29, 2000!*, Datamation, 1 Apr 1996, pp 8-9.

32. *Tick Tock...*, Midrange Systems, 24 May 1996.

33. *Time running out for Year 2000 transition*, Network World Canada, Vol. 6, No. 2, 2 Feb 2 1996

34. *Troubled Time*, Associated Press, Denver Post, (and various other USA newspapers), 25 May 1995.

35. *Users Invest in Millennium Changes*, Xephon 1995, Insight IS, pp. 12-13.

36. *Vendors Focus On Year 2000 Conversion Issue*, Computerworld, 10 Apr 1995, p. 73.

37. *VIASOFT and Litton Computer Services Enter into a Strategic Alliance*, Business Wire, 1 Dec 1995.

38. *Viasoft's Unveil Code Decipher Software*, Computerworld, 15 Aug 1994, p. 16.
Worried About 2000? Group Shares The Fretting, Investor's Business Daily, 7 Feb 1996, p.A6.

39. *Year 2000 causes double trouble for feds (Countdown to Calamity)*, Federal Computer Week, 1 Apr 1996.

40. *Year 2000: Keeping Doomsday Costs Down*, Enterprise Data Center Strategies, Meta Group, Inc., File No. 431, 29 Dec 1994.

41. *The Year 2000 Doesn't Compute*, Electronic Buyers' News, CMP Publications, Inc., 2 Jan 1996, p. 2.

Electronic (Internet/World-Wide Web) Documentation

1. ANSI Online Home Page (access to ANSI catalog and standards documents):

   http://www.ansi.org

   You can order both ANSI and ISO standards from ANSI by calling (212) 642-4900 between 8:45 a.m. and 4:45 p.m. eastern time. (Note: this is New York City, New York, USA.)

3 2. The Boston Globe Home Page:

3 http://globe.com

3. Cable News Network - *The year 2000 does not compute*, CNN′s Headline News, 7 January 1996:

   http://www.cnn.com/TECH/9601/2000/index.html

4. Federal Year 2000 Interagency Task Force (draft proposal for "Recommended Year 2000 Contract Language") at:
   http://www.itpolicy.gov.80/library/yr2000/y209rfp1.htm

5. Datamation Home Page (*Year 2000: Fix It Now!*, Special Report: Year 2000, 1 January 1996)

   http://www.datamation.com/PlugIn/issues/1996/jan1/FEATURES.html

6. *The Globe and Mail* (Canada′s National Newspaper):

   http://www.globeandmail.ca/Cyber-17.html

7. IBM Software Page

   To access this document (*The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation*) through the Internet′s World Wide Web, visit the IBM Software Page at:

   http://www.software.ibm.com/

   or IBM System/390 Home Page at:

   http://www.s390.ibm.com/stories/tran2000.html

   Select an appropriate version of this document to download the file for printing to your printer; the following formats are available:

   ASCII text
   3 Self-extracting zip file of ASCII text
   Browseable BookManager
   PostScript
   UNIX compressed

   You can also access this document through an anonymous ftp whose URL is:

   3 ftp://lscftp.pok.ibm.com/pub/year2000/y2kpaper.**format_type**

   where **format_type** is specific to one of the print formats listed above.

8. IBM DFSORT/MVS Home Page

   To access DFSORT/MVS URL and read about DFSORT′s Year2000 enhancements:

   http://www.storage.ibm.com/storage/software/sort/srtmhome.htm

   You can also download PostScript and/or text versions of DFSORT′s year2000 enhancements (name: sortpy2k.zip) via anonymous ftp:

   ftp://lscftp.pok.ibm.com/pub/mvs/docs/

The SORT2000 package is also available on MKTTOOLS from your IBM marketing representative.

9. IBM VSE and Year 2000 Home Page

   To access VSE URL and read about VSE and Year 2000, access VSE/ESA's home page at:

   http://www.ibm.de/go/d00000166

10. InfoWorld, *Enterprise Computing: Client-Server Year 2000 problems*, 3 Jun 1996, pp.69-71.

11. Information Technology Association of America (ITAA). White paper available at:

    http://www.itaa.org

12. InformationWeek Year 2000 chat session:

    http://www.cmp.com/cgi-bin/techtalk/2000

13. InformationWeek web site:

    http://techweb.cmp.com/techweb/iw/current/

14. ISO Online Home Page (access to ISO catalog and standards documents):

    http://www.iso.ch

    You can order both ASO and ANSI standards from ANSI by calling (212) 642-4900 between 8:45 a.m. and 4:45 p.m. eastern time. (Note: this is New York City, New York, USA.)

15. Microsoft website at:

    http://www.microsoft.com/corpinfo

16. Microsoft's Technet (Year2000 article) at URL:

    http:/www.microsoft.com/technet/tnnews/features/2000.htm

17. MITRE Corporation - Year 2000 Home Page (to include *Year 2000 Problem* and tools) at:

    http://www.mitre.org/research/y2k/

    The Year 2000 Tools Pages' URL is:

    http://www.mitre.org/research/y2k/docs/TOOLS_CAT.html

18. National Bulletin Board for Year 2000 Home Page:

    http://www.it2000.com

19. Project 21st. Century Home Page at:

    http://www.webhelp.com/future.htm

20. Questicon Home Page (to include a year 2000 checklist (QCON2000.PPT)).

    http://questicon.com/products.htm.

21. Royal Greenwich Observatory (RGO) (2000 AD, Leap Years,...):

    http://www.ast.cam.ac.uk/RGO/leaflets/

22. *Summary of the International Standard Date and Time Notation*. WWW site at:

    http://www.ft.uni-erlangen.de/~ mskuhn/iso-time.html.

23. *TechLink Newsletter* from CMP's TechWeb - The Technology Information Source, 8 Feb 1996.

http://techweb.cmp.com

- Cover story location: http://techweb.cmp.com/iw/565/65mtyr2.htm
- Open Labs location: http://techweb.cmp.com/iw/565/65olyr2.htm
- Threaded chat location: http://www.cmp.com/cgi-bin/techtalk/2000

24. Tick! Tick! Tick! Home Page (to include a list of Year2000 user groups):

    http://www.henterprises.com/tick3/

25. Texas Tech University Health Sciences web site (links to magazine articles and Year2000 Home Page and an archive of the Year 2000 newsgroup's daily digests) at:

    http://www.ttuhsc.edu/pages/year2000/ttuy2k.htm

26. USA Today Web site at:

    http://www.usatoday.com/news/comment/colmane.htm

27. U.S. Dept. of Commerce, National Institute of Standards and Technology, 25 Mar 1996. (Change to FIPS Publication 4-1 concerning the format of year information.) NIST WWW site at:

    http://www.itpolicy.gsa.gov:80/library/yr2000/

28. XPRESS Software, Inc. Home Page:

    http://www.xpsoft.com/

29. Year 2000 Frequently Asked Questions (FAQ) access

    ftp://www.year2000.com/pub/year2000/y2kfaq.txt

30. The Year 2000 Home Page (facilitator: Peter de Jager):

    http://www.year2000.com/

    To subscribe to the year2000 general forum, send:

    SUBSCRIBE YEAR2000

    in the body of a message to: listmanager@hookup.net

31. *Year 2000 News...* (E-zine) (facilitator: Dr. Robert S. Arnold):

    To subscribe, send E-mail note to: new2000-request@andrew.cais.com

    enter SUBSCRIBE

    on the SUBJECT line

32. Year 2000 Software Control

    http://www.iac.honeywell.com/Pub/Tech/CM/CMTools.html

33. Year 2000 Technical Audit Center

    http://www.auditserve.com/yr2000/countdown.cgi?2000,1,0,XX,XX,XX

34. Year 2000 Web Links

    http://www.club.innet.be/~ janjedsp/y2k.html

Audios and Videos

1. *Millennium: A Billion Dollar Software Crisis*, Computer Channel, Inc., Presenters: Dr. Howard Rubin (Hunter College) and Jim Woodward (CAP Gemini America). Contact:

   Computer Channel
   6801 Jericho Turnpike
   Syosset, New York 11791
   telephone: (516) 921-5170

2. *Millennium Madness*, panel discussion at 7th Annual Data Administration Management Association International Symposium, panel members: Larry English, Clive Finkelstein, Ron Ross, and John Zachman, Vancouver, Canada, May, 1995. To order the VHS format video:

   DAMA International
   P.O. Box 6096
   Chesterfield, Missouri 63006-6096

3. *Systems for the year 2000: The Upcoming Data Crisis*, Peter de Jager, 1995 Systems Forum & Exhibit (LOMA), 1995. To order the 60-minute audio tape:

   Peter de Jager
   22 Marchbank Cres.
   Brampton, Ontario L6S 3B1
   CANADA

IBM Publications

By Author

1. Frawley, M. D. and H. L. Stuck, *Year 2000 Clean*, Version 1.3, Document Number TR 29.1601, 1994.
2. Hillier, Richard, *Year 2000 Working Document*, Document Number YE-01, IBM Euroco.
3. Ohms, Bruce G., *Computer Processing of Dates Outside the Twentieth Century*, IBM System Journal, Vol. 25, No. 2, 1986, pp.244, G321-5274

Standard Publications

1. *CICS/ESA (Version 4 Release 1): System Definition Guide*, SC33-1164
2. *DB2 SQL Reference*, SC26-4890
3. *IBM High Level Assembler/MVS & VM & VSE: Language Reference (Release 1)*, SC26-4940
4. *IBM VSE/Enterprise Systems Architecture (Version 1 Release 3): System Macros Reference*, SC33-6516
5. *IMS/ESA Version 4: Utilities Reference: Database*, SC26-4627
6. *OS/390 MVS Assembler Services Guide*, GC28-1762
7. *OS/390 MVS System Commands*, GC28-1781
8. *OS/390 MVS Assembler Services Reference*, GC28-1910
9. *National Language Design Guide Volume 2: National Language Support Reference Manual*, Fourth Edition, IBM Corp., 1994.
10. *ESA/390 Principles of Operations*, SA22-7201
11. *Systems Application Architecture: Common Programming Interface C Reference - Level 2*, SC09-1308
12. *Systems Application Architecture: Common Programming Interface COBOL Level 2 Reference*, SC26-4726
13. *Systems Application Architecture: Common Programming Interface Language Environment Reference*, SC26-4970
14. *Systems Application Architecture: Common Programming Interface PL/I Reference*, SC26-4381
15. *Systems Application Architecture: Common Programming Interface REXX Level 2 Reference*, SC24-5549
16. *Virtual Machine/ Enterprise Systems Architecture: CP Command CP Command and Utility (Release 2)*, SC24-5519
17. *VS COBOL II: Application Programming Language Reference*, GC26-4047
18. *VS FORTRAN (Version 2 Release 5): Language and Library Reference*, SC26-4221
19. *Why Migrate to COBOL/370 and LE/370?*, COBMGVAL PACKAGE on MKTTOOLS.

Appendix D. Glossary

This glossary defines data processing and communication terms used in this publication.

Numerics

2-digit-year format. A format that provides a year date as two digits only to represent a year within a specific century. The two high-order digits of the year are truncated; for example 1995 is represented as 95, and 00 represents years ...1800, 1900, 2000....

4-digit-year format. A format that provides a year date as four digits: the two high-order digits represent the century and the two low-order digits represent the year within the century. For example, 1995 represents the year 1995; 2095 represents the year 2095.

C

CCYY format. A 4-digit-year format that uses two century digits (CC) to indicate the century and two year digits (YY) to indicate the year within the century. The CC representation is provided as either the actual century digits (for example, 18, 19, or 20) or as a encoded value (for example, as 00 to represent 19, 01 to represent 20 as in, 0095 represents the year 1995 and 0195 represents the year 2095.)

century. A period of 100 consecutive years.

Although IBM recognizes that the 21st century begins at 0000 hrs, 1 January 2001, for purposes of this document, we are defining the 20th—21st century boundary to be between 2400 hrs, 31 December 1999 and 0000 hrs, 1 January 2000. This allows a discussion of the 21st century to include all dates with a 20yy format inclusive of the year 2000. Hence, the year 2100 is likewise relegated to the 22nd century.

century byte. The high order byte of a field used to contain the two high order digits of a 4-digit year. (For example, 19 in 1995, 20 in 2000 and 2001).

cosmetic. Referring to a 2-digit-year date that is viewed by human eyes only, such as a print date on hardcopy output or a date on a selection panel. Because it is neither read nor further processed by a program you might be able to exclude its modification from your year2000 work effort.

E

external side. The receiver of a data entity. Used in this document to mean a module or routine that accepts a 2- or 4-digit-date format entity for further processing from another module or routine.

F

fixed window. A technique to determine the century (high-order digits) of a year when represented by two digits. The 2-digit year is compared against a hardcoded threshold. The century designation is limited to a 100-year range spanning only two centuries. For example, assume the threshold is 60, then if the 2-digit year is $\geq$60, the year is in the 20th century; if the 2-digit year is <60, the year is in the 21st century.

G

Gregorian calendar. Today's general-use calendar of 12 months and 365 days that employs the current leap year algorithm (refer to **Leap year** below).

I

internal side. The creator or manipulator of a data entity. Used in this document to mean a module or routine that externalizes a 2- or 4-digit-year format entity to another module or routine.

J

Julian date. A date in the format YYDDD. A date format that contains the year in positions 1 and 2, and the day in positions 3 through 5. The day is represented as 1 through 366, right adjusted, padded with zeroes on the left.

L

Leap year. A year either evenly divisible by 400 **or** evenly divisible by 4 and not evenly divisible by 100. For example, the year 1900 was not a leap year but the year 2000 is a leap year.

Lilian date. The number of days since 14 October 1582. 15 October 1582 is Lilian day 1, 16 October 1582 is Lilian day 2, and so on. (Named for Aloysius Lilius (an advisor to Pope Gregory XIII) who, together with his brother, constructed the current Gregorian calendar.)

R

rolling window. Synonymous with **sliding window**.

S

sliding window. A technique to determine the century (high-order digits) of a year when represented by two digits. The user specifies the number of years (both past and future) within a 100-year window spanning two centuries. For example, assume the window is set at 19 future years (1996-2014) and 80 past years (1915-1994). Dates in the range 00-14 (inclusive) are designated 21st century dates because they fall into the future window. Dates in the range 15-99 (inclusive) fall into the 20th century.

T

TRANSFORMATION 2000. IBM's Information Systems Solution Corporation (ISSC) comprehensive set of solutions that takes into account applications, systems software and hardware in both a centralized and/or distributed environment.

Y

Year2000 challenge. The work-effort required to complete a Year2000 transition, to include planning, identifying, reformatting, testing, and migrating phases.

Year2000 problem. The potential problems and its variations that might be encountered in any level of computer hardware or software from microcode to application programs, files, and databases that need to correctly interpret year-date data represented in 2-digit-year format.

Year2000 ready. The capability of a software program to correctly interpret and manipulate year-date data outside the 1900-1999 year range and produce valid arithmetic results.

Year2000 transition. The process of revising all programming entities (programs, databases, and so on) to correctly process date data outside the range 1900-1999.

YY format. Synonymous with **2-digit-year format**.

YYYY format. Synonymous with **4-digit-year format** and a subset of **CCYY format**.

Numerics

20th century. The period of time 0000.00 hrs 1 January 1901 through 2400.00 hrs 31 December 2000.

21st century. The period of time 0000.00 hrs 1 January 2001 through 2400.00 hrs 31 December 2100.

Index

U

V

W

Y

Year2000-Ready Solution Developer Product and Tool Authorization

If your product(s) is year2000-ready or you market tools designed to assist others attain year2000 readiness, and you would like IBM to consider including that product(s) in this publication, please supply the following information. For products not currently "Year2000 ready," please notify us as they become so. Please return to:

Year2000 Solution Developer Liaison
IBM Corporation
Dept. MNRA, Bldg 005-3, M/S P136
522 South Road
Poughkeepsie, NY 12601-5400
Fax: (914) 432-9418

Your Product's Name *	Your Product's Version/Release	IBM Platform (and Version/Release) your product supports	Technical Contact (name, tele/e-mail)	Year2000 Tool? **

* Please indicate any trademarks.
** If your product is a tool, please categorize it as one of the following types (fully defined in Chapter 7, "Tool Categories and Available Tools to Ease Year2000 Changes" on page 7-1):

IA - Impact Analysis CER - Code Editing and Restructure PM - Project Management
CG - Code Generation PLA - Project Level Anaylsis AT - Automate Testing

I hereby authorize IBM to reference my company's name, address and phone number, along with the products and registered trademarks in conjunction with IBM's marketing activities. This information may be edited as IBM deems appropriate, and may be included in, but not limited to, IBM's publication on the year 2000 date format transition, *The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation.*

Company: ____

Address: ____

____

Phone Number: (____)-_______-_______ FAX Number: (____)-_______-_______

E-mail address: ______________________________________

Name: __________________ Title: ______________________________________

Signature: ____

Communicating Your Comments to IBM

Fourth Edition

The Year 2000 and 2-Digit Dates:
A Guide for Planning and Implementation

Publication No. GC28-1251-03

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM. Whichever method you choose, make sure you send your name, address, and telephone number if you would like a reply.

Feel free to comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. However, the comments you send should pertain to only the information in this manual and the way in which the information is presented. To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

If you are mailing an RCF from a country other than the United States, you can give the RCF to the local IBM branch office or IBM representative for postage-paid mailing.

- If you prefer to send comments by mail, use the RCF at the back of this book.
- If you prefer to send comments by FAX, use this number:
  - FAX: (International Access Code)+1+914+432-9405
- If you prefer to send comments electronically, use this network ID:
  - IBMLink: (United States customers only): KGNVMC(MHVRCFS)
  - IBM Mail Exchange: USIB6TC9 at IBMMAIL
  - Internet e-mail: mhvrcfs@vnet.ibm.com
  - World Wide Web: http://www.s390.ibm.com/os390

Make sure to include the following in your note:

- Title and publication number of this book
- Page number or topic to which your comment applies

Optionally, if you include your telephone number, we will be able to respond to your comments by phone.

Reader′s Comments — We′d Like to Hear from You

Fourth Edition

The Year 2000 and 2-Digit Dates:
A Guide for Planning and Implementation

Publication No. GC28-1251-03

You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you. Your comments will be sent to the author′s department for whatever review and action, if any, are deemed appropriate.

Note: Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.

Today′s date: ______________________________

What is your occupation?

Newsletter number of latest Technical Newsletter (if any) concerning this publication:

How did you use this publication?

[ ]	As an introduction	[ ]	As a text (student)
[ ]	As a reference manual	[ ]	As a text (instructor)
[ ]	For another purpose (explain)		

Is there anything you especially like or dislike about the organization, presentation, or writing in this manual? Helpful comments include general usefulness of the book; possible additions, deletions, and clarifications; specific errors and omissions.

Page Number: Comment:

Name

Address

Company or Organization

Phone No.

Reader's Comments — We'd Like to Hear from You
GC28-1251-03

Cut or Fold
Along Line

Fold and Tape

Please do not staple

Fold and Tape

NO POSTAGE
NECESSARY
IF MAILED IN THE
UNITED STATES

BUSINESS REPLY MAIL

FIRST-CLASS MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

IBM Corporation
Department 55JA, Mail Station P384
522 South Road
Poughkeepsie NY 12601-5400

Fold and Tape

Please do not staple

Fold and Tape

Cut or Fold
Along Line

Printed in U.S.A.

GC28-1251-03