1998 February

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

GC28-1251-08

1998 February

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

GC28-1251-08

Note

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

Ninth Edition, February 1998

This is a major revision of, and obsoletes, GC28-1251-07.

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

Executive Summary

The Year 2000 and 2-Digit Dates: Guide

Figures

About This Book

Content

This book provides information to help you:

- Understand the cause and scope of using dates represented by 2-digit years
- Identify problems with programs using 2-digit-year 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

Softcopy Accessibility

The official copy of the book is the on-line version. You can access this document at:

`http://www.software.ibm.com/year2000/resource.html`

in the following formats:

- ASCII text
- Self-extracting zip file of ASCII text
- Browseable BookManager

You can also access this book through the IBM Year 2000 Technical Support Center website at:

`http://www.ibm.com/year2000`

or contact the IBM Year 2000 Technical Support Center by telephone at:
1-800-IBM-4YOU (1-800-426-4968)
and mention the “Callown Keyword” of “Y2K.”

Refer to “Electronic (Internet/World-Wide Web) Documentation” on page C-10 for further information and related IBM on-line documentation.

This publication is continuously being updated as IBM obtains new information and updates old information.

Year 2000 Technical Support Center (TSC)

IBM's Year 2000 Technical Support Center offers help, advice, and information to customers about the Year 2000 transition. If you have questions relating to the Year 2000 transition, you can E-mail the TSC or join their YEAR2000 forum. For more information on contacting the TSC, visit their HomePage at:

`http://www.software.ibm.com/year2000`

Technical questions regarding specific products, release levels, and other technical issues discussed in this book should be directed to IBM's Year 2000 TSC.

Phone Support Lines

International phone support lines, broken out into international geographic regions, are as follows:

Europe, Middle East, and Africa (EMEA)

The following are the phone numbers (international direct dial numbers) for Year2000 TSC in EMEA:

English 353 1 8159600
Fax Number 353 1 8159601

Freefone Numbers

France 0800 907180
Germany 0130 815244
Italy 1677 80310
Spain 900 982908
United Kingdom 0800 973219

EMEA E-mail Y2KTSC@IE.IBM.COM

Asia and Pacific (AP)

The following are the toll-free phone numbers (international direct dial numbers) for Year2000 TSC in AP:

Australia (domestic) 1800 637713
China 10800 3643
Hong Kong 800 933 594
Indonesia 001 800 61 423
Japan 0031 61 6429
Malaysia 1800 80 1999
New Zealand 0800 44 4714
Philippines 1235 616 784 772
Singapore 800 6161 468
South Korea 0078 611 2156
Taiwan 0080 611 256
Thailand 001 800 611 2159

Note: IBM personnel needing to contact the AP Support Centers from within the AP region should use tie-lines to reach the TSC. For most countries, use the following:

- Japan, Greater China Group, Korea: 1 161 49922
- All other AP countries: 6 161 49922

AP E-mail y2ktscap@vnet.ibm.com

Latin America (LA)

The following are the phone numbers (international direct dial numbers) for Year2000 TSC in LA:

Argentina 1-319-7347
Latin America 54-1-319-7347

Note: Due to phone limitations, the initial (customer-initiated) call is not free. IBM will call you back to minimize your cost.

USA and Canada

Contact the IBM Year 2000 Technical Support Center by telephone at:
1-800-IBM-4YOU (1-800-426-4968)
and mention the “Call-down Keyword” of “Y2K.”

IBM Year2000 Education Packages

Business Partners can attend Year2000 classes and order CD-ROMS.

CD-ROM Training

Year 2000 Transition -- A Business Challenge, (PID# 5639-C30) or PUB# ZBOF-5244) has content similar to that of the seminars. This is a self-paced, computer-based training (CBT) course for executives, IS managers, and IT development designed to define the Year2000 business challenge. This course supports IBM's Year2000 Five-Point strategy by increasing customer awareness and assisting customers in planning their own Year2000 strategies.

To order:

United States:	call (800) IBM-TEACh (800-426-8324)	
Canada:	call (800) IBM-4YOU (800-426-4968)	
Latin America and Asia Pacific:	mail orders to:	IBM Direct Services Sortemossvej 21,3450 Allerød, DENMARK
	or call:	(+45) 48 10-1540 (English)
Europe, Middle East, and Asia:	CD currently unavailable	

Classroom Courses

Classes are available in COBOL and Language Environment for VSE (N2020); COBOL and Language Environment for MVS (N2027).

Additional Information

For IBM education and training call 1-800-IBM-TEACh (1-800-426-8322) or visit web site:

AS/400 Partners in Development:	http://www.training.ibm.com/ibm.edu
(including education offerings):	http://www.softmall.ibm.com/as400/year2000.html

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 2, "The Year 2000 - A Transition" on page 2-1
- 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 Chapter 1, "Executive Summary" on page 1-1 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:
  - Chapter 1, "Executive Summary" on page 1-1
  - Chapter 2, "The Year 2000 - A Transition" on page 2-1.
- For project planning information and recommendations, read Chapter 3, "Planning to Resolve Your Year2000 Exposures" on page 3-1.
- For migration information and recommendations, read Chapter 7, "Migration Consideration for Year2000 Transition" on page 7-1.
- For a list of IBM and Solution Developer tools available to assist in identifying, coding, and testing Year2000 changes, read Chapter 8, "Tool Categories and Available Tools to Ease Year2000 Changes" on page 8-1.

System Programmers

- For an overview, you should read Chapter 2, “The Year 2000 - A Transition” on page 2-1.
- To determine the best technique(s) for reformatting year-date notation, you should read Chapter 5, “Reformatting Year-Date Notation” on page 5-1.
- For migration information and recommendations, read Chapter 7, “Migration Consideration for Year2000 Transition” on page 7-1.
- To assist you in testing your Year2000 modifications, you should read:
  - Chapter 6, “Testing Techniques for Year2000 Changes” on page 6-1
  - Chapter 8, “Tool Categories and Available Tools to Ease Year2000 Changes” on page 8-1.

Application Programmers

- For an overview, you should read Chapter 2, “The Year 2000 - A Transition” on page 2-1.
- To understand how to identify the potential exposures caused by using 2-digit-year representations of dates, read Chapter 4, “Identifying 2-Digit-Year Exposures” on page 4-1.
- To determine the best technique(s) for reformatting year-date notation, you should read Chapter 5, “Reformatting Year-Date Notation” on page 5-1.
- To assist in your testing of your Year2000 modifications, you should read:
  - Chapter 6, “Testing Techniques for Year2000 Changes” on page 6-1.
  - Chapter 8, “Tool Categories and Available Tools to Ease Year2000 Changes” on page 8-1.

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 is providing information in this document to assist you in understanding and addressing your Year2000 challenge. IBM does not guarantee your results. You are solely responsible for implementation of your Year2000 project.

Any pointers in this publication to non-IBM Web sites are provided by IBM for convenience only, and are not in any manner an endorsement by IBM of these Web sites or the information contained on them.

This publication is constantly being updated as IBM obtains new information and updates old information.

You should always remember that when accessing or modifying the intellectual property of others (for example, Solution Developer software programs), you may need the consent of the licensor or owner. Always check your license agreement to make sure your actions do not infringe on the property rights of others.

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
- Client Access
- Client Access/400
- COBOL/370
- COBOL/400
- CT
- CUA
- C/370
- C/400
- DATABASE 2
- DataHub
- DataJoiner
- DataPropagator
- DB2
- DB2/400
- DB2/6000
- 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
- geoGPG
- geoGPG RTF
- geoGPG/6000
- geoManager
- HACMP/6000
- Hardware Configuration Definition
- IBM
- IBMLink
- ImagePlus
- IMS
- IMS/ESA
- InfoCrafter
- InfoExplorer
- Integrated Language Environment
- ISPF
- ISSC
- KnowledgeTool
- LANDP
- Language Environment
- LinkWay
- LoadLeveler
- MERVA
- MQSeries
- MVS/DFP
- NetFinity
- MVS/ESA
- NetDoor
- NetView
- NTuneMON
- NTuneNCP
- Nways
- OfficeVision
- OfficeVision/MVS
- OfficeVision/400
- OPC
- OpenEdition
- Operating System/400
- OS/2
- OS/390
- OS/400
- Personalized Learning Series
- PowerPC
- Power Series
- Presentation Manager
- Print Services Facility
- ProBranch
- PROFS
- ProductManager
- PSF
- PS/1
- PS/2
- QMF
- RACF
- ReDiscovery
- Repository Manager/MVS
- Resource Measurement Facility
- RISC System/6000
- RMF
- RMONitor
- RPG/400

- RPG/4000
- RS/6000
- RT
- SAA
- SchoolVista
- Series/1
- SMARTdata UTILITIES
- SOMobjects
- SP
- SQL/DS
- StorePlace
- SwitchServer/2
- Sysplex Timer
- SystemView
- System/36
- System/390
- System/38
- S/370
- S/390
- TeamConnection
- The Integrated Reasoning Shell
- Time and Place
- Trouble Ticket
- TURBOWAYS
- Ultimedia
- ValuePoint
- VisualAge
- VisualGen
- VisualInfo
- Visualization Data Explorer
- VisualLift
- VM/ESA
- VSE/ESA
- Write Along
- Writing to Read
- Writing to Read 2000
- Writing to Write
- XT
- 3890
- 3890/XP

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
ADINA	ADINA R&D, Inc.
Advantage Financial	American Management Systems
Advantage HR	American Management Systems
Aegis	New World Systems
AFM	Interprocessor Systems, Inc.
APM	Programart Corporation
AutoTester 2000	AutoTester Inc.
Bankon DES Encryption	Sercon Corporation
Bankon Telebank System	Sercon Corporation
BEA Tuxedo	Data Services Gmbh & Co. KG
BYPASS2000	HAL Informatica slr
CA-Impact	Computer Associates
C/QUE	Systemware
Canada's National Newspaper	Globe and Mail
Challenge 2000	Micro Focus
CL/2000	Shaw Systems Associates, Inc.

Trademark	Company
CS/2000	Shaw Systems Associates, Inc.
COBOL Analyst	SEEC, Inc.
Change Action	Mazda Computer Corp.
DateServer	Computer Software Corp.
DATE/2000	DATE/2000
Docu-Mint	Business Computer Design (BCD)
DRS	Levi, Ray & Shoup
D2K/Plus	Government Micro Resources Technologists International
Fusion FTMS	Proginet Corporation
Globe and Mail	Globe and Mail
EnCura Managed Care Systems	Health Systems Integration, Inc.
GOLD-line	Data Services Gmbh & Co. KG
Hermes	New World Systems
Host Bridge	Amsys North America
IL/2000	Shaw Systems Associates, Inc.
Ind$ File Plus	Proginet Corporation
IntelliCONSOLE	IntelliWare Systems, Inc.
IntelliRESOURCE	IntelliWare Systems, Inc.
INTO 2000	Information Business Systems
Logos	New World Systems
LT/2000	Shaw Systems Associates, Inc.
JHS	Systemware
MailBook	MailBook
MAX BATCH/PLUS	MAX Software LLC
MAX DATA/UTIL	MAX Software LLC
MAX/REXX	MAX Software LLC
MAX/SPF	MAX Software LLC
MAX Software	MAX Software LLC
METHODMANAGER	Manager Software Products Ltd.
MOVEX	INTENTIA
MPS	Systemware
MXG	Merrill Consultants
NOMAD	Thomson Software Products
NOMAD for UNIX	Thomson Software Products
NXL2000	Formal Systems Inc.
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.
OPAC-Batch	Osys AG
OPAC-Online	Osys AG
Open Workgroup Repository	Manager Software Products Ltd.
MessengerNet	Bytware, Inc.
MessengerOne	Bytware, Inc.
MessengerPlus	Bytware, Inc.
PeekPlus	Bytware, Inc.
Portal 2000	DTS Software, Inc.
ProCreate	Banner Software Inc.
ProTerm	ProTerm Technologies
Sector Trading Systems	Quant Trading Inc.
RECYC2000	Informission Group, Inc.
REFINE Language Tools	Reasoning Systems
SecurPass	Proginet Corporation
SELCOPY	Compute (Bridgend) Ltd.
Software Refinery	Reasoning Systems
Strobe	Programart Corporation

Trademark	Company
Super Check	G. G. Pulley & Associates, Inc.
Super Image	G. G. Pulley & Associates, Inc.
Super Sort	G. G. Pulley & Associates, Inc.
SuperWylbur	SuperWylbur Systems, Inc.
SystemVision	ADPAC Corp.
Tool-Time	Shelby Software
Thoms	New World Systems
Tool-Time 2000	R&L Software Associates, Inc.
Training Management Systems	Systems Design & Development, Inc.
TransCentury	TransCentury Data Systems
Turnover	SoftLanding Systems, Inc.
Vantage YR2000	Millennium Dynamics, Inc.
Version Merger	Princeton Softech
VIA/Renaissance	ViaSoft
VISION: Assess	Sterling Software (UK) Ltd.
VISION: Builder	Sterling Software (UK) Ltd.
VISION: Excel	Sterling Software (UK) Ltd.
VISION: Inform	Sterling Software (UK) Ltd.
VISION: Inquiry	Sterling Software (UK) Ltd.
VISION: Inspect	Sterling Software (UK) Ltd.
VISION: Recode	Sterling Software (UK) Ltd.
VISION: Redocument	Sterling Software (UK) Ltd.
VISION: Report	Sterling Software (UK) Ltd.
VISION: Result	Sterling Software (UK) Ltd.
VISION: Testpro	Sterling Software (UK) Ltd.
VISION: Transact	Sterling Software (UK) Ltd.
VPS	Levi, Ray & Shoup
Windows NT	Microsoft Corporation
Windows 95	Microsoft Corporation
WinView+	Midcontenient Business Systems, Inc.
X/PTR	Systemware
Y2K	Data Base Architects, Inc.
3M	Minnesota Manufacturing and Mining, Inc.
36/2000 Change Propogation	KDP Software

Summary of Changes

Summary of Changes
for GC28-1251-08
1998-February

This book contains information previously presented in *The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation*, GC28-1251-07.

The following summarizes the changes to that information.

New Information

- Chapter 6, “Testing Techniques for Year2000 Changes” has add a new section entitled: “CMOS Machines which Support the Sysplex Test Datasource Facility” which desribes how a MVS images in an LPAR can belong to a multi-system Sysplex with a date and/or time value which is different from the remaining LPARs on the same physical machine.
- Chapter 6, “Testing Techniques for Year2000 Changes” has add a new section entitled: “Year2000 Testing and the AS/400 System Date” on page 6-18 which describes basis AS/400 functions and recommendations for Year2000 operation and testing.
- Chapter 6, “Testing Techniques for Year2000 Changes” has add a new section entitled: “Year2000 Testing and AIX Systems” on page 6-29 which describes procedures you might find useful when performing Year2000 testing of your AIX System.
- “The IBM COBOL Family for MVS & VM” has add a new section entitled: “VisualAge 2000 Test Solution” on page 8-17 which describes the Year2000-related services available with the IBM VisualAge 2000 Test Solution.

Executive Summary

Chapter 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 can cause 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 might be present on **any** platform (IBM and non-IBM). In recent months, the IT trade press and newspapers have 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. Your household and personal financial situation could encounter a dilemma if your creditors do not also successfully meet this challenge.

Although referred to as the Year2000 challenge and often as the ‘Year2000 Problem’, this is really a 2-digit-year representation problem. Your Information Systems (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; in the banking industry years ago application programs encountered problems with amortization and interest table calculations for long-term mortgages. 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, many computing environments might not function as expected because systems and/or application programs will not process dates later than 31 December 1999 properly.

But I've Been Told...

There are some general misunderstandings and myths regarding the Year2000 challenge. These include:

- This is a problem that occurs only after 31 December 1999.

  In fact, difficulties caused by the Year2000 challenge are already occurring. For example, some applications that deal with future dates, in such areas as mortgages, insurance policies, and driver's licence and credit card expiration dates have already encountered problems.

- This is just a hardware clock problem.

In fact, both hardware and software are affected. The internal timers in hardware might need to be tested for Year2000 readiness and reset. In software, application programs, and operating systems using two digits for year representation could experience difficulties even if the hardware on which they are running has clock and/or system timer services that provide a 4-digit format. And it is a problem that will appear in a variety of programs, including those purchased from Solution Developers, written in-house by your own system and application programmers, licensed from various software vendors, or shared among the IS community.

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

  In fact, any system or program (large or small) can be affected if it uses two digits for year representation.

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. It should. This is not a trivial project. Expect the need for your sponsorship to dedicate personnel and computer resources, approve the requisition of potential Solution Developer tools and/or service support, and provide a budget to fund it all.

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 introduce 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?

Support is currently available from both IBM and many Solution Developers. IBM is committed to addressing the Year2000 Challenge within its own hardware and software product offerings and to assist customers in their efforts.

Detailed information about the broad range of products, service offerings, and other resources that IBM provides is available on IBM's Year2000 Internet Web Site at

`http://www.ibm.com/year2000`

The IBM document (*The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation*), summarized here, is the result of IBM's effort to articulate the overall IBM support with respect to the impending Year2000 transition. It is intended to assist the IS community with planning, methodology, and guidance information needed to convert their application systems to correctly manipulate dates within and between the 20th and 21st centuries.

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.)

But remember, this document is continuously being updated as IBM obtains new information and updates old information.

This document also provides information that addresses:

- **Multiple Platform Solutions** (including IBM product-specific information for):
  - IBM and non-IBM IS platforms
  - 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 document 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 might be at risk if you fail to take timely action. This document provides direction to help your organization to begin a proactive 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 ask, “Now that you spent xxx dollars on this project, what do we have now that we didn't have in 1995?,” you can respond: “We have had an opportunity to take inventory, assess our IT strategy, and make more efficient our IT infrastructure. And more importantly, **we still have a viable business**, one that accepts billing, correctly computes interest and invoices, and calculates our stock dividends. Not all of our competitors can make that same claim!”

The Year 2000 and 2-Digit Dates: Guide

Chapter 2. The Year 2000 - A Transition

Although 2000-January-01[1] 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 2000-January-01, your computing environment might not work as expected because your systems and/or application programs might not properly process dates beyond 1999-December-31. Consider the following scenarios to help put the problem into perspective. 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, 2000-January-01 is represented as 01/01/00 (if using MMDDYY format in your data) and might be interpreted as 1900-January-01 rather than 2000-January-01. This might cause 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 might be encountered, and their variations (listed in Section "Year2000 Exposure Classification" on page 2-2) 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 can be present on **any** (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.[2]

   The challenges of handling the Year2000 changes can 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.

---

[1] Although currently not in popular usage throughout the world, ISO Standard 8601 defines acceptable date formats as yy-mm-dd or yyyy-mm-dd. To apply that standardization and some consistency to this document, when a date is used in a general (non-technical, non-product-specific) manner, the date is written in a similar format (yyyy-month-dd).

[2] IBM recognizes that the 21st century begins at 0000 hours, 2001-January-01, but for purposes of this document, we are defining the 20th—21st century boundary to be between 2400 hours, 1999-December-31 and 0000 hours, 2000-January-01. 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 fact, it is both a hardware and software problem. The problem resides mostly, however, in application programs and data using two digits for year representations.

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 in these examples:

- Incorrect century

  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, 9/9/99, 365/99, or 12/31/99 (31/12/99) 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 representations might have potential exposures. For example, a person with a birthday of 1961-November-10 will be considered to be –61 years old rather than 39 years old on 2000-November-10 if the years 1961 and 2000 are represented by 61 and 00, respectively (that is 00–66=–61). 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:

  In programs where historical dates are used, all events occurring in, for example, 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 years 1700, 1800, 1900, and 1995 were not leap years but the years 1600, 1996, and 2000 are leap years. 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. 2000-February-28 is a Monday and March-01 is a Wednesday, not a Tuesday which is 2000-February-29.

  – Week-of-the-year calculations. The 11th week of the year 2000[3] is March 13 through 19, not March 14 through 20.

- Date fields on hardcopy forms:

  Although not directly a Year2000 exposure, the format used on hardcopy forms to record date data will, in many cases, require change. Examples include, the date field on a bank check which is typically printed as follows:

  _____ 19____

  The century, 19, is pre-printed, and the format for day and month is non-restrictive, thus allowing any day-month ordering and any month indication (numeric, abbreviated or full character spelling, Roman numeral, or any other indication of date provided at the writer's whim). Minimally, the 19____ must be changed or deleted. But the non-restrictive date field only helps propagate non-standard date formats and might cause misinterpretation when the data is later input into a database. Therefore, if you are required to change your stock of forms, consider structuring the date fields they contain to match your proposed database changes.

Scope of Year2000 Transition

The computing environment[4] 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

[3] ISO Standard 8601 provides the following: 1) a week begins on a Monday and 2) the first week of a year is the first week containing four or more days (that is, the first week containing a Thursday).

[4] However, the Year2000 challenge is not restricted to the computing environment alone; it goes far beyond the traditional "computing environment" or "IT structure." Potentially, the Year2000 challenge extends into any piece of machinery that contains a microchip to include elevators, traffic lights, telephone systems, power supplies, cooling systems, security access control, fax machines, photocopiers, and so on. This book focuses on traditional IT software systems, and limits its attention to hardware as presented in the appendices.

- 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, and 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 2-1 on page 2-5 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.

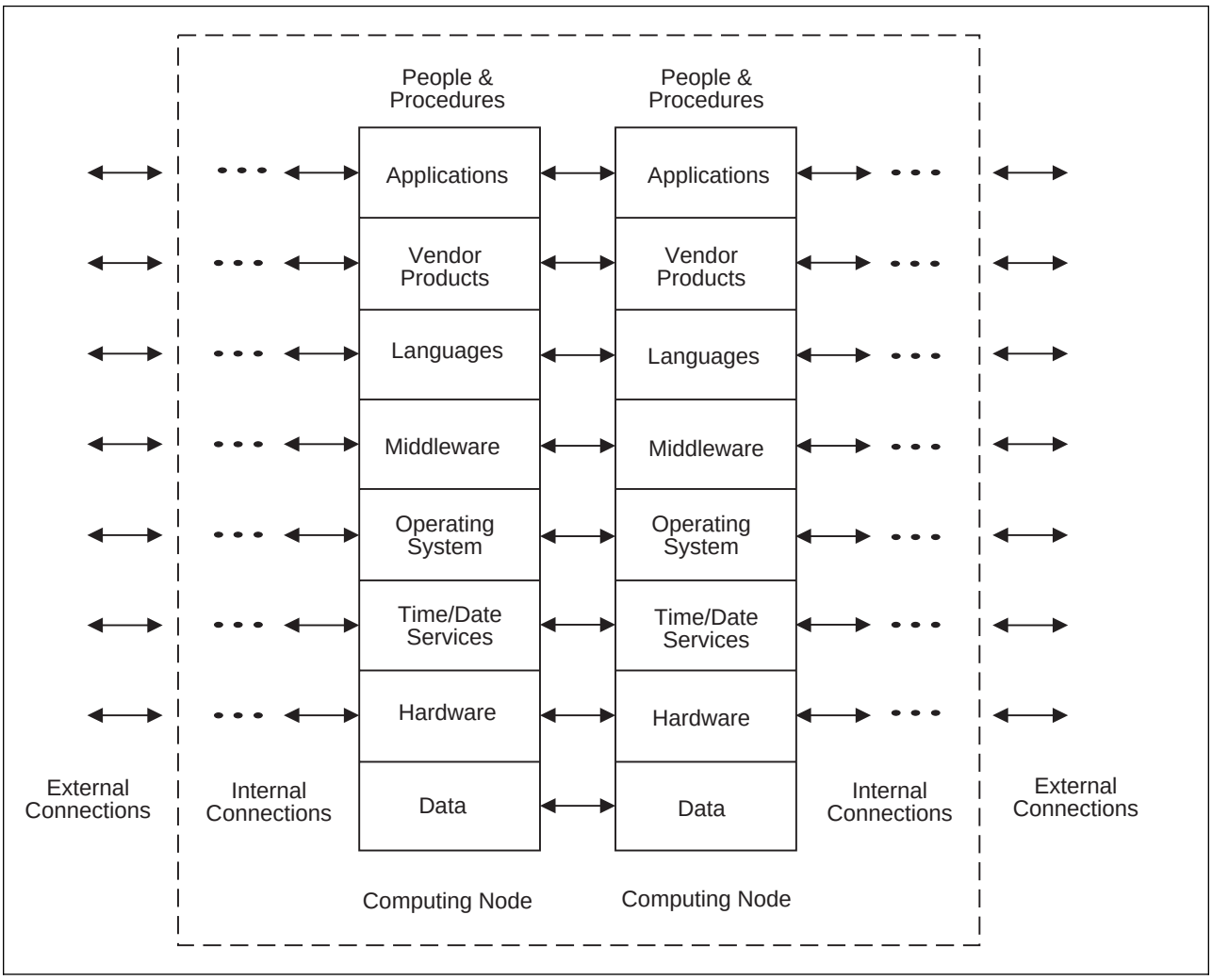

Figure 2-1. A Corporate Networked Computing Environment

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 3. 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 those that follow focus on Year2000 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. (If you have not kept current with the latest technology, your challenge might be greater and your options fewer.) Handling the problem in real time will disrupt your customer services, and the business impact might be significant.
- Most organizations are already short-handed when addressing their current workload and the challenges they face. 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 might 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 experiencing problems today, can reasonably expect problems in the future. For example, insurance companies in 1996, when calculating rates for persons born in ’95, 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.

Time to plan, design, implement, and test your Year2000 transition is limited. Each application is different and many can potentially fail long before the year 2000. Individual applications can have unique thresholds that could fail years earlier. Consider expiration dates on credit cards, drivers' licenses, 3- and 5-year business planning cycles as a few examples. For many organizations, the time remaining might be too short to complete a comprehensive Year2000 project across the entire software and hardware inventory. It is increasingly important to prioritize efforts on critical systems. Also, consider that each day that passes prior to your Year2000 transition allows more data to be added to databases and the potential for

additional routines and programs that are not Year2000 ready to be added to existing system. Also, expertise that is present today might not 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 seriously impact 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. IBM defines **Year2000-Ready** as follows: The capability of a Product, when used in accordance with its associated documentation, to correctly process, provide and/or receive date data within and between the 20th and 21st centuries, provided that all other products (for example, hardware, software, and firmware) used with the Product, properly exchange accurate date data with it.

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 improve overall business efficiency by taking inventory, accessing your IT strategy, and making more efficient use of your IT infrastructure. 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
   - 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 5-14 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.

Inventory Your Hardware Systems

Contact hardware manufacturers regarding the Year2000 readiness of hardware components of your system.

Chapter 4. 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 element of your IS organization. (Of course, the scope of the Year2000 challenge is far greater than that contained in your IS center, but this book has a focus mostly limited to traditional software programming.) 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 Appendix B, "Year2000-Ready Solution Developer Products" on page B-1 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 2000-January-01. 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 you vary 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 generally 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 your migration criteria.

   Refer to Chapter 6, “Testing Techniques for Year2000 Changes” on page 6-1 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:

- Low Impact
  - The program uses a 4-digit-year representation in all occurrences.
  - The program uses a 2-digit-year representation within itself, 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 itself, but does not have internal exposures. The 2-digit year **is** externalized, but can not be referenced (for example, for display-only) by any other 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 would the clerk compiling a list of 6-year-old children in 1995 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.
- High Impact
  - The program uses a 2-digit-year representation within itself. 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 data (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 5, “Reformatting Year-Date Notation” on page 5-1) to reformat these date and time representations.

Miscellaneous Date-Related Issues and/or Exposures

Various programming techniques and practices that relate to the use of dates and date fields might also require your consideration when identifying potential exposures. Such areas of concern are discussed below.

Support for Expiry Dates

For compatibility with existing function, to prevent accidental data loss, and the common usage of dates to indicate 'never expire', IBM products will continue to respect non-expiry dates, such as 99.365 and 99.366 as indicated here:

- **MVS and OS/390** - If an expiration date is stored as the last day in 1999 (whether 99.365, 1999.365, 99.366, 1999.366, or any of the packed decimal forms - mm/dd/yy or mm/dd/yyyy - 12/31/99 or 12/31/1999), the data set is considered as a "never-expiring" data set.

  Data sets with their expiration dates set to "never expire" can be deleted in the same manner as deleting a data set whose expiration date has not yet occurred; for example, using the PURGE parameter with the DELETE command.

  Various components of the operating environment minimize any impact to your system if you accidentally code 99.365 when you did not intend "never expire." For example, if you create a new data set using JCL, and you inadvertently provide a retention period (RETPD) number of days that when added to the creation date is 1999-December-31, MVS changes the expiration date to 2000-January-01.

  When using System-Managed Storage, the management class stores the retention period and the creation date rather than calculate the precise date. This permits code in DFSMShsm (and formerly DFHSM) to allow data sets to have an expiration date of 1999-December-31, without being confused with the "never expire" value. Additionally, SMS permits specification of NO LIMIT to indicate that this data set does not expire and can only be deleted if special conditions are followed, such as using the PURGE parameter with the DELETE command.

  When using DFSMSrmm (or DFRMM) to manage tape data sets, there are variable options within the defined policies to control data set expiration. Therefore, it is possible to determine whether to continue to recognize 99.365, 99.366, 1999.365 and 1999.366 as never expire dates or to use new indicators in the control data set (CDS) to determine if the data sets should be allowed to expire.

  The current MVS and OS/390 environment allows you to continue to treat 99.365 (and its variants) as a "never expire" date while components such as SMS and DFSMS are moving to the use of flags to specify this condition.

- **VSE/ESA** - BAM, VSAM, and utility processing that build file creation dates and expiration dates and check for file expiration is extended to include the century field in the determination of the current date.

  VSE/ESA will continue to respect files with an expiration date of 1999/365 as never expire, **except** when the date was calculated from a retention period specification. That is, if the file creation date plus the retention period equal 99/365, VSE/ESA will expire the file on 1999-December-31.

- **VM/ESA** - Standard tape label processing supports the use of 99365 and 99366 as never expire dates. When a tape containing a label with one of these dates is mounted, an UNEXPIRED TAPE warning message is issued. The user can then either respond IGNORE (write to tape) or ERROR (do not write to tape).

  For expiry dates of 99365 and 99366, the UNEXPIRED TAPE warning will continued to be issued even after 1999-December-31.

Language Environment and Date Simulators

Attention:

Be aware that Date Simulators (as marketed by Solution Developers) should only be installed on a test version of DB2 (DATABASE 2) and Language Environment. These tools corrupt a module in both the DB2 and Language Environment products. This is acceptable for your Year2000 testing purposes, but the product is no longer warranted by IBM after you install such Date Simulator products. Therefore, install a separate copy of DB2 and Language Environment or both for use in Date Simulation testing than the copy used for production or final 'Time Machine' testing.

Chapter 5. 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; however, IBM is not aware of particulars regarding your unique system environment or external influences.

IBM is not responsible for your results or consequences of your implementation.

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 4, “Identifying 2-Digit-Year Exposures” on page 4-1, 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 might 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 5-13 for more details on bridge programs. You should accomplish this program and data conversion in steps. Otherwise, you will likely encounter immediate 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 and is otherwise of little functional value. 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 might 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 currently considered to be the **only** complete, long-term, 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 (as described in Appendix D, “Glossary” on page D-1 in the description for `integer date`) can potentially produce ambiguous data and errors. This can occur because each integer-date scheme employs a unique starting date. Therefore, avoid mixing incompatible integer dates.

Solution #2: Compressed Date Data

The **compression** solution externalizes a 4-digit-year format by manipulating (prefixing or packing) 3- or 4-digit-year dates in limited space. These techniques require changes to both your data and your programs.

This section presents several specific examples, many others exist and might prove more applicable to your specific needs. These techniques are presented to show alternative methods of “squeezing” more data into existing space.

CAUTION:
Apply this approach with caution. Be certain that the new compression scheme does not affect the proper functioning of your programs after all the data changes are implemented.

Some examples include:

Unsigned Packed Decimal Technique

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.

Signed Packed Decimal Technique

Convert the data type from the 6-byte character representation of the date field (YYMMDD) to a 4-byte packed decimal (eight digit) representation. As shown in Figure 5-1 on page 5-4, this is a 2-step process.

Step **1** - use the assembler PACK instruction (or equivalent) to convert the date from character to packed (or compressed) format. The PACK instruction

pads the unused high-order bytes with leading zeros and places the positive sign indicator (X'C') in the low-order right half byte.

Step **2** - append the century indicator value (CC) in bytes 2 and 3. For example, append X'F1F9' for “19” or X'F2F0' for “20” to the original 2-digit (YY) value to produce a full, 4-digit (CCYY) date. High order byte 1 remains X'F0F0'.

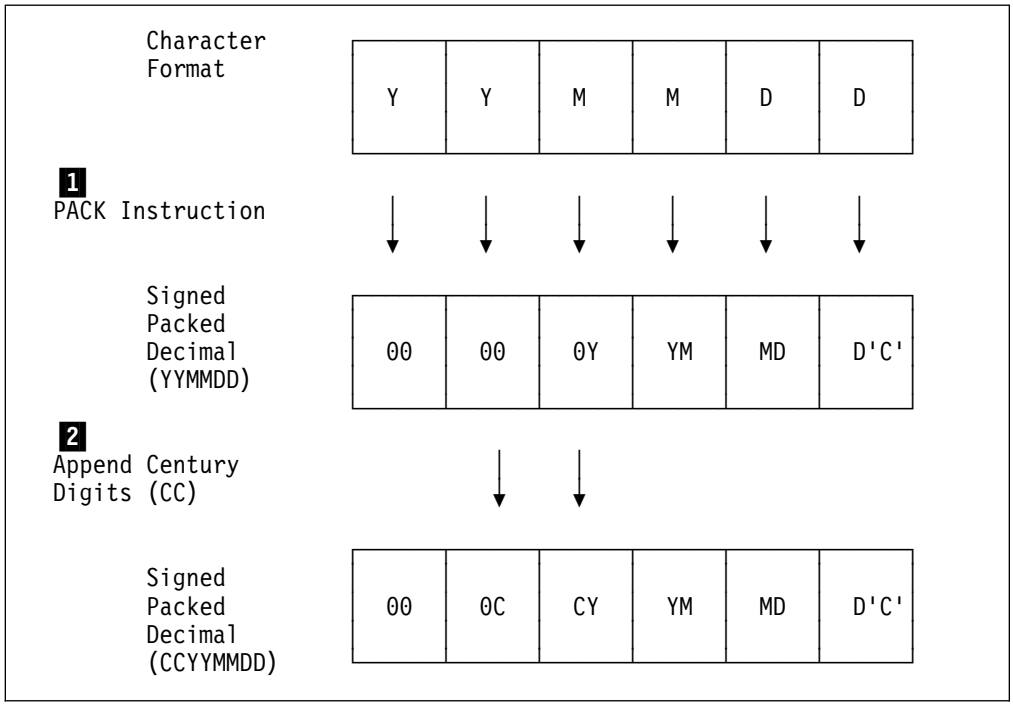

Figure 5-1. Using a 6-Byte Data Field to Contain 8-Digit-Date Data

A main advantage to this method is that the size of the data field remains unchanged at 6 bytes. And, of course, by preserving the low-order sign indicator, you allow arithmetic manipulation.

You must now rework those programs that reference this data to interpret it correctly. To distinguish character data from packed decimal data, your routine can examine the first byte; a value of X'00' indicates “packed decimal,” otherwise the routine can assume the field contains character date data (a 2-digit year date).

Append a Century Indicator

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 5-1).

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. Programs that use this scheme do not need to access the output of the 2-character conversion simultaneously. Programs can be updated to handle the new and old date formats one at a time. When all programming considerations are made, you can then update your database to the new compressed format at a time convenient to other year2000 projects.

   Consider converting individual programs when each is due of maintenance while your dedicated year2000 team continues on other aspects of the conversions.

2. 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) or the need to change record sizes and JCL.

3. Can distinguish years from different centuries using the 2-character-year format.

4. Sort algorithms do not require change.

5. 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.

6. 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, you might experience incorrect data sequencing if you do not add further programming logic.

2. Your choice of compression technique might prove incompatible when moving or sharing data across different platforms. For example, packed decimal, although useful on a mainframe system might cause incompatibilities when used on other smaller systems such as a personal computer.

3. You must convert the data before it can be displayed in Gregorian format. You can, of course, read the date date in a hexadecimal dump without conversion.

Solution #3: 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 1961-January-01 to 2060-December-31, 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 1996, 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:

- $xy \geq 61$, then it is a 20th century date (19xy)
- Otherwise (that is, $xy \leq 60$), 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 in 1997 your application can successfully deal with dates in the range 1962 through 2061, you need to adjust this program checking every year. The inherent future risk when employing this technique should be obvious, and when compared to the sliding window technique is far less desirable.

Furthermore, you will likely need to use a different 100-year window for each application because the date context varies among them. In so doing, you are adding one more level of complexity to your overall programming environment - that of potentially losing track of when each application must be adjusted or which applications require yearly window adjustment.

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 a 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 might 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 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.

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 5-2 on page 5-8 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.

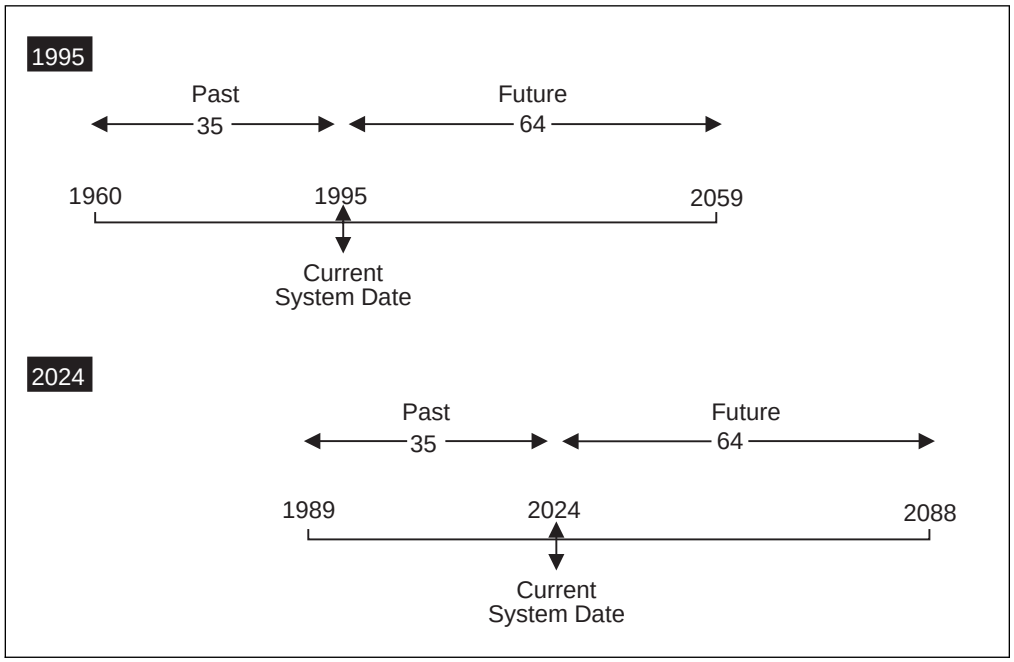

Figure 5-2. Graphical Representation of the Sliding Window Technique. (Using two current system dates, 1995 and 2024, as an example.)

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's 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 8, “Tool Categories and Available Tools to Ease Year2000 Changes” on page 8-1. For an example of how DFSORT/MVS implemented a sliding window to sort, merge, and transform 2-digit year data to 4-digit year data, refer to “DFSORT” on page 8-25

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 #4: A 2-Digit Encoding Scheme

The **encoding** solution 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 include **encoding** 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 generally considered to be the least desirable approach. 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 two decimal digits to two hexadecimal digits. Two hexadecimal digits allow the ability to represent numbers up to 255; therefore, providing the ability to represent dates between 1900 and 2155, for example:

  ```
  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 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 5-3 on page 5-10 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 5-9 by 1040 more years. For example:

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

Figure 5-3. 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

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. Your choice of encoding technique might prove incompatible when moving or sharing data across different platforms.

4. 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.
5. Depending upon the choice of data representation you implement, you might experience incorrect data sequencing if you do not add further programming logic.
6. 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.
7. 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 might have Year2000 exposures of their own: for example, the routine(s) might only provide a 2-digit-year format. Fixing the exposed date/time routine(s) is one possible solution. Selecting a vendor date/time routine that is 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 might 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 4-3, potential 2-digit-year exposures can be classified into two categories (low impact and high 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 when assessing how to address each application. You can:

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

Certainly, most IS organizations will build their Year2000-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 #3 (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** (compression techniques) to a certain module, another module that receives data from this module might need to apply Solution #2 as well to maintain data consistency with the data representation scheme. Another alternative is to apply Solution #2 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.

  Further, if the date data is stored in packed format, you must be certain to unpack that data before it can be accessed and manipulated appropriately. The constant process or packing and unpacking vast quantities of data could adversely affect data access and performance.
- **Solution #3** (windowing techniques) to a certain module, another module that receives data from this module might either receive 2-digit-year data as before and apply Solution #3 itself or receive 4-digit-year data and apply Solution #1 (full 4-digit solution) for its own exposure removal.
- **Solution #4** (encoding techniques) to a certain module, another module that receives data from this module might need to apply Solution #4 as well to maintain data consistency with the data representation scheme. Another alternative is to apply Solution #4 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 1999-December-31 (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 likely fail or give unpredictable results after 1999-December-31.

- 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 (1999-December-31 to 2000-January-01) or a single week (1999-December-25 through 2000-January-01) 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 currently published standards such as those 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 propagating 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 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 all other years that are a 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 (low-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 5-1) 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 ability to recognize and correctly manipulate both 2- and 4-digit-year formats, thereby providing 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 (compressed date data that externalizes a 4-digit year format)
     - Has potential exposures of data sequencing if you do not add further programming logic.
     - Should be used only when expanding the date-data field is too costly and when the need to preserve your current date data field size is critical. (Remember, it was the need to preserve storage space that was a major factor that caused today's Year2000 challenge.)
   - Solution #3 (windowing techniques that externalize both 2- and 4-digit formats)
     - Considered a temporary solution and **should only be used when Solutions #1 or #2 are not practical**. (This is a debatable 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 program will be phased out before any exposure occurs.

- Solution #4 (encoding scheme that externalizes 2-digit format)
  - 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 too costly and the function of the software program will be phased out before any exposure occurs.

Chapter 6. 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. There are other testing techniques. You should choose whatever techniques are best for your situation.

Structural Testing Techniques

Structural testing enables sufficient testing of a function's implementation and helps to determine that all structures of the system are integrated to form a cohesive unit. Structural testing techniques include:

Operations Testing

Apply operations testing to determine whether the system is ready for normal 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 enables, prior to production, your IS staff to 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 rates, 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 #4 (refer to “Solution #4: A 2-Digit Encoding Scheme” on page 5-9) 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 5-1) is applied and expands the year data field from two to four digits.

Recovery Testing

Apply recovery testing to enable the system to 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 confirm 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 enable these requirements to be 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 confirm 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 verify 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 confirm 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 their very 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 **extremely careful** 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.

Attention

The Most Vulnerable Resources/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.

Alternatively, or additionally, a complete Year2000 test set should include leaving date-dependent functions on and using “aged” data that will not cause loss of access and expiration problems. When you implement your new Year2000-ready system in a production environment, that is not the time to introduce new, untested system or application code.

Ensure that you do not contaminate your production system or production databases when running various test scenarios. Be sure that data is not unexpectedly deleted from the system. For example, if your system is set up to scratch all files that are 1-year old, all files that are created prior to 1999/01/01 will be scratched when the system clock is change to 2000/01/01 or later. Because timestamps are saved when new data is created, be sure to remove data created before moving the system clock back to an earlier time. Unpredictable results might occur because programming might not be in place to handle normally improbable conditions, such as when the creation date of a data set is later than the current time.

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 OS/390 or 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 OS/390 or MVS:**
     - **Year2000 Testing within a Single-System Sysplex Environment**

       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. You must include `,GMT` on your response to ensure that the clock is set correctly; issue: `CLOCK=hh.mm.ss,DATE=yyyy.ddd,GMT`. (Do not set the time of day (TOD) using the Sysplex Timer.)

     - **Year2000 Testing within a Multi-System Sysplex Environment**

       Note: Testing your system using the following test techniques is seldom required; you need to do this testing only for those applications that you consider truly multisystem (that is, those applications which you cannot test under a single-system Sysplex image).

       To set up a multi-system Sysplex Year2000-test environment on bi-polar and CMOS machines (prior to the G3 machines) requires that you dedicate at least an entire physical machine to the test system. **The Year2000 test machine cannot include LPARs which contain members of a different Sysplex with a different date.** This is

because it is not possible to set manually different dates (and times) in the several LPARs which are accurate enough to allow the Sysplex to function correctly.

There are two ways you can set a future date for the test Sysplex.

a. **Using a Sysplex Timer**

   Set the date ahead on a dedicated (to the test Sysplex) Sysplex Timer and then IPL your Sysplex images. The images, which are part of the Year2000 test Sysplex, can be on a single physical machine or on multiple physical machines, but you cannot include Sysplex images on these machines which are not in the Year2000 test Sysplex. Be sure to set the ETR network ID on this Sysplex Timer to a value different from the ETR network ID used by other Sysplexes.

   For more information on Sysplex timers as related to Year2000 testing in an S/390 environment, refer to: *S/390 Time Management and IBM 9037 Sysplex Timer* (SG24-2070).

b. **Without Using a Sysplex Timer**

   - **Bi-polar Machines:**

     If the Sysplex images all reside on a single physical machine, you can can set the Process Controller clock to the future date using the SYSDEF frame.

   - **CMOS Machines Prior to Microcode Level EC F10640**

     Set the service element clock to the future date using the “Service Element Option” menu on the hardware master console.

   Then Power_On_Reset the machine, activate PR/SM, and IPL your Sysplex images with CLOCKxx parmlib members which specify SIMETRID=xx.

   Note: Be certain that the SIMETRID is different from the ETR network ID used by other Sysplexes.

– **CMOS Machines which Support the Sysplex Test Datesource Facility**

The Sysplex test datasource facility is provided by CMOS machines (G3 and later) at microcode level EC F10640 or higher. In addition, MVS APAR OW28604 is required for releases of MVS and OS/390 through OS/390 Version 2 Release 5.

The Sysplex test datesource facility allows you to set up a multi-system Year2000 test Sysplex where you do not need to dedicate the entire machine to that test Sysplex. This facility allows you to:

- define a group of LPARs with the same time and date values
- specify the time and date values for the LPAR group.

As a result, the MVS images in these LPARs can belong to a multi-system Sysplex with a date and/or time value which is different from the remaining LPARs on the same physical machine. For more details on how to set up this facility, refer to *PR/SM Planning Guide*

(GA22-7236) and *S/390 Time Management and IBM 9037 Sysplex Timer* (SG24-2070).

Once you have defined the LPARs to be “test datesource LPARs,” IPL your Sysplex images with CLOCKxx parmlib members which specify SIMETRID=xx.

- **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).

  If you change the system date, unless you have applied APAR VM60324 (this APAR is available for Version 1 Release 2.2 and Version 2 Release 1), issue a SHUTDOWN REIPL command immediately after completion of the system IPL as indicated in message HCPITM1161I (this message is displayed only on Version 2 systems). 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.

  For more information on VM/ESA as related to Year2000 testing, refer to: *VM/ESA Year 2000 Migration - A Case Study* (SG24-2042).

- **On VSE/ESA:** 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 OS/390 or 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 8-79 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 OS/390 or MVS and control block references.

Testing on AS/400 Platforms

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.

Refer to “Year2000 Testing and the AS/400 System Date” on page 6-18 for a more complete description of AS/400 function and test recommendations.

Testing on PCs

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 the IBM Product Year2000 Database at URL: `http://www.ibm.com/year2000` or contact the IBM Technical Support Center (TSC) for your geography for PC-specific, internal clock setting information. (See “About This Book” on page ix for TSC address and contact information.) Some suggested scenarios for testing for Year2000-readiness of your PC system clock follow:

Be certain to heed relevant cautions as noted in “How to Change Date and Time for Testing” on page 6-4 before proceeding with resetting a system clock.

- 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

Testing the Year 2000—or, An IBM Excursion Through Time in the OS/390 Parallel Sysplex Environment

A Word About This Section

This section on IBM testing in the OS/390 environment has been extracted, with very minimal editing, from *OS/390 Parallel Sysplex Test Report*, GC28-1963. That document also details many other aspects of OS/390 parallel Sysplex testing, but the following offers some very specific advice and findings as IBM conducts its internal Year2000 testing. Even if your environment differs, you might find that this section provides some useful insight into test procedures and methodology, based on actual test findings.

The *OS/390 Parallel Sysplex Test Report* documents the efforts of a team of IBM S/390 testers and system programmers simulating a customer production environment. Written from the perspective of a system programmer, this 1996-December report describes cross-product and integrated testing in a Parallel Sysplex. The following excerpt regarding Year2000 testing is taken directly from that report.

As much as many of us in the world of computing would like to have the year 2000 postponed, or even cancelled altogether, it looks like it will occur as scheduled, and we'll just have to deal with it.

Having faced this reality, we'd like to share with you our approach to Year2000 testing, and what it was like venturing into the future and returning safely to the present.

The impact of the changeover to the year 2000 is a serious matter. While we focused mainly on system software, the brunt of the impact will be to applications (see “Year2000 impact” in Figure 6-1 on page 6-11) and will be unique to those applications. ***We cannot stress enough how important it is that you create a Year2000 project that is staffed with adequate resources to position your enterprise for this transition.*** While you can take advantage of many services that are available, you still must recognize the challenge and participate in the effort. If you have not yet begun to size this effort and position yourselves, we strongly recommend that you begin as soon as possible.

How Our Environment Might Differ from Yours

As you read about our experiences, keep in mind that our environment might minimally be different from yours in the following ways:

Figure 6-1. How Our Year2000 Test Environment Might Differ from Yours

Experience Related To	Details
Year2000 impact	***Our focus was almost entirely on system software***, but we realize that ***Year2000 impact is primarily on applications***. We reviewed the applications we use and found them to be Year2000 compliant—not much for us to share. But remember that as customer-like as we try to be, we are still a test organization. ***The impact on application programs in your environment will be vastly more dramatic and will require much more of your focus.*** Fortunately, there are publications, Year2000-ready software products, and services available to help you with this effort. See “Preparing for Year2000 Testing” for more information.
Required outages	We did not dynamically move to a Year2000 environment—we had to take some outages. Even though one of our main objectives is to provide end-user availability, in this case we had to relax that objective. Because we are a test organization, our production systems are also test systems. You won't be doing Year2000 testing on your production systems, so this won't be as much of an issue for you.
Additional Sysplex Timer	We normally share our Sysplex Timer with many other systems and Sysplexes used by other organizations on our test floor. For Year2000 testing, we obtained another Sysplex Timer that we could use exclusively. Depending on how you set up your test environment, you might not have to do this.

Preparing for Year2000 Testing

A good way to start preparing for Year2000 testing is to consult *The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation* (that is, this book). The following are some of the topics you will find there:

- Identifying 2-digit-year exposures
- Solutions and techniques for reformatting year-date notation
- Testing techniques for Year2000 changes
- Tool categories and available tools to ease Year2000 changes
- IBM consulting and services, including information about a comprehensive set of Year2000 solutions that takes into account applications, systems software, and hardware in both centralized and distributed environments. This comprehensive solution set was developed by IBM's Information Systems Solution Corporation (ISSC).
- IBM Year2000-ready key program products and hardware (lists key IBM products and offerings that are available as Year2000-ready, including versions, releases, and PTF numbers where appropriate).
- Year2000-ready Solution Developer products (a list of product names provided to us by their respective companies; IBM hasn't tested these products, but you

can check directly with the companies to get the specifics of their Year2000-readiness implementation).

- A bibliography of both IBM and non-IBM publications that address the Year2000 subject.

Major Concerns: As we studied how to go about our testing, our concerns fell into 2 major categories:

- How should we position ourselves to travel forward in time to the year 2000? (Described in "Preparing to Travel to the Year 2000.")
- Perhaps more frightening, how could we make sure we'd get back safely to the year 1996? (Described in "Preparing to Travel Back to the Year 1996" on page 6-13.)

Key Questions: Some key questions you might have that we'll answer here are:

- Do you have to isolate the DASD used by your Year2000 systems from DASD used by other systems? The answer is ***yes***, and we explain below how we handled that (see the row labelled "Isolating DASD" on page 6-12 in Figure 6-2 on page 6-12).
- Do you have to isolate your Year2000 test networks from other networks? The answer is ***not necessarily***, depending on the applications you are running. We explain below why it was not necessary for us to isolate our test networks (see the row labelled "Network considerations" on page 6-13 in Figure 6-2 on page 6-12).

Preparing to Travel to the Year 2000: Here's how we positioned ourselves prior to resetting the clock:

Figure 6-2 (Page 1 of 2). Preparing to Travel to the Year 2000

Experience Related To	Details
Preliminary planning	• We consulted *Year2000 and 2-Digit Dates: Guide for Planning and Implementation* (that is, this book). • We made sure our applications were Year2000 compliant. • We made sure we had Year2000-ready software installed, including the applicable service. Here are some additional APARs/PTFs you will need that are not listed in the year 2000 guide: – IMS APAR PN89281—enables the use of IMS utilities (such as recovering a database) in the year 2000. – IMS APAR PN89694—the PTF for this APAR will contain various Year2000 fixes, including the fix for an emergency restart problem. – LAN Server for MVS APAR PN91131—allows LAN Server for MVS to start in the year 2000. – SDSF PTF UN96995—enables SDSF to correctly display the date in YYYYDDD format when viewing the OPERLOG.
Isolating DASD	We developed a plan to completely isolate our Sysplex DASD from all systems outside our Sysplex. We normally share 192 3390 Model 3 DASD volumes with other systems (that's in addition to the roughly 550 volumes of private DASD we have). We did the following: • Reviewed all the data on those 192 volumes to determine exactly what data we required. • Determined the subset of DASD volumes needed to contain the required data and obtained that number of volumes for our private use. • Copied the required data from the shared volumes to the private volumes. We did this off-shift, because we had to bring down our workloads.

Figure 6-2 (Page 2 of 2). Preparing to Travel to the Year 2000

Experience Related To	Details
Network considerations	We made a conscious decision ***not*** to isolate our test TCP/IP and SNA networks from our IBM global network. We recommend you make a conscious decision, one way or the other, about whether to isolate your test networks. To make the decision, you should review each application that communicates across the network to determine if data containing dates might flow from your Year2000 systems back to systems that are in the present. You can think of your Year2000 systems as a black hole into which date information can flow, but out of which you do not want date information to escape. Otherwise Year2000 data could corrupt your production environment. In our case, we found that one-way communications, such as remote logon to the Sysplex, sending files to the Sysplex using ftp, and NJE were the type of applications we were running across the network, and these applications would not cause a problem. Our decision not to isolate our test networks turned out to be correct—we did not experience any network-related problems.

Preparing to Travel Back to the Year 1996: To make sure we could get back to the year 1996 and protect ourselves from data loss, we did the following ***before setting the clock ahead***:

Figure 6-3. Preparing to Travel Back to the Year 1996

Experience Related To	Details
Dumping DASD	We set up a special dump class and used DFSMShsm to dump all our DASD.
Copying control data sets	We made copies of the DFSMShsm control data sets so DFSMShsm could use those copies to restore all our DASD volumes once we were back in 1996.

Turning off automatic space management	We turned off DFSMShsm automatic primary and secondary space management to avoid migrating or deleting any DASD data sets that would expire in 1999. We did so by changing YYYYYYY to NNNNNNN on the following statements in our ARCCMDxx parmlib member:
	
	`DEFINE PRIMARYSPMGMTCYCLE(NNNNNNN`
	`  CYCLESTARTDATE(1996/05/13))`
	`DEFINE SECONDARYSPMGMTCYCLE(NNNNNNN`
	`  CYCLESTARTDATE(1996/05/13))`
	
	Note: We did run space management manually using selected test volumes.

Recycling tapes	We did not automatically recycle any of our tapes while the clock was in the year 2000, to avoid losing any expired data sets that were on tape. **Note:** We did manually recycle selected tapes.

Stepping into the Future

With all the necessary precautions taken, the following are the Sysplex-related steps we followed on specific dates and times:

Figure 6-4. Dates/Times for Year2000 Testing

Future Date (Time)	Details
1999-December-28	First we set the clock on our Sysplex Timer to 1999-December-28 so we could spend a few days in the year 1999 before going to the year 2000. This way we could: • Generate some data in the year 1999. • Make image copies of our IMS and DB2 databases in the year 1999. • Test our migration off of the shared DASD to our own private DASD. We set the clock with a 12-hour time difference, so that 12:00 midnight in the future would actually be 12:00 noon in the present (most of our team members prefer to come to work during the day, and nobody wanted to miss the New Year's Party we had planned). Here's what we did to set the clock: • Varied ***every*** system out of the Sysplex. • Reset the time (to 1999) on the Sysplex Timer. (Remember, this is our own private Sysplex Timer, not shared with any other systems.) • IPLed 1 system and replied 'I' to message IXC420D (to initialize the Sysplex). • IPLed the remaining systems in the Sysplex. This is one of those outages we had to take that you should not. We sent our entire multi-CPC Sysplex into the year 2000; the only way to do that is to reset the Sysplex Timer, and that requires a Sysplex-wide IPL. An alternative approach for you might be to set up a Sysplex for testing purposes using only LPARs on 1 CPC. Then you would not even need a Sysplex Timer—you could just change the time on that CPC.
1999-December-31	On 1999-December-31, it was time to bring out the balloons, hats, and noisemakers, and blast a tape of Prince's "Party Like It's 1999" (none of us could find any Guy Lombardo tapes). We had all of our workloads running, which included 9000 OLTP users running applications that were doing IMS, DB2, and VSAM RLS data sharing. We also had 40 OpenEdition users running the spectrum of our OpenEdition workloads, and about 5 network clients running client/server workloads. (See "Year2000 Networking and Application Enablement Experiences" below for additional experiences related to networking and OpenEdition services.)
1999-December-31 (11:30 PM)	At 11:30 PM, we quiesced the DB2 and IMS subsystems and the CICS regions on system J80 and brought them down cleanly.
1999-December-31 (11:45 PM)	At 11:45 PM, we crashed the DB2 and IMS subsystems and some of the CICS AORs and TORs on system J90.
1999-December-31 (11:59:50 PM)	At 11:59:50 PM, we started the countdown and lowered the ball (well, it was an apple on a string attached to the ceiling).
1999-December-31 (12:00 midnight)	When the clock struck midnight, we made a lot of noise, but then immediately checked to make sure everything that ***had been*** up and running ***was still*** up and running—everything was.
2000-January-1	Here's how we rang in the New Year: • We restarted the crashed IMSs, DB2, AORs, and TORs on system J90 to be certain they would recover in the year 2000 from 1999 log data. We experienced no problems. • We brought up the quiesced IMSs, DB2s, AORs, and TORs that had been brought down cleanly on system J80. Again we experienced no problems.

Thankfully, the changeover to the year 2000 turned out to be fairly uneventful for us. But we stress again that these experiences relate primarily to system software; you must examine all aspects of your computing environment, especially your application suite, and plan to test accordingly.

Year2000 Networking and Application Enablement Experiences

We include the following networking and application enablement experiences in this section to give you a total picture of our Year2000 testing.

Figure 6-5 (Page 1 of 2). Year2000 Networking and Application Enablement Experiences

Experience Related To	Details
PC clients	Note that for our applications, PC clients are able to communicate with a Year2000 host system whether or not we change the date on our PCs.
LAN Server for MVS and its NFS	We tried to run both DFSMS/MVS NFS and LAN Server for MVS NFS. We had a problem with LAN Server for MVS not being able to start in the year 2000. You need the fix for APAR PN91131 to solve this problem.
LANRES and NetWare	We tried changing the date and time on our NetWare clients to the year 2000. They refused to stay in the year 2000, but kept returning to 1996. We then realized that NetWare has a time synchronization function. When you have just 1 NetWare server in your configuration, you are subject to the single reference time server, whereby all NetWare clients synchronize their clocks to the NetWare server. Once we changed the date and time on the NetWare server, we rebooted the clients and they synchronized with the server.
OpenEdition services—DCE and conflicting time	We had a problem with one of our RS/6000 workstations when we tried to change its date to the year 2000. That workstation is configured in a DCE cell, and is also configured in 2 different LANs (our own local LAN, and a regional LAN). The regional LAN was still in 1996. This created a conflict. If you find yourself in this situation, we recommend you isolate your Year2000 test system from any systems that are still in the year 1996 to avoid problems synchronizing time in the DCE cell. (Note that this case is different from the one we describe in “Preparing to Travel to the Year 2000”—see the row labelled “Network considerations” on page 6-13 in Figure 6-2 on page 6-12).
OpenEdition services—AIX level	We were unable to change the date on another of our RS/6000 workstations because that machine happens to be running AIX 3.2.5 (for internal reasons we did not install the service on AIX 3.2.5 that would bring it to Year2000 compliance). The Year2000 support is in AIX 4.1.3.

Figure 6-5 (Page 2 of 2). Year2000 Networking and Application Enablement Experiences

Experience Related To	Details

OpenEdition services—2-digit displays	Some OpenEdition messages continue to display with 2-digit years. 2-digit year displays do not cause a problem as long as they are ***display only*** (not used in calculations or sorts). However, if you prefer your displays to contain 4-digit year notation, we found a way to code time and date formatting using the locale specification. The default POSIX locale specifies a 2-digit year when formatting the date with %x. Here are instructions to create a user-defined locale named CENTURY that will format the date with 4 digits for the year with %x:
	• In the OpenEdition shell, issue the following commands:
	`cp /usr/lib/nls/localedef/C /usr/lib/`
	`nls/localedef/CENTURY`
	`oedit /usr/lib/nls/localedef/CENTURY`
	• In the edit session, change lowercase “y” to uppercase “Y” as highlighted in the following statements:
	`% appropriate date representation (%x)`
	`"%m/%d/% Y "`
	`d_fmt "<percent-sign><m><slash><percent`
	`-sign><d><slash><percent-sign>< Y >"`
	If you want to keep the timezone in the display, you must also add the text highlighted in the lines below:
	`% appropriate date and time representation (%c)`
	`"%a %b %e %H:%M:%S %Z %Y"`
	`d_t_fmt "<percent-sign><a><space><percent`
	`-sign><b><space><percent-sign>/`
	`<E><space><percent-sign><H><colon><percent`
	`-sign><M>/`
	`<colon><percent-sign><S><space><percent`
	`-sign> <Z><space><percent-sign> <Y>"`
	• Issue the following command to build the new locale (you issue the command in one continuous line, but we broke it up here):
	`localedef -f /usr/lib/nls/charmap/IBM`
	`-1047 -i /usr/lib/nls/localedef/CENTURY`
	`/usr/lib/nls/locale/CENTURY`
	• Add the following statement to /etc/profile:
	`export LC_TIME=CENTURY`

Spending Time in the Future

We spent about a month in the future doing the following:

- We did forward recovery from the image copies taken in 1999 for IMS and DB2 databases.
- We ran our normal production workloads. That included IMS, DB2, and VSAM RLS data sharing, along with OpenEdition and DCE workloads. We also ran our normal RMF Postprocessor reports, which invoke SORT and use SMF records (containing timestamps).
- We tested Leap Year by moving the date to 2000-February-28 to see what would happen when the date changed over to February 29. Every subsystem and component we checked seemed to know it was February-29 and not March-1.
- Then we set the date to 2000-December-29 and ran for a couple of days to see what would happen when the date changed to 2001-January-1. Again, we experienced no problems.

Back to the Present

Once we completed our testing in the future, we had to restore our computing environment to the present. Depending on how you conduct your testing, you won't have to restore to the present. You might avail yourself of external services and create your test environment off-site; once testing is completed, you just walk away. Or you might have a process whereby you can create a new test bed whenever you need to. If so, you could complete your Year2000 testing just as you would any other set of test scenarios, and then start over for the next required test.

If you have a test bed that you treat more like a production environment, such as we have, then you might need to restore it. Our recommendation is that you ***avoid attempting to restore back to an earlier date***, if at all possible, to simplify your testing and minimize your risks. In fact, if you have a contingency plan to return your systems to an earlier date once the year 2000 actually occurs (if, for example, you encounter problems so severe that you think returning to an earlier date will allow you to keep running while you debug the problems) we would caution you that such a contingency plan is not foolproof. You are much better off devoting the resources ahead of time to adequately address this testing so that you will not encounter severe problems once the date arrives.

Having said that, we'll tell you what we did to return to the present:

- We printed any output we needed from our JES2 and JES3 spools and ensured no output that we needed remained on those spools (that way we could cold start our JES2 and JES3 subsystems without having to do a spool offload/reload).
- We reset the clock.
- We brought up a single system with only MVS, JES, and DFSMShsm. (We didn't bring up any other subsystems or attempt to process any workloads yet.)
- We started restoring data. Remember we had dumped all our data before we set the clock ahead; after we set the clock back, we restored ***most*** of our data using the DFSMShsm full-volume restore function. Here's what we ***did not*** restore:
  - Data sets that were read-only in the year 2000 (such as those on the SYSRES volume)
  - Data sets that are normally reset, such as page data sets
  - The master catalog.
- Once our data was restored, we brought up our remaining systems and subsystems and resumed our workloads.

A word of caution about not restoring the master catalog: if you are like us and have a master catalog containing more entries than absolutely necessary (“absolutely necessary” being those entries required early in the IPL process, along with aliases to user catalogs), you need to be careful. Consider this scenario: you delete data sets while the clock is set to the future. Then when you perform the full-volume restore after returning to the present, you restore the deleted data sets, but they are no longer cataloged. This is particularly messy for SMS-managed volumes, which require that all data sets be cataloged.

Summing Up—or, How Important Is This?

At the risk of repeating ourselves, we feel we must reiterate how important it is that you start a specific Year2000 project with dedicated resources. Even though IBM and many other companies are providing services to help you with this effort, you must at the very least recognize that such an effort is required and engage those services. And you must realize that the brunt of the impact falls on applications that are unique to your installation.

There are also additional issues to be considered, such as ensuring that input you receive from your suppliers, and products and services you provide to your customers, are all correct. A system crash brought about by a code problem is just one possibility for disaster. Incorrect data opens up all sorts of other unpleasant possibilities. Imagine sending out a whole month's bills that are incorrect. Nobody pays them and you have no cash flow until the problem is corrected. The consequences could be dire.

Without wishing to be alarmists, we cannot help but sound the alarm. If you have a Year2000 project in place, stay focused on it and keep it a high priority. If you haven't started to size this effort, ***start now***. We won't mind at all if you interrupt reading our test report while you look into this really important matter.

Year2000 Testing and the AS/400 System Date

A Word About This Section

This section on IBM AS/400 System Date and Testing has been duplicated in its entirety with minimal editing from the document entitled *Year 2000 Testing and the AS/400 System Date* at URL:

`http://www.softmall.ibm.com/as400/y2sysdat.html`

Overview

As the year 2000 approaches, and as more AS/400 installations are preparing for the new century, our customers have asked us to provide information regarding Year2000 testing and the AS/400 System Date. Members of the IBM AS/400 Year 2000 Technical Support Center have compiled the following information to provide you with ongoing year 2000 technical support. Through IBM's preparation of OS/400 for the year 2000, we have identified areas that may need attention prior to advancing the system date to or beyond 2000-January-01, and then setting it back to the current date, after completing your Year 2000 testing efforts.

IBM's preparation of OS/400 for the year 2000, included testing the operating system under several different date scenarios. A few examples are:

- Allowing the system date to age from 1999-December-21 to 2000-January-06 while running specific date-related tests in addition to a standard suite of system test scripts. This testing encompassed OS/400 in addition to many common applications such as OfficeVision/400, Client Access, DB2/400 Query Manager and SQL Development Kit, and Query/400.
- Requiring IBM-offered applications to conduct independent year 2000 testing in addition to the system testing described above. The application offerings were not made aware of what areas the system test would cover.

- Allowing the system date to range from 1996 to 2050 for specific component and licensed program testing such as Work Management, Command Analyzer, Database, and language compilers.
- Checking leap year calculations after the date has changed to or beyond 2000-January-01.
- Checking for non-valid dates.
- Performing save operations with the system date set to the current date and performing restore operations with the system date set to or beyond 2000-January-01.
- Performing save operations with the system date set to or beyond 2000-January-01, and performing restore operations after setting the system date back to the current date.
- Communicating from an AS/400 with the system date set to the current date, to an AS/400 with the system date set to or beyond 2000-January-01.
- Checking password expiration dates with the system date set to the current date and then setting the system date to or beyond 2000-January-01.
- Job scheduling dates with the system date set to the current date and then setting the system date to or beyond 2000-January-01.

Note

This section only addresses the functions of OS/400, the operating system for AS/400, Backup Recovery and Media Services (BRMS) and DataPropagator Relational. Our efforts do not encompass your application programs, and you will need to take necessary steps to determine what effects could occur to your application programs and user data.

The AS/400 system operates with the current system date and time. Messages, system logs, system utilities, and the like use date and timestamps to identify activities on the system. Changing the system date could affect data and system operational characteristics. This document covers many areas, but it is not an exhaustive report covering all possible areas that could be affected by setting and resetting the system date.

IBM recommends that customers use a non-production, stand-alone system to perform Year2000 testing. Prior to testing, you should perform a complete system save that includes all of your application programs and user libraries. When testing is complete, the system should be restored back to its original state before being used for any other purpose. If you must test on your production system, do so using highly controlled environments after carefully studying and considering the impacts of each testing scenario.

You should make every effort to avoid unpredictable results and retain the ability to recover from such problems that could occur when testing on a production system. We recommend the following general procedures:

- Schedule testing of Year2000 changes immediately after complete system saves.

- Develop a test of basic production activities as part of your testing plan. Perform the basic production activity after the system is returned to a normal production environment.
- Schedule time in your testing plan for recovery activities that might be required based on your test of production activities.
- Make any changes to source code when the system date is set to the current date. Unpredictable results could occur when your source code contains a date for a changed line of code set in the future.
- Make all basic system definition and system object changes while the system is in a restricted state both going to and returning from the Year2000 testing environment.
- As far as possible do not use production objects such as files, journals, subsystems, user ID's when testing. The goal is to create an independent environment within the limits of required activities.
- Design your testing in a modular fashion and do not start any production subsystems that are not required for testing the current module when coming out of restricted state into the Year2000 testing state.
- Any subsystems that are required for testing should be analyzed for potential activity outside the test environment. Potential unpredictable activity might require putting the subsystem's job queue on hold and manually activating only those jobs required for testing.
- Continually monitor (and contribute to) our Year2000 Web sites and publications for updated information on techniques to help meet the Year2000 challenge.

AS/400 FUNCTIONS

The following is a list of some of the AS/400 functions, how they work, and recommendations when considering advancing the system date to or beyond 2000-January-01 and then changing the system date back to the current date.

APPN

Advanced peer to peer networking (APPN) and advanced peer to peer connectivity (APPC) for the AS/400 handles all of the SNA protocol requirements when your system is communicating with a remote system using the LU session type 6.2 and node type 2.1 architectures. The remote system can be any of the following systems:

- AS/400 system
- System/36
- System/38
- IBM personal computer
- Displaywriter
- Series/1
- 5520 Administrative System
- RISC System/6000 (Reduced Instruction Set Computer)
- DPPX/370 (Distributed Processing Programming Executive)

One of the following host systems:

- System/390
- System/370

- 30XX processor
- 43XX processor
- 9370 system
- Any other system that supports the appropriate level of architecture.

Recommendations: If your system is part of an APPN/APPC network, it should be configured as an end-node and varied off the network before changing the system date to or beyond 2000-January-01, otherwise, unpredictable results could occur to your local system as well as other systems in your network. You can return the configuration back to the original state in the network and vary it back on after returning the system date back to the current date.

SNADS

When using systems network architecture distribution services (SNADS), to route and distribute messages and objects on the AS/400, the distribution items are enqueued in the appropriate distribution queue. The objects are date and timestamped, but are processed in the order of occurrence within each queue (that is, first in, first out). Therefore, when advancing the system date to or beyond 2000-January-01, and setting it back to the current date, distribution items are processed in the sequence in which they were received. However, they might not display as expected on screens or panels. To perform tests, you might decide to advance the system date to or beyond 2000-January-01, and change the system date back to the current date frequently. While you are doing this, if you allow distribution items to be sent and received, then unexpected results could occur, and the distribution items might not process correctly. In addition, if you allow distribution items to be sent from your AS/400 while the system date is advanced to or beyond 2000-January-01, to systems that are set at the current date, unpredictable results could occur.

Recommendations: Do not send distribution items or receive distribution items from the AS/400 while the system date is been set to or beyond 2000-January-01. To do this, end the subsystem QSNADS.

Note: If your system is IPLed at any time when the system date is set to or beyond 2000-January-01, the QSNADS subsystem can be activated during the start up process.

Performance Monitor

Performance monitor periodically collects performance data and puts it into system-supplied database files. The data is collected as often as specified on the time interval (in minutes) prompt (INTERVAL parameter) on the start performance monitor (STRPFRMON) command. Unpredictable results could occur with the integrity of the data in the database used by performance monitor, if the system date is changed to or beyond 2000-January-01, and then changed back to the current date, while the performance monitor is active.

Recommendations: End all performance monitor functions prior to advancing the system date to or beyond 2000-January-01, for testing, and not turning performance monitor functions back on until after you return to the current date. Turning off performance monitor functions can be done by issuing the end performance monitor (ENDPFRMON) from a command line. Turning on the performance monitor functions can be done by issuing the start performance monitor (STRPFRMON) from the command line.

PM/400

PM uses the performance monitor functions to gather data and create records that are stored in a database file for transmitting to IBM to analyze. If the system date is set to or beyond 2000-January-01, and PM/400 is running on your system, it will continue to gather data based on input that PM/400 gathers from the performance monitor functions. When the month-end information is transmitted to IBM for analysis, data that has not been processed is sent. This includes data gathered when the system date is set to or beyond 2000-January-01 and data gathered when the system date is set to the current date. As IBM processes the data transmitted by your system, it does not include the data gathered when your system date was set to or beyond 2000-January-01. The data will remain in the IBM database for processing at a later time (after the year 2000). After 1999-December-31, your system will continue to gather and transmit data from beyond 2000-January-01, for IBM to process. This data could collide with the data that was transmitted when you were performing Year2000 testing on your system.

Recommendations: Turn off PM/400 while your system date is set to or beyond 2000-January-01, and turn it back on when your system date is set to the current date. This can be done by ending the subsystem Q1PEGSCH before setting the system date to or beyond 2000-January-01, and starting the subsystem Q1PEGSCH after setting the system date back to the current date.

Note: If your system is IPL'd at any time during testing at or beyond 2000-January-01, the Q1PEGSCH subsystem may be activated during the start up process.

Job Scheduling

The job schedule function allows for time-dependent scheduling of AS/400 batch jobs. You can schedule jobs to be released from the job queue at a particular time, or you can use a job schedule entry to submit your jobs to the job queue automatically at the time you specify. Job scheduling allows you to control the date and time a batch job is submitted to or becomes eligible to start from a job queue. Scheduled jobs require user intervention before changing the system date to or beyond 2000-January-01, and back to the current date for testing purposes. Consider jobs that you scheduled to run at year end, like your financials. When changing the system date to or beyond 2000-January-01, all jobs scheduled to run from the current date to the date you set will process. This includes the job scheduled to run at year-end. You might not have all the data necessary to complete year-end processing. For instance, it might be July and your company's year-end might not be until December, in which case you could lose valuable data.

Recommendations: Hold all scheduled jobs before changing the system date to or beyond 2000-January-01. After you complete your testing and move the date back to the current date, reset the dates of your scheduled jobs. To do this, first create a save file with the create save file (CRTSAVF) command. Then perform a SAVOBJ for the object named QDFTJOBSCD of object type *JOBSCD in library QUSRSYS to the save file. Immediately restore the object, and all job run times are be recalculated. This will allow all of your scheduled jobs to execute at the times you originally established.

Operational Assistant

Operational assistant provides users with a menu interface to help them perform common end user and operator tasks. The user can benefit from enhanced menus and simplified terminology that allow to quickly perform common tasks. Operational assistant helps end users with the following tasks:

- Work with their printer output on the system
- Manipulate jobs they created and that reside on the system
- Work with and send messages to other users
- Record information related to a problem on the system
- Perform automatic cleanup of user messages, system and workstation messages, job logs and other system output, system journals and system logs, and OfficeVision for OS/400 calendar items.
- Schedule regular intervals for tape backups and system IPL's.
- Unpredictable results could occur if your system is using operational assistant to perform scheduled functions when you set your system date to or beyond 2000-January-01, and change the system date back to the current date. For example, all OfficeVision/400 calendar items prior to and including 1999 might be removed.

Recommendations: End all scheduled operational assistant features. The user can then go into each feature that is scheduled on their AS/400 through operational assistant. The user can then turn them off subsequent to changing the system date to or beyond 2000-January-01, and turning them back on when returning the system date back to the current date. To do this, go to the SETUP menu for operational assistant and select the features for clean up tasks, power on and off tasks, disk space tasks, and backup tasks. Another method would be to remove all operational assistant values from the system to prevent the scheduled features from processing. To do this, save the objects listed below before changing the system date to or beyond 2000-January-01, delete the objects from the system, and then restore them back to the system after moving the system date back to the current date. The objects in library QUSRSYS are listed in Figure 6-6 on page 6-23.

Figure 6-6. AS/400 QUSRSYS Library Objects

OBJECT	TYPE	TEXT
QEZBACKUPF	*USRIDX	OPERATIONAL ASSISTANT
QEZBACKUPL	*USRIDX	OPERATIONAL ASSISTANT
QEZPWRCLN	*USRIDX	OA CLEANUP AND POWER

Note: If you begin any operational assistant features while the system date is 2000-January-01 or beyond, new objects of type *USRIDX will be created if operational assistant does not find any on the system. This means that you will have to delete the new objects of type *USRIDX prior to restoring the old objects of type *USRIDX when you change the system date back to the current date.

Passwords

The length of time a password is valid for a user profile is controlled by the password expiration interval (PWDEXPITV) attribute of the user profile. The PWD EXPITV value can be set on the create user profile (CRTUSRPRF) or change user profile (CHGUSRPRF) command. The valid values are:

- *NOMAX -- the password will never expire
- 1-366 -- the number of days the password is valid
- *SYSVAL -- use the system value QPWDEXPITV to determine the expiration value. The values for QPWDEXPITV are:
  - *NOMAX -- the password will never expire
  - 1-366 -- the number of days the password is valid.

When you change the system date to or beyond 2000-January-01, you can exceed the password expiration interval for your user profile. At sign on time, if you have exceeded the expiration interval, you must change your password to complete the sign on process. Every user who changes their password while the system date is 2000-January-01 or beyond, will be expired the first time they sign on when the date is set back to the current date. This is because the sign on code checks the date the password was last changed and detects that it was changed at a date beyond the current date.

Recommendations: There is no recommendation for passwords because this condition is limited to users that sign on to the system when the system date is set to or beyond 2000-January-01. However, there are two options to prevent passwords from expiring:

- Change the PWDEXPITV value with the change user profile (CHGUSRPRF) command, and specify *NOMAX as the value for user profiles that will be used during the time that the system date is set to or beyond 2000-January-01, and then change the PWDEXPITV back to the original value when the system date is set back to the current date.
- Set the QPWDEXPITV system value to *NOMAX and prevent passwords from expiring when you change the system date to or beyond 2000-January-01. Then change the QPWDEXPITV system value back to the original value when the system date is set back to the current date.

Software Keys

Software keys and license agreements can expire when advancing the system date to or beyond 2000-January-01.

Recommendations: Contact all of your software providers to determine if this will affect your system. When reviewing your software, do not forget personal computer server-based applications, as the IPCS date is determined from the AS/400.

Journals

The main purpose of journal management is to enable you to recover the changes to a database file that have occurred since the file was last saved, if an error occurs with the database file. You can also use journal management for:

- An audit trail of activity that occurs for a database file or other objects on the system.

- Recording activity that has occurred for objects other than database files.
- Quicker recovery of access paths if your system ends abnormally.
- Quicker recovery when restoring from save-while-active media.
- Assistance in testing application programs.

Journals are created to record the activities that occur during the journal management process. The system keeps a record of changes you make to files that are journaled and of other events that occur on the system. These records are called journal entries. They are written to an object called a journal receiver. Some journal entries identify activity for a specific record, (added, updated, or deleted), and for a save, open, or close operation for a file. Journal entries can also identify other events that occur, such as security-relevant events on the system or changes made by dynamic performance tuning. In the publication *OS/400 Backup and Recovery - Advanced* (SC41-4305), Appendix B, “Journal Entry Information,” describes all possible journal entry types and their contents. Each journal entry includes additional control information that identifies the source of the activity, including, for example, the user, job, program, time, and date.

Changing the system date to or beyond 2000-January-01, will cause any journal entries written to application or system journal receivers to be written with that date, and a consecutive sequence number starting from the previous sequence number. Journal entries that are written after the system date is set to or beyond 2000-January-01, will be out of synchronization with the journal entries that were written when the system date was set to the current date. The journal entries would still be accessible using the sequence number option on journal commands, but attempting to utilize the starting/ending date and time keywords would fail.

Recommendations: Create new journal receivers when setting the system date to or beyond 2000-January-01, and again when returning the system date to the current date. You may also create test files without journals attached.

Security Audit Journaling

The QAUDCTL system value controls security journaling and should be changed to *NONE prior to changing the system date to or beyond 2000-January-01, and then back to the original value after changing the system date back to the current date. In addition, issue a change journal command (CHGJRN QAUDJRN *GEN) before setting the system date to or beyond 2000-January-01, and again after returning the system date back to the current date. The QAUDCTL system value controls when security auditing takes place. If the QAUDCTL system value is set to a value other than *NONE, security-relevant audit records will be placed in the active journal receiver QAUDJRN in library QSYS.

Recommendations: If the QAUDCTL system value on your system is not set to *NONE when the system date is set to or beyond 2000-January-01, then create a journal receiver to contain all of the security audit journal data collected. To do this, use the change journal (CHGJRN) command and specify *GEN for the QAUDJRN journal in library QSYS prior to changing the system date to or beyond 2000-January-01. This will detach the old journal receiver that contains the date and time security audit trail and creates and attaches a new journal receiver. When the system date is set back to the current date, use the change journal (CHGJRN) command again, specifying *GEN for the QUADHRN journal in library QSYS to create and attach a new journal receiver. This will isolate the security audit date obtained during the time that the system date is set to or beyond 2000-January-01.

DataPropogator Relational

DataPropogator Relational is a set of automated tools that copy relational data within and between DB2, DB2/2, DB2/6000, and DB2/400 relational databases. Data Propagator Relational consists of three components: administration, capture, and apply. The Data Propagator Relational/400 product contains all three components. Data Propagator Relational administration is the Data Propagator Relational component that supports registration, subscription, and the setting up of authorizations to Data Propagator Relational control tables.

Registration is the process of enabling a table to be copied by Data Propagator Relational. The table that is to be copied by Data Propagator Relational is referred to as the data (or source) table. Subscription is the process of setting up a copy definition. A copy definition defines the tables on the target system that receive data from the data table. These tables are called copy (or target) tables. The authorization process establishes the correct authorizations to all of the tables that Data Propagator Relational accesses.

The capture component captures changes made to registered database files and puts the changes in staging tables. The apply component use these staging tables to update copy tables as described in the next section. The capture component captures changes made to tables that have been registered using the Data Propagator Relational/2 product have been registered using the Data Propagator Relational/2 product (or the alternative AS/400 commands). The capture component is designed for automatic operation, which means that you can start the Capture component at the data server manually or as part of a system startup program. The data server is the location of the source table. On the AS/400, the Capture process is started using the Start DPP Capture (STRDPRCAP) command.

The apply component refreshes the copy tables, and is started on the system where the copy is being maintained. Like the capture component, the apply component is designed for automatic operation. You can start the apply component at each of your copy servers manually or as part of a system start-up program. The copy server is the location of the target table. On the AS/400, the apply process is started using the start DPR apply (STRDPRAPY) command.

Recommendations: End DataPropogator Relational when changing the system date from the current date to or beyond 2000-January-01 for Year2000 testing. This can be done by entering the End DataPropogator Relational (ENDDPRCAP) command on the command line. After your Year2000 testing is complete, you can then turn DataPropogator Relational back on by entering the Start DataPropogator (STRDPRCAP) command on the command line. DataPropogator Relational uses the OS/400 journal function extensively, and it is recommended that you follow the recommendation listed for journals and security audit journaling.

LICLOG (VLOGS)

LIC log (LICLOG) entry descriptions are used to aid in discovering code design errors. The LIC log entry identifier, sometimes referred to as the note ID or dump ID, appears in some error messages. The LIC log provides online recording for several types of occurrences in the LIC. LIC log entries are stored in the order of occurrence based on the sequence in which they occur. A date and time are associated with each entry, and they are displayed in the sequence of occurrence in relationship to the sequence of other entries in the log.

Recommendations: Notify your software support specialist that you have been performing Year2000 testing of your application software, in the event that you should need to contact software support for assistance with a problem on your system. This might help them identify the LIC log entries they might need to look for if the problem occurred during the time your system date was set to or beyond 2000-January-01.

Product Activity Log (PAL)

The product activity log (PAL) entry is primarily used for tracking or providing information about an event. Entries are logged by LIC or by other licensed programs. Entries are sent to the product activity log by the LIC I/O managers (IOMs), hardware drivers, and components such as machine check handler, machine facilities, and others. Product activity logs are stored and displayed using a date and timestamp retrieved from the system date and time.

Recommendations: Notifying your software support specialist that you have been performing Year2000 testing of your application software, in the event that you should need to contact software support for assistance with a problem on your system. This might help them identify the PAL entries they may need to look for if a problem occurred during the time your system date was set to or beyond 2000-January-01.

Spool Files

Often when jobs are processed by a subsystem, the rate that data can be handled by an input or output device is different from the rate that the system can handle it. This is obvious on a printer for example. It takes more time for a printer to print the data than for the system to get the data from a database file. Also, when a system has several users doing similar jobs, each user would like to use the system as if it were dedicated to his or her job. The system uses spooling to help in both of these situations. Spooling allows the system to store data in an object called a spooled file. The spooled file collects data from a device until a program or device is available to process the data. A program uses a spooled file as if it were reading form or writing to an actual device. This is input and output spooling.

When the system writes to a spooled file, the system will use system date and timestamps for the records that it writes. In addition, the spooled file has system date and timestamps associated with it regarding the time the job started, finished, and for displaying the spooled file on a screen or panel. When the system date is set to or beyond 2000-January-01 for testing, if a spooled file is open and the system is writing records to it, unpredictable errors could occur when you change the system date back to the current date, and attempt to access the spooled file. In addition, unpredictable results could occur when displaying the spooled file on a screen or panel.

Recommendations: The system should be as quiesced as possible during the time that the system date has been set to or beyond 2000-January-01, and then set back to the current date. Suspending jobs before changing the system date to or beyond 2000-January-01, would be helpful.

Save and Restore

There are a number of considerations for any saves done when you change your system date beyond the current date. The AS/400 operating system maintains a changed object list for each library that it references to optimize performance when SAVCHGOBJ operations are performed. The changed object list is used anytime the specified reference date/time (REFDATE and REFTIME parameters) are equal to or greater than the date/time that the last SAVLIB UPDHST(*YES) operation was performed. The changed object list is reset/cleared whenever SAVLIB UPDHST(*YES) is used (default mode of operation), which means that it is **not** reset when SAVLIB UPDHST(*NO) is used.

Recommendations: If you will be saving and restoring objects on your system when your system date is set to or beyond 2000-January-01, you should plan to scratch install the system when you change your system date back to the current date, or risk having unpredictable results with commands such as SAVCHGOBJ.

Backup Recovery and Media Services (BRMS)

There is a potential impact to a stand-alone BRMS system and a BRMS network when the system date is set beyond the current date and BRMS maintenance, movement, saves, and so forth, are performed, and then changing the system date back to the current date. Several of the BRMS functions will update the timestamp of tape volume or save history records with a future date, causing volumes to not expire properly, move properly, that will cause inaccurate reports when the system date is set back to the current date. This could be a problem in a BRMS network, where movement processed network-wide by the system that has a system date set beyond the current date, could corrupt the networked database with erroneous date and timestamps.

For the BRMS system performing Year2000 testing, there are additional database considerations. BRMS is used to backup a system for recovering to the current date, and when your system has the system date set to or beyond 2000-January-01, you are indicating to BRMS that you want to restore to that date. This will be the date stamped on any save history information for saves done during that time. When the system date is reset to the current date, the records with timestamps from 2000-January-01 and beyond, will appear to be more recent than any potential updates from operations done at the current date, and BRMS will not replace them. BRMS relies solely on the date and timestamp to determine the updates that are the most recent. This has impact on system recoverability. BRMS recovery reports for the test system will only show the last save performed while the date was set to or beyond 2000-January-01, no matter how many saves are subsequently completed after setting the system date back to the current date. In addition, BRMS uses journaling on data files, and all of these will contain the updates that occurred while the system date was set to or beyond 2000-January-01.

Recommendations: Follow one of the two procedures below to reduce the number of potential problems. Because of the potential problems, it is important that systems in a BRMS network doing Year2000 testing, (or any other date-forward/back changes), be isolated from the other systems. The system needs to be removed logically from the BRMS network so that the Year2000 system does not attempt to update the media records on the system that has the system date set to the current date. Be certain your system is up to date on all BRMS PTFs before starting these procedures.

Preferred Procedure: It is best if the entire system is saved before setting the system date to or beyond 2000-January-01, and then restored when you change the system date back to the current date. This can be done by preparing a full BRMS backup of the system and running a BRMS recovery report just before setting the clock forward. After completing your Year2000 testing, power the system down and IPL from your SAVSYS tape following the BRMS report instructions and complete a full recovery of your system. Restore the system as if you were doing a disaster recovery to a new system, so that old information is completely removed from disk.

Note: PTFs that were loaded while the system date was set to or beyond 2000-January-01, will not be on the system after a scratch install.

Alternate Procedure: If a scratch install of the system is not feasible, then do not perform BRMS activity on your system and do not include BRMS in your testing plans. The BRMS database relies on accurate timestamp comparison, and would not continue normally when you change the system date back to the current date. Check for and quiesce BRMS activity that would update any media volume. This includes STRMNTBRM (maintenance), MOVMEDBRM (movement), STRBKUBRM or STRARCBRM (saves or archives) and any change done to WRKMEDBRM volumes. Volumes should not be added to or removed from BRMS during the time the system date is set forward. Be certain to check the job scheduler and stop any BRMS jobs from being submitted automatically. The BRMS media monitor can be turned off from the BRMS menu under system policy, if native saves will occur.

Year2000 Testing and AIX Systems

A Word About This Section

This section on IBM RS/6000 and AIX Testing has been gleaned from various AIX documentation and supplied for inclusion here by members of the IBM RS/6000 AIX Development Lab.

This document only addresses the functions of AIX. Our efforts do not encompass your application programs, and you will need to take necessary steps to determine what effects could occur to your application programs and user data.

The AIX operating system operates with the current system date and time. Messages, system logs, system utilities, and so forth use date and timestamps to identify activities on the system. Changing the system date could affect data and system operational characteristics. This document covers many areas, but it is not an exhaustive report covering all possible areas that could be affected by setting and resetting the system date.

Overview

As the year 2000 approaches, and as more AIX installations are preparing for the new century, our customers have asked us to provide information regarding Year2000 testing and the AIX System Date. Members of the IBM RS/6000 AIX Development Lab have compiled the following information to provide you with ongoing Year2000 technical support. Through IBM's preparation of AIX for the Year 2000, we have identified areas that might need attention prior to advancing

the system date to or beyond 2000-January-01 and then setting it back to the current date, after completing your Year2000 testing efforts.

For further information on RS/6000 AIX products, refer to *AIX, UNIX Operating Systems, and the Year 2000 Issue*, available at the AIX Year2000 Web site at URL:

`http://www.software.ibm.com/year2000/papers/aixy2k.html`

Test Preparation

RS/6000 and the AIX OS

Before doing any system date and time modifications in testing software for Year2000 issues, please be aware of all current fix information; some updates might be required for your current system to operate correctly. There are certain hardware models that might require firmware updates, and releases of AIX prior to Version 4.2.1 might require software updates. Consult your AIX service representative and/or the AIX Year2000 paper mentioned above for complete information on these updates.

Other Software Products

Additionally, IBM provides an online and up-to-date database of the Year2000 status of all IBM AIX software offerings. The Year2000 Technical Support Center Web site at: `http://www.ibm.com/IBM/year2000/` contains an entry point to this database. It is highly recommended that you check the status of all IBM-supplied software for AIX on this database.

There are many software products (from IBM and non-IBM Solution Developers) that are date-sensitive because they have date-based licensing. Changing the date could possibly disable this software from legitimate use once Year2000 testing is finished. Be certain to contact your software vendor(s) before proceeding with testing.

Procedures

IBM recommends that customers use a non-production, stand-alone system to perform Year2000 testing. Prior to testing, you should perform a complete system save that includes all of your application programs and user libraries. When testing is complete, restore the system to its original state before being used for any other purpose. If you must test on your production system, do so using highly controlled environments after carefully studying and considering the impacts of each testing scenario.

You should make every effort to avoid unpredictable results and retain the ability to recover from such problems that could occur when testing on a production system. We recommend the following general procedures:

- Schedule testing of Year2000 changes immediately after complete system saves.
- Develop a test of basic production activities as part of your testing plan. Perform the basic production activity after the system is returned to a normal production environment.
- Schedule time in your testing plan for recovery activities that might be required based on your test of production activities.

- Make any changes to source code when the system date is set to the current date. Unpredictable results could occur when your source code contains a date for a changed line of code set in the future.
- Make all basic system definition and system object changes while the system is in a restricted state both going to and returning from the Year2000 testing environment.
- To the extent possible, do not use production objects such as files, journals, subsystems, user IDs when testing. The goal is to create an independent environment within the limits of required activities.
- Design your testing in a modular fashion and do not start any production subsystems that are not required for testing the current module when coming out of restricted state into the Year2000 testing state.
- Any subsystems that are required for testing should be analyzed for potential activity outside the test environment. Potential unpredictable activity might require putting the subsystem's job queue on hold and manually activating only those jobs required for testing
- Continually monitor (and contribute to) our Year2000 Web sites and publications for updated information on techniques to help meet the Year2000 challenge.

Testing Scenarios

IBM's preparation of AIX for the Year 2000, included testing the operating system under several different date scenarios. A few recommended scenarios are:

1. Set the system date to 1999-January and test. (This is to test for “magic numbers[1] .”
   a. Set the system date to 1999-September and repeat. (magic numbers testing)
   b. Set the system date to 1999-December (but not December 31st) and repeat. (magic numbers testing)
   c. Set the system date to 1999-December-31 (normal business hours) and repeat. (magic numbers testing)
2. Set the system time to 11:50 PM local time) 1999-December-31 and run over the midnight change. (Year2000 rollover testing)
   a. Set the system time to 11:50 PM UCT (Universal Coordinated Time - formerly known as Greenwich Mean Time) and repeat (Year2000 roll-over and system time-zone process testing)
   b. Set the system time to 2000-January-01, 11:59 PM and repeat
   c. Set the system date to 2000-February-29 and repeat (leap year testing)
   d. Set the system date to 2000-March-01 and repeat (leap year testing)
   e. Set the system to 2000-December-31 (normal business hours) and repeat. (year 2001 roll-over/leap year testing)

---

1 Magic number are defined here as those numbers such as 99 and 00 used to mean something special rather than a date. For example, the practice of using 99 to indicate no expiration date or 00 to indicate no data available.

f. Set the system time to 11:50 PM (local time) 2000-December-31 and run over the midnight change. (year 2001 roll-over/leap year testing)

g. Set the system time to 11:59 PM (local time) 2001-January-01 and run over the midnight change. (year 2001 roll-over/leap year testing)

Chapter 7. 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 might not be necessary for your specific environment, and your environment might require steps other than those listed here.

An Example 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.

   This might include upgrading your operating system to a machine capable of running Year2000-ready software as well as other system components such as storage resources. Additional storage (DASD or tape) that could double your current needs might be required to provide:
   - Backup of the original source code prior to your Year2000 conversion
   - Space to store the new source libraries following your Year2000 conversion
   - Space for the new versions of the system software while it is being installed and tested, while the old versions are still in production
   - Backup of the original data bases prior to your Year2000 conversion
   - Space for the new versions of the data bases (which will require additional space due to expanded date fields) while the old versions are still in production
   - Test file space
   - New JCL (and PROC) libraries to handle situations such as: sort problems, system parameter changes, program name changes, data set name changes, while the old JCL (and PROCs) are still in production
   - Backup of your current load libraries
   - Space for the new load libraries while the old versions are still in production.
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 run 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 8. Tool Categories and Available Tools to Ease Year2000 Changes

With the critical time constraint of the Year2000 challenge there is an immediate need for Year2000-ready applications. In addition, there is a demand for Year2000-ready applications of high quality at a 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 Year2000 challenge. Refer to "IBM Year 2000 Solution/Services Database" accessible through the IBM Year 2000 HomePage at URL:

`http://www.ibm.com/year2000`

for a partial listing 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 general information and lists tool products being marketed by Solution Developers and IBM. You should contact the Solution Developers directly for specific information regarding their products.

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: OS/390, MVS, VM/ESA, VSE/ESA, 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.

That is, your system performs essentially as it did before running the tools.

Tool Categories

The following sections highlight some useful tool types for handling the Year2000 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.

- ***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 OS/390 and MVS

The IBM COBOL Family for MVS & VM

The IBM COBOL Family

IBM COBOL Family extends over multiple platforms: workstation (OS/2 and AIX) and host (MVS, VM, VSE, and OS/400). All the IBM COBOL compilers except for ILE COBOL (AS/400) share common components, which gives source and compiler compatibility across multiple platforms.

The MVS & VM COBOL product is discussed in "The IBM COBOL Family for MVS & VM." The VSE COBOL product is discussed in "The IBM COBOL Family for VSE/ESA" on page 8-33. The OS/400 product is described in "IBM Tools for AS/400" on page 8-43. The OS/2 and Windows and the AIX products are described in "The IBM COBOL Family for the Workstations" on page 8-68.

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

- OS/390 COBOL
- 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

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 supports analysis, reporting, and restructuring
- CICS Application Migration Aid converts CICS macro-level source
- COBOL Report Writer Precompiler supports Report Writer code
- EDGE Portfolio Analyzer is a load library inventory tool.
- The Redeveloper in VisualAge for COBOL

Year2000 Highlights of IBM COBOL for MVS & VM

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

- Intrinsic functions
- Sliding 100-year window

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

COBOL for MVS & VM provides full Year2000 support. It provides ANSI COBOL Standard Intrinsic Functions that 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. When the COBOL application program queries the system for the current date, COBOL for MVS & VM returns a 4-digit year.

You can use IBM VisualAge for COBOL, Professional for OS/2 to identify, change, recompile, and test your legacy code on the workstation. Using the VisualAge for COBOL, you can work from the workstation on code residing on the mainframe. Refer to “The IBM COBOL Family for the Workstations” on page 8-68 for more detail.

Features of IBM COBOL for MVS & VM

IBM COBOL for MVS & VM is the high-performance compiler with a supporting run-time environment product (Language Environment for MVS & VM), that facilitates multiple-language interaction. This is different from the single packaging of compiler and run-time environment for OS/VS COBOL and VS COBOL II.

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 allows 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. The System Object Model (SOM) is the core object oriented (OO) architecture for SOMobjects for MVS and is IBM's strategic architecture for building and manipulating class libraries. Enabling to SOM on MVS lets IBM COBOL programmers 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.
- 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 help extend the useful life of existing applications.

- Support for Year2000.
- Capabilities to help application programmers incrementally enhance applications.
- Help in maintaining and enhancing the investment in existing programmer skills.

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

- Source-level compatibility with IBM VisualAge for COBOL for OS/2 and IBM COBOL Set for AIX.
- Improved interoperability with C and C++.
- Performance enhancements.
- Mainframe Interactive Debug Tool Feature.
- Object-oriented language extensions—letting developers 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.

IBM COBOL for MVS & VM Release 2 is compatible with VS COBOL II and IBM COBOL/370 Release 1. For object-oriented applications, customers must also order SOMobjects for MVS.

IBM COBOL for MVS & VM, with its improved interlanguage communications (ILC), allows you to:

- develop COBOL applications and integrate them with existing applications, irrespective of the language used,
- use existing code in new applications (code reuse) regardless of the source code language used, and
- take advantage of functionality available in other software packages.

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, and flexible run-time options.

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.

IBM Language Environment for MVS & VM

IBM Language Environment for MVS & VM is IBM's common run-time environment for enterprise applications written in COBOL, PL/I, C, and Fortran. In addition to providing standard language run-time library support, 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 and VM, interlanguage communication in mixed-language applications is easy, efficient, and consistent. You can share and reuse code easily. You can write a service routine in the language of your choice—C, C++, COBOL, Fortran, PL/I, or assembler—and then allow that routine to be called from C, C++, COBOL, Fortran, PL/I, or assembler applications. Similarly, vendors can write one application package in the language of their choice and allow the application package to be called from C, C++, COBOL, Fortran, PL/I, and assembler routines.

Language Environment for MVS & VM enables existing applications to function as with pre-Language Environment runtime libraries 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 run-time 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 run-time environment for their applications, regardless of the programming language or system resource needs.

Today, IBM Language Environment for MVS & VM is the pre-requisite run-time library for 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 that 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:

- 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 sliding 100-year window feature. If you are unable to change all of your applications and data at the same time, this feature allows 2-digit years to be interpreted in a 100-year window. Any 100-year window can be selected. You pass a valid date containing a 2-digit year to Language Environment and Language Environment returns an integer date based on the 100-year window.

The advantage to the 100-year window is that you need to change only the application code and not the databases or files with 2-digit years. This lets you change the application programs one at a time or groups at a time without affecting 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.

Disadvantages of the 100-year window are:

- It doesn't last forever. If your application has a long life expectancy, you may need to go back and replace the 100-year window with full 4-digit year support. Some of your applications will be replaced prior to those changes.
- Your system will likely experience some performance impact due to the additional programming logic used to determine and append the century digits.

IBM CoOperative Development Environment/370 (CODE/370)

IBM CoOperative Development Environment/370 (CODE/370) has been replaced by the VisualAge for COBOL for OS/2. This lets you use the workstation to edit, compile, and debug COBOL source code that resides on a host computer. For details about this product, refer to "The IBM COBOL Family for the Workstations" on page 8-68.

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 describes issues involved with migrating from OS/VS COBOL to COBOL for MVS & VM.

Migrating from OS/VS COBOL: Because OS/VS COBOL service has been discontinued, IBM encourage 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 the Year2000 solution providing 4-digit year support. Depending on which product is currently being used and how fast a company is willing to migrate, you can take three possible migration paths:

1. **OS/VS COBOL to COBOL for MVS & VM**
   We encourage companies with OS/VS COBOL 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 met. See the IBM Language Environment for MVS & VM Licensed Program Specification for prerequisite information.
2. **OS/VS COBOL to VS COBOL II to COBOL for MVS & VM**
   Companies with OS/VS COBOL who do not yet meet the IBM Language Environment for MVS & VM prerequisites must migrate to VS COBOL II first.

After the IBM Language Environment for MVS & VM prerequisites are met, then companies can move to IBM COBOL for MVS & VM.

3. **VS COBOL II to COBOL for MVS & VM**
   After the IBM Language Environment for MVS & VM prerequisites are met, companies with VS COBOL II can migrate to IBM COBOL for MVS & VM.

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

VS COBOL II: VS COBOL II builds on the functions of OS/VS COBOL but has a variety of features that provide 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 meet the IBM Language Environment for MVS & VM prerequisites should run VS COBOL II until they can migrate to Language Environment.

VS COBOL II does provide some support for the Year2000 transition:

- The date-identifier CALL 'IGZEDT4' USING BY REFERENCE returns the current date in the form YYYYMMDD. IGZEDT4 is provided because VS COBOL II does not include support for the 4-digit year. The ability to obtain a 4-digit-year date was first included in COBOL/370 release 1.
- For VS COBOL II programs that are running under Language Environment the following routines can be using dynamic CALLs only:
  CEECBLDY--Convert Date to COBOL Integer Format
  CEEDATE--Convert Lilian Date to Character Format
  CEEDATM--Convert Seconds to Character Timestamp
  CEEDAYS--Convert Date to Lilian Format
  CEEDYWK--Calculate Day of Week from Lilian Date
  CEEGMT--Get Current Greenwich Mean Time
  CEEGMTO--Get Offset from Greenwich Mean Time to Local Time
  CEEISEC--Convert Integers to Seconds
  CEELOCT--Get Current Local Date or Time
  CEEQCEN--Query the Century Window
  CEESCEN--Set the Century Window
  CEESECI--Convert Seconds to Integers
  CEESECS--Convert Timestamp to Seconds
  CEE3CTY--Set Default Country

Thus, there are various ways to make VS COBOL II programs Year2000-ready. However:

- IGZEDT4 is not supported under CICS.
- The VS COBOL II compiler is not Year2000 ready because it has no COBOL language support for 4-digit years and none is planned.
- IBM COBOL for MVS & VM, IBM COBOL for VSE/ESA, IBM VisualAge for COBOL of OS/2, IBM COBOL Set for AIX, and ILE COBOL for AS/400 are all Year2000 ready.

Benefits of COBOL Migration: OS/VS COBOL is the ANSI 68/74 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 host product, COBOL for VSE/ESA. Customers who have migrated to the newer COBOL Standard have additional function, increased developer productivity, and exploitation of S/390 hardware capabilities. 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.
- Fully Year2000 ready.
- 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 for ILC and dynamic call users.
- Increased maintenance productivity from structuring tools such as IBM's COBOL Structuring Facility that is part of the Redeveloper in VisualAge for COBOL, Professional for OS/2.
- 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 and Language Environment?*, which is available from your IBM representative (COBMGVAL PACKAGE on MKTTOOLS) or by calling 1-800-IBM-7777 extension STAR703. This document contains examples of customer benefits of migrating to COBOL for MVS & VM.

For a detailed explanation of why and how to migrate to COBOL for MVS & VM and to Language Environment, refer to *Compiler and Run-Time Migration Guide*, GC26-4764.

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

Converting on the Workstation

The Redeveloper in the VisualAge for COBOL, Professional for OS/2 lets you convert and structure COBOL programs using the workstation with a GUI interface. Refer to “Redeveloper Tools” on page 8-72 for details on this component instead of CCCA and COBOL/SF.

COBOL and CICS/VS Command Level Conversion Aid (CCCA) is an effective tool that makes 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. You can use CCCA to convert from OS/VS COBOL to COBOL for MVS & VM and 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, you can use the COBOL Structuring Facility, a re-engineering tool, to automatically invoke CCCA. This lets you convert programs before structuring them with the COBOL Structuring Facility.

CCCA identifies and converts source code incompatibility, to reduce the effort required to convert programs, and to minimize conversion errors. You can customize the conversion process 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

Converting on the Workstation

The Redeveloper in the VisualAge for COBOL, Professional for OS/2 lets you convert and structure COBOL programs using the workstation with a GUI interface. Refer to “Redeveloper Tools” on page 8-72 for details on this component instead of CCCA and COBOL/SF.

IBM COBOL Structuring Facility/MVS & VM (COBOL/SF) 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 you can use in a client/server application architecture:

1. Converting OS/VS COBOL to VS COBOL II or to COBOL for MVS & VM
2. Restructuring
3. Modularizing to create functional program units.

If the functional units are wrapped with an object-oriented COBOL wrapper, then you can use these primitive parts for new OO applications.

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.
- Online tutorial.
- Comprehensive online 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. You can use it 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. It is generally recommended that a program be restructured whenever 10 to 15 percent of a program has changed because of program errors or enhancements.

CICS Application Migration Aid

The CICS Application Migration Aid assists you to convert 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. Instead it requires a command level 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

You can use the COBOL Report Writer Precompiler to perform two functions: 1) to 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 2) 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 were 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 you.

EDGE Portfolio Analyzer—COBOL and PL/I Migration Tool

The EDGE Portfolio Analyzer Version 1 can significantly reduce the effort necessary to migrate from earlier versions of IBM host COBOL and PL/I compilers to the latest levels of these products.

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

- Performance improvements of certain load modules simply by recompiling them with different options.

The EDGE Portfolio Analyzer frees you from many hours of tedious investigation and lets you move more quickly into the new complier environment.

Workstation Interactive Test Tool Year2000 (WITT Year2000) for OS/2

WITT Year2000 for OS/2 is a test tool to assist you in testing of the Year 2000 transformation.

COBOL programmers can use WITT Year2000 as a GUI test tool that automatically facilitates the creation of reusable unit test, function test, system test, and regression test cases. You can create these reusable test cases when you develop or implement a newly developed program, and you should run them 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 you make enhancements and fixes to a program. This automatic playback and comparison capability is best suited for testing of Year 2000 transformation work. Because once you make a modification to an application programs, you should test repeatedly with a variety of system dates such as:

- 1997/02/28 (1997-February-28)
- 1999/12/31 (1999-December-31)
- 2000/01/01 (2000-January-01)
- 2000/02/29 (2000-February-29)
- 2004/02/29 (2004-February-29)

WITT also allow you to add program intelligence to your test cases though scripting in 2/REXX, thus easing your modification of WITT to meet your needs.

WITT Year2000 installs on an OS/2 desktop and allows the testing of OS/390, MVS, VM, IMS, CICS, OS/400, and OS/2 PM and Text (that is, Micro Focus and CICS/OS2) applications.

VisualAge 2000 Test Solution

The VisualAge 2000 Test Solution offers MVS users of COBOL a process and a tool set to test year 2000 changes to applications. Rather than testing to determine whether new enhancements work correctly, year 2000 testing focuses on determining whether applications behave the same way they did before year 2000 modifications were applied. VisualAge 2000 Test Solution targets year 2000 testing with a combination of a document, step-by-step process, and real-life examples.

The tools described in the offering help you determine path coverage, distill data, script input, and compare output.

VisualAge 2000 Test Solution is available as an IBM Redbook, *Testing Your Year 2000 Conversion*, SG24-2230, which comes with detailed examples on CD-ROM as

well as specific information on IBM's Year 2000 test process and tool usage. This step-by-step guide is intended to help you through your Year 2000 test process.

You can use the Redbook as a stand-alone reference or -- for added value -- in conjunction with the CD-ROM-based tutorials.

Detailed Description: The VisualAge 2000 Test Solution documentation outlines the steps you take to complete year 2000 testing through a combination of the detailed process, help with using tools, and real-life examples.

The tools described in this offering assist in determining path coverage, distilling data, scripting input, and comparing output.

Coverage Assistant and Distillation Assistant are part or the IBM Application Testing Collection for MVS/ESA. (These tools are currently available from IBM Global Services as part of an IBM services engagement. Contact your IBM representative for availability in your area. These services are not available in all countries.)

- **Coverage Assistant** measures code coverage in programs written in COBOL and PL/I languages.
- **Distillation Assistant** helps you eliminate test data that does not test Year 2000 fixes and redundant test data. You can then use the new, smaller distilled file during the test process to obtain equivalent code coverage yet decrease the resources required for testing.
- **Debug Tool** is IBM's strategic multi-language, general-purpose, interactive debug tool for use in debugging programs that are not performing correctly. You can use the frequency count feature of Debug Tool to check path coverage. Also useful is the scripting feature. You can enter data in a legacy program and “replay” that data in a year 2000 program.
- **DFSORT** helps you analyze and transform data at the record, field, and bit level. With the year 2000 features of DFSORT, you can transform a wide variety of dates with 2-digit years to dates with 4-digit years, based on a specific sliding or fixed-century window.
- **Enhanced SUPERC** compares two flat files, reports, or screen images. Enhanced SUPERC can handle year 2000-specific comparisons. For example, it can compare data fields of different sizes, such as 2-digit and 4-digit dates. It can also compare dates that area stored differently, such as PIC 9(6) dates and COMP-3 dates. Because Enhanced SUPERC can handle the structured differences in date fields, and differences it finds point to genuine program problems.
- **COBOL Tester**, part of IBM VisualAge for COBOL, Test for OS/2, provides the means to interactively create test data and determines the path coverage with the test data. COBOL Tester can compare a calculated output with an expected output for given input values as a variation of regression testing.
- **Auto Test Performer (ATP)**, also based part of VisualAge for COBOL, Test for OS/2, captures data entry and screen images for interactive legacy applications. At a later stage of testing, ATP plays back the script in year 2000 applications. You can also compare the benchmark, or legacy, screen image and the current year 2000, screen image.

- **WITT Year2000** is another data entry and screen image capturing tool that runs in the Windows NT and OS/2 environment.

Key Prerequisites: The following set of IBM products (at the specified version/release) comprise the VisualAge 2000 Test Solution product set:

- Language Environment Version 1.3, or later (5688-198)
- CICS Version 4.2 or later (5655-018)
- DFSORT Version 1 Release 13 (5740-SM1) with PTFs UN90139, UQ05520, and UQ00530 installed
- ISPF Version 3 Release 5, or later (5685-054)
- Enhanced SUPERC (5696-234) Toolkit Feature of High Level Assembler
- IBM VisualAge for COBOL, Test for OS/2 (33H0005)
- IBM Debug Tool to run the operational tutorial. This tool is available with the full-function feature of these compliers:
  - IBM COBOL for OS/390 and VM
  - IBM COBOL for MVS and VM

Ordering Information

- Phone: 800-879-2755
- FAX: 900-445-9269
- Mail address:

  IBM Publications
  P.O. Box 29570
  Raleigh, NC 27626-0554

- Internet: http://www.redbooks.ibm.com/redbooks

Be sure to include the book name, order number, and quantity as well as your name, address, phone, fax number, and value added tax number if applicable.

For billing, specify your IBM customer number, or credit card number with expiration date.

The IBM PL/I Family for MVS & VM

For application code written in PL/I for S/370 and S/390 (for OS/390, MVS, and VM) platforms, IBM has developed a PL/I compiler. 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** – 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 100-year window

PL/I for MVS & VM provides full Year2000 support. It provides a 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 to:

- generate statistics
- modularize, migrate, or restructure code
- search class libraries.

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 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 assembler, FORTRAN, and 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 OS/390 and 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 Year2000
- 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, with its improved interlanguage communications (ILC), allows you to:

- develop PL/I applications and integrate them with existing applications, irrespective of the language used,
- use existing code in new applications (code reuse) regardless of the source code language used, and
- take advantage of functionality available in other software packages.

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 is IBM's common runtime environment for enterprise applications written in assembler, 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 the most commonly used 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 100-year window feature. If you are unable to change all of your applications and data at the same time, the 100-year window feature allows 2-digit years to be interpreted in a 100-year window. Any 100-year window can be selected. You pass a valid date containing a 2-digit year to Language Environment and Language Environment returns an integer date based on the 100-year window.

  The advantage to the 100-year 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 100-year 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 100-year window with full 4-digit year support. Some of your applications will be replaced prior to this formatting change.

IBM CoOperative Development Environment/370 (CODE/370)

IBM CoOperative Development Environment/370 (CODE/370) — 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

Other host PL/I compilers developed for the MVS and VM mainframe: OS PL/I V1, OS PL/I V2.

OS PL/1 V1 was withdrawn from service in 1995-December. 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, here 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 - Language Migration Tool

The Edge Portfolio Analyzer Version 1 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. (The Edge Portfolio Analyzer also recognizes COBOL, C/C++, FORTRAN, Pascal, and Assembler programs.)

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 Year2000 (WITT Year2000) for OS/2

WITT Year2000 for OS/2 is a test tool to assist you in testing of the Year 2000 transformation.

PL/I programmers can use WITT Year2000 as a GUI test tool that automatically facilitates the creation of reusable unit test, function test, system test, and regression test cases. You can create these reusable test cases when you develop or implement a newly developed program, and you should run them 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 you make enhancements and fixes to a program. This automatic playback and comparison capability is best suited for testing of Year 2000 transformation work. Because once you make a modification to an application programs, you should test repeatedly with a variety of system dates such as:

- 1997/02/28 (1997-February-28)
- 1999/12/31 (1999-December-31)
- 2000/01/01 (2000-January-01)
- 2000/02/29 (2000-February-29)
- 2004/02/29 (2004-February-29)

WITT also allow you to add program intelligence to your test cases though scripting in 2/REXX, thus easing your modification of WITT to meet your needs.

WITT Year2000 installs on an OS/2 desktop and allows the testing of OS/390, MVS, VM, IMS, CICS, OS/400, and OS/2 PM and Text (that is, Micro Focus and CICS/OS2) applications.

DFSORT

DFSORT V1R13 (with PTF UN90139) provides Year2000 capabilities by providing the ability to sort, merge, and transform 2-digit years according to a specified sliding or fixed 100-year 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 100-year window for your applications. For example, set a 100-year window of 1915-2014 or 1950-2049.
- Order 2-digit character, zoned decimal, packed decimal, or decimal year data, according to the 100-year 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 100-year 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 100-Year Window

A new installation and run-time option allows you to specify a sliding or fixed 100-year window to be used with 2-digit years. Y2PAST=s specifies a **sliding** 100-year window starting s years before the current year. For example, if the current year is 1996, Y2PAST=80 starts the 100-year window at 1996 – 80 = 1916, providing a 100-year window of 1916 through 2015. In 1997, this 100-year window automatically slides to 1917 through 2016.

Y2PAST=f specifies a **fixed** 100-year window starting at f. For example, Y2PAST=1950 starts the 100-year window at 1950, providing a 100-year 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 100-year 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 100-year 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 100-year 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 100-year 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 100-year window.

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

```
* Set the 100-year 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
*                      *   100-year 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 is a common date routine that offers necessary functions for validation, conversion, and calculation of dates in any commonly-used 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 NCALs, 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 has previously been created by the calendar owner.

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

COMUDAS is a standard programming request for price quotation (PRPQ) and can be ordered through standard IBM ordering channels.

IBM Tools for VM/ESA

The IBM COBOL Family for MVS & VM

For application code written in COBOL for S/370 and S/390 (for OS/390, 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 8-7 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 8-19 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/ESA

The IBM COBOL Family for VSE/ESA

The IBM COBOL Family

IBM COBOL Family extends over multiple platforms: workstation (OS/2 and AIX) and host (MVS, VM, VSE, and OS/400). All the IBM COBOL compilers except for ILE COBOL (AS/400), share common components, which gives source and compiler compatibility across multiple platforms.

The MVS & VM COBOL product is discussed in "The IBM COBOL Family for MVS & VM" on page 8-7. The VSE COBOL product is discussed in "The IBM COBOL Family for VSE/ESA." The OS/400 product is described in "IBM Tools for AS/400" on page 8-43. The OS/2 and Windows and the AIX products are described in "The IBM COBOL Family for the Workstations" on page 8-68.

IBM has developed COBOL compilers for application code written in COBOL for S/370 and S/390 for the VSE platform. 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

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

Year2000 Highlights of IBM COBOL for VSE

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

- Intrinsic functions
- Sliding 100-year 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. When the COBOL application program queries the system for the current date, COBOL for VSE/ESA returns a 4-digit year.

Many 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

Features of IBM COBOL for VSE/ESA

IBM COBOL for VSE/ESA is the high-performance compiler with a supporting run-time environment product (Language Environment for VSE/ESA), that facilitates multiple-language interaction. This is different from the single packaging of compiler and run-time 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 gives you simple 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.
- 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 functions 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.

IBM Language Environment for VSE/ESA

IBM Language Environment for VSE/ESA is IBM's common run-time environment for enterprise applications written in COBOL and PL/I. Language Environment for VSE/ESA provides 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, interlanguage communication in mixed-language applications is easy, efficient, and consistent. You can share and reuse code easily. You can write a service routine in the language of your choice—COBOL, PL/I, or assembler—and then allow that routine to be called from COBOL, PL/I, or assembler applications. Similarly, vendors can write one application package in the language of their choice and allow the application package to be called from COBOL, PL/I, and assembler routines.

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 run-time 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 run-time 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 IBM COBOL for VSE/ESA and IBM 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 that 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:

- 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 sliding 100-year window feature. If you are unable to change all of your applications and data at the same time, this feature lets 2-digit years be interpreted in a 100-year window. Any 100-year window can be selected.

You pass a valid date containing a 2-digit year to Language Environment and Language Environment returns an integer date based on the 100-year window.

The advantage to the 100-year window is that you need to change only the application code and not the databases or files with 2-digit years. This lets you 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 100-year window is that it doesn't last forever. If your application has a long life expectancy, you may need to go back and replace the 100-year window with full 4-digit year support. Some of your applications will be replaced prior to this.

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 describes issues involved with migrating from DOS/VS COBOL.

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

1. **DOS/VS COBOL to COBOL for VSE/ESA**
   We encourage companies with DOS/VS COBOL to migrate directly to IBM COBOL for VSE/ESA (compiler) and IBM Language Environment for VSE/ESA (run-time library) if the prerequisites are met.
2. **DOS/VS COBOL to VS COBOL II to COBOL for VSE/ESA**
   Companies with DOS/VS COBOL who do not yet meet the IBM Language Environment for VSE/ESA prerequisites should migrate to VS COBOL II first. After the IBM Language Environment for VSE/ESA prerequisites are met, then companies can move to IBM COBOL for VSE/ESA.
3. **VS COBOL II to COBOL for VSE/ESA**
   After the IBM Language Environment for VSE/ESA prerequisites are met, companies with VS COBOL II can migrate to IBM COBOL for VSE/ESA.

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

VS COBOL II: VS COBOL II 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.

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

For VS COBOL II programs that are running under Language Environment, the following routines can be called using dynamic CALLs only:

CEECBLDY--Convert Date to COBOL Integer Format

CEEDATE--Convert Lilian Date to Character Format
CEEDATM--Convert Seconds to Character Timestamp
CEEDAYS--Convert Date to Lilian Format
CEEDYWK--Calculate Day of Week from Lilian Date
CEEGMT--Get Current Greenwich Mean Time
CEEGMTO--Get Offset from Greenwich Mean Time to Local Time
CEEISEC--Convert Integers to Seconds
CEELOCT--Get Current Local Date or Time
CEEQCEN--Query the Century Window
CEESCEN--Set the Century Window
CEESECI--Convert Seconds to Integers
CEESECS--Convert Timestamp to Seconds
CEE3CTY--Set Default Country

Thus, there are various ways to make VS COBOL II programs Year2000 ready. However:

- The VS COBOL II compiler is not Year2000 ready because it has no COBOL language support for 4-digit years and none is planned.
- IBM COBOL for MVS & VM, IBM COBOL for VSE/ESA, IBM VisualAge for COBOL of OS/2, IBM COBOL Set for AIX, and ILE COBOL for AS/400 are all Year2000 ready.

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 functions, increased developer productivity, and exploitation of S/390 hardware capabilities. 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.
- Fully Year2000 ready.
- 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 and Language Environment?*, which is available from your IBM representative (COBMGVAL PACKAGE on MKTTOOLS) or by calling 1-800-IBM-7777 extension STAR703. This document contains some examples of customer benefits of migrating to COBOL for MVS & VM that also apply to the VSE environment.

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

Converting on the Workstation

The Redeveloper in the VisualAge for COBOL, Professional for OS/2 lets you convert and structure COBOL programs using the workstation with a GUI interface. Refer to “Redeveloper Tools” on page 8-72 for details on this component instead of CCCA.

COBOL and CICS/VS Command Level Conversion Aid (CCCA) for VSE is an effective tool for converting 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. You can use CCCA to convert from DOS/VS COBOL to COBOL for VSE/ESA, and 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 identifies and converts source code incompatibility, to reduce the effort required to convert programs, and to minimize conversion errors. You can customize the conversion process to meet unique conversion requirements. Installation and usage are 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 DOS/VS COBOL and the current release of VS COBOL II 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

You can use the COBOL Report Writer Precompiler to perform two functions: 1) to permanently convert Report Writer statements to valid COBOL statements that can be compiled in IBM COBOL for VSE/ESA or IBM COBOL for MVS & VM, or 2) to precompile applications containing Report Writer statements so the code will be acceptable to the IBM COBOL for VSE/ESA or IBM COBOL for MVS & VM compiler.

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

Workstation Interactive Test Tool Year2000 (WITT Year2000) for OS/2

WITT Year2000 for OS/2 is a test tool to assist you in testing of the Year 2000 transformation.

COBOL programmers can use WITT Year2000 as a GUI test tool that automatically facilitates the creation of reusable unit test, function test, system test, and regression test cases. You can create these reusable test cases when you develop or implement a newly developed program, and you should run them 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 you make enhancements and fixes to a program. This automatic playback and comparison capability is best suited for testing of Year 2000 transformation work. Because once you make a modification to an application programs, you should test repeatedly with a variety of system dates such as:

- 1997/02/28 (1997-February-28)
- 1999/12/31 (1999-December-31)
- 2000/01/01 (2000-January-01)
- 2000/02/29 (2000-February-29)
- 2004/02/29 (2004-February-29)

WITT also allow you to add program intelligence to your test cases though scripting in 2/REXX, thus easing your modification of WITT to meet your needs.

WITT Year2000 installs on an OS/2 desktop and allows the testing of OS/390, MVS, VM, IMS, CICS, OS/400, and OS/2 PM and Text (that is, Micro Focus and CICS/OS2) applications.

The IBM PL/I Family for VSE

To write application code in PL/I for S/370 and S/390 for the VSE platforms, IBM has developer a PL/I complier. 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 100-year window

PL/I for VSE/ESA provides full Year2000 support. It provides a 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 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 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 the most commonly used 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 100-year window feature. If you are unable to change all of your applications and data at the same time, the 100-year window feature allows 2-digit years to be interpreted in a 100-year window. Any 100-year window can be selected. You pass a valid date containing a 2-digit year to Language Environment and Language Environment returns an integer date based on the 100-year window.

  The advantage to the 100-year 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.

  Disadvantages of the 100-year window are:

  – It doesn't last forever. If your application has a long life expectancy, you may need to go back and replace the 100-year window with full 4-digit year support. Some of your applications will be replaced prior to those change.

  – Your system will likely experience some performance impact due to the additional programming logic used to determine and append the century digits.

Workstation Interactive Test Tool Year2000 (WITT Year2000) for OS/2

WITT Year2000 for OS/2 is a test tool to assist you in testing of the Year 2000 transformation.

PL/I programmers can use WITT Year2000 as a GUI test tool that automatically facilitates the creation of reusable unit test, function test, system test, and regression test cases. You can create these reusable test cases when you develop or implement a newly developed program, and you should run them 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 you make enhancements and fixes to a program. This automatic playback and comparison capability is best suited for testing of Year 2000 transformation work. Because once you make a modification to an application programs, you should test repeatedly with a variety of system dates such as:

- 1997/02/28 (1997-February-28)
- 1999/12/31 (1999-December-31)
- 2000/01/01 (2000-January-01)
- 2000/02/29 (2000-February-29)
- 2004/02/29 (2004-February-29)

WITT also allow you to add program intelligence to your test cases though scripting in 2/REXX, thus easing your modification of WITT to meet your needs.

WITT Year2000 installs on an OS/2 desktop and allows the testing of OS/390, MVS, VM, IMS, CICS, OS/400, and OS/2 PM and Text (that is, Micro Focus and CICS/OS2) applications.

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. You can find a more complete list of available tools in *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, to include support for 2-digit-year data.
- Retrieval of the job date through special words *DATE and *YEAR, to include support for 4-digit-year data.
- Retrieval of the system date through operation TIME to include support for both 2-digit and 4-digit-year data.
- Date, Time, and Timestamp data type support to include support for date operations such as subtraction, addition, extraction, testing, comparison, and move.
- Calling OPM System APIs, to include support for both 2-digit and 4-digit-year data.
- Calling ILE Date APIs, to include support for both 2-digit and 4-digit-year data.
- SQL date, time, and timestamp data type support, to include support for 4-digit-year data.

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 specification.

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 8-1. 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 8-2. 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 8-3. 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 8-4. 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 unit; 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 includes the following support for dates:

- Retrieval of the job date through special words UDATE and UYEAR, to include support for 2-digit-year data.
- Retrieval of the job date through special words *DATE and *YEAR, to include support for 4-digit-year data.
- Retrieval of the system date through operation TIME, to include support for both 2-digit and 4-digit-year data.
- Calling OPM System APIs, to include support for both 2-digit and 4-digit-year data.

- SQL date, time, and timestamp data type support, to include support for 4-digit-year data.

Using System/36 Compatible Date Support

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

- Retrieval of the job date through special words UDATE and UYEAR, to include support for 2-digit-year data.
- Retrieval of the system date through operation TIME, to include support for 2-digit-year-data.
- Calling OPM System APIs, to include support for both 2-digit and 4-digit-year data.

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 V3R7 ILE COBOL Date Support

The COBOL ACCEPT statement supports both 2-digit and 4-digit year data.

The COBOL intrinsic functions CURRENT-DATE, WHEN-COMPILED, DATE-OF-INTEGER, DAY-OF-INTEGER, INTEGER-OF-DATE, INTEGER-OF-DAY, and YEAR-TO-YYYY provide 4-digit-year support.

Calling OPM System APIs provides for both 2-digit and 4-digit-year data.

Calling ILE Date APIs provides for both 2-digit and 4-digit-year data.

SQL supports 4-digit-year data.

Using OPM and pre-V3R7 ILE COBOL Date Support

The COBOL ACCEPT statement is supported by the three compilers listed above and supports 2-digit-year data when referencing DATE or DAY.

Calling OPM System APIs is supported by the three compilers listed above and supports both 2-digit and 4-digit-year data. years.

The COBOL WHEN-COMPILED special register is supported in both the OPM and ILE COBOL/400 compilers and supports 2-digit-year data.

Calling ILE Date APIs is supported by the ILE COBOL/400 compiler and supports both 2-digit and 4-digit-year data.

SQL is supported by both the OPM and ILE COBOL compilers and supports 4-digit-year data.

Workstation Interactive Test Tool Year2000 (WITT Year2000) for OS/2

WITT Year2000 for OS/2 is a test tool to assist you in testing of the Year 2000 transformation.

COBOL programmers can use WITT Year2000) as a GUI test tool that automatically facilitates the creation of reusable unit test, function test, system test, and regression test cases. You can create these reusable test cases when you develop or implement a newly developed program, and you should run them 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 you make enhancements and fixes to a program. This automatic playback and comparison capability is best suited for testing of Year 2000 transformation work. Because once you make a modification to an application programs, you should test repeatedly with a variety of system dates such as:

- 1997/02/28 (1997-February-28)
- 1999/12/31 (1999-December-31)
- 2000/01/01 (2000-January-01)
- 2000/02/29 (2000-February-29)
- 2004/02/29 (2004-February-29)

WITT also allow you to add program intelligence to your test cases though scripting in 2/REXX, thus easing your modification of WITT to meet your needs.

WITT Year2000 installs on an OS/2 desktop and allows the testing of OS/390, MVS, VM, IMS, CICS, OS/400, and OS/2 PM and Text (that is, Micro Focus and CICS/OS2) applications.

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 and supports 4-digit-year data.

Calling OPM System APIs is supported by both compilers and supports both 2-digit and 4-digit-year data.

Calling ILE Date APIs is supported by both compilers and supports both 2-digit and 4-digit-year data.

SQL is supported by the ILE C for OS/400 compiler and supports 4-digit-year data.

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 valid date containing a 2-digit year to ILE and ILE returns an integer date 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 supports 4-digit-year data.

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 year, *mm* the month, and *dd* the day. 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 hour, *mm* the minute, and *ss* the second. 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 year, month, day, hour, minute, second, and microsecond. 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.

---

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

- 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 System Values

QCENTURY (Century)

The century (QCENTURY) system value is used to specify the century. It is used with the system value QDATE and QYEAR to determine the specific date currently being used by the system. The possible values are:

- 0 (the years from 1928 to 1999)
- 1 (the years from 2000 to 2053)

You can set the value of QCENTURY with the century indicator, or the system sets the value of QCENTURY based on the following two situations:

- At the time of the first IPL, the system sets the initial value of QCENTURY based on the following rules:
  - If QYEAR is equal to or greater than 40, the system assigns a value of 0 to QCENTURY.
  - If QYEAR is less than 40, the system assigns a value of 1 to QCENTURY.
- When QYEAR or the year in QDATE is changed:
  - QCENTURY is set to 0 if QYEAR is 54 to 99
  - QCENTURY is set to 1 if QYEAR is 00 to 27

For example, if you change QYEAR from 95 to 13, the system changes QCENTURY from 0 to 1, indicating a year of 2013. However, if you change QYEAR from 95 to 45, the system will not change QCENTURY, because both 1945 and 2045 are valid dates.

If you change this value, the change takes effect immediately. Changing this value also affects the system value QDATE.

QDATE (Date)

The system date (QDATE) is used to indicate the year, the month, and the day on the system. This value is made up of the QYEAR, QMONTH, and QDAY system values. The format in which QDATE appears is specified by the QDATFMT system value. You can change the system date. If you change QDATE, the change may affect the system values for QCENTURY, QYEAR, QMONTH, QDAY, and QDAYOFWEEK. Any change you make to QDATE takes effect immediately.

QYEAR (Year)

The system year (QYEAR) is used to specify the last two digits of the year on the system. This value ranges from 0 to 99. The system assigns the first two digits for the year based on the current setting for the QCENTURY system value. If the calculated year falls outside the range of dates supported by the system (1928 to 2053), the QCENTURY system value is changed so that the calculated year is within the supported range.

OS/400 CL

Date parameter types

CL commands which use a date (*DATE) parameter type can be used with both 2-digit years and 4-digit years. In either case the parameter value is a character string passed to the command processing program in the format cyymmdd (c = century digit, y = year, m = month, and d = day).

When using 2-digit year support the system sets the century digit based on the year in the specified date. The century digit is 0 if year equals a number from 40 through 99; it is 1 if year equals a number from 0 through 39.

When using 4-digit year support the system sets the century digit based on the specified 4-digit year where '0' represents years from 1928 to 1999 and '1' for years from 2000 to 2071.

The user must enter the date for the command in the format specified by the date format (DATFMT) job attribute. The date separator (DATSEP) job attribute determines the optional separator character to be used when the date is entered.

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 year, *mm* the month, and *dd* the day.

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 hour, *mm* the minute, and *ss* the second. 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 year, month, day, hour, minute, second, and microsecond. 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.

RTVJOBA (Retrieve Job Attributes)

The Retrieve Job Attributes (RTVJOBA) command is used to retrieve the current attributes of a job so that they can be used in an application program. Year 2000 related job attributes include:

- DATE (providing a 2-digit year)
- CYMDDATE (providing a 2-digit year qualified by a century digit)

RTVSYSVAL (Retrieve System Value)

The Retrieve System Value (RTVSYSVAL) command is used to retrieve the current value of a system value so that it can be used in an application program. Year 2000 related system values include:

- QCENTURY
- QDATE
- QYEAR

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 and simple to learn 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

The IBM COBOL Family

IBM COBOL Family extends over multiple platforms: workstation (OS/2 and AIX) and host (MVS, VM, VSE, and OS/400). All the IBM COBOL compilers were built from the same base product, which gives source and compiler compatibility across multiple platforms.

The MVS & VM COBOL product is discussed in "The IBM COBOL Family for MVS & VM" on page 8-7. The VSE COBOL product is discussed in "The IBM COBOL Family for VSE/ESA" on page 8-33. The OS/400 product is described in "IBM Tools for AS/400" on page 8-43. The OS/2 and Windows and the AIX products are described in "The IBM COBOL Family for the Workstations."

For application code written in COBOL for the workstation or developed for the MVS, VM, VSE, or OS/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 with Windows** 95/NT support
- IBM COBOL Set for AIX

These COBOL compilers—IBM VisualAge for COBOL for OS/2 and IBM COBOL Set for AIX—are available today and feature full Year2000 support. They provide ANSI COBOL Standard Intrinsic Functions which give full date-manipulation capability with 4-digit year support. In addition, IBM VisualAge for COBOL for OS/2 includes a native Windows 95 or Windows NT (2.51) COBOL application development environment.

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. Those tools are available today in VisualAge for COBOL for OS/2 and COBOL Set for AIX.

Year2000 Highlights of IBM VisualAge for COBOL for OS/2

Many 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 VisualAge for COBOL, Professional for OS/2 contains tools helping you identify code that is not Year2000 compliant, analyze the impact of changing that code, then make the change, and recompile your program. Refer to “Redeveloper Tools” on page 8-72 for details about this support as well as support for converting your old level COBOL code to ANSI 85 standard source and for structuring unstructured COBOL programs.

Features of 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. IBM VisualAge for COBOL for OS/2 is now available in two separate editions:

- IBM VisualAge for COBOL, Standard for OS/2 includes the compiler, run-time, visual builder, debug tool, LPEX editor, performance analyzer, and assistant tools (code, data, and transaction).
- IBM VisualAge for COBOL, Professional for OS/2 includes all of the Standard features plus the Redeveloper tools.

Printed and softcopy books are available with both editions along with tutorials and online help.

Remote Edit/Compile/Debug: The remote edit/compile/debug function lets you work with your host (MVS or OS/390) applications from your OS/2 workstation. This function provides a seamless workstation development environment for the development of host programs. Files are directly loaded from the host to the workstation without the need for a workstation naming convention and mapping. Remote edit/compile/debug is integrated with workstation components, such as LPEX and WorkFrame. Remote debugging features the use of a graphical user interface when interacting with the host debug tool. Communications to the host is provided through the Advanced Program-to-Program Communication (APPC) protocol.

ODBC Support: VisualAge for COBOL for OS/2 supports native imbedded SQL statements for Oracle and Sybase. It also includes Open Database Connectivity (ODBC) support that lets you access data from a variety of databases and file systems that support the ODBC interface. The Windows for 95 and Windows for NT support also includes ODBC support.

IBM COBOL Support for Windows 95 and Windows NT: IBM COBOL support for Windows 95 and Windows NT provides you with a 32-bit direct-to-SOM-based, object-oriented compiler and run time for the Windows 95 and Windows NT operating systems. This Windows support also includes a set of high-productivity, Windows-based tools (WorkFrame, the LPEX editor, a debug tool, and Performance Analyzer) for the development of applications targeting Windows 95 and Windows NT execution systems. LPEX, the debug tool, and the Performance Analyzer contain the same functions as the OS/2-based tools.

STL File System: For COBOL I/O support, the STL (STandard Language) file system is supplied for local files. The STL file system enables the processing of files created on other systems. The Windows for 95 and Windows for NT support also includes the STL file system.

MQSeries: IBM's award-winning MQSeries is now supported by VisualAge for COBOL for OS/2. MQSeries enables business applications to exchange information across different operating-system platforms in a way that is straightforward and easy to implement.

MQSeries simplifies the task of connecting your application across unlike environments. Programs communicate using the MQSeries programming interface, which is easy to use and shields programmers from the complexity of different operating systems and the underlying networks. You focus on the business logic, while MQSeries manages your connections to the computer.

Local and Remote Data Access: IBM VisualAge for COBOL for OS/2 provides local and remote data access including:

- ***IBM SMARTdata UTILITIES (SdU) that provide:***
  - 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 Version 3 (this is part of IBM Transaction Server for OS/2 Warp Version 4) or CICS Client for OS/2 Version 2.

Visual GUI Designer: The visual GUI designer lets you build 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. You can create applications by selecting controls from the control palette and 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, you can 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 takes full advantage of the features offered by OS/2. When used as the integration medium for application development tools, the fully configurable WorkFrame/2 increases the effectiveness of these tools as agents for enhancing user productivity. WorkFrame/2 organizes the programmer's workplace by grouping files into logical units or projects. As an organizer, WorkFrame/2:

- Adapts to your 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, (e.g., .EXE or .DLL).
- Associates each project with multiple actions, including compiling, debugging, making, linking, browsing, profiling/analyzing, and preprocessing.
- Allows multiple developers to work concurrently on a single project by plugging in their own source control system.

Note: WorkFrame is also included in the Windows 95 and Windows NT support.

Context Sensitive Editor: The LPEX Editor is a language-sensitive, color editor that supports COBOL. You can use the LPEX Editor 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. You can quickly find items you are looking for in your source code. Using LPEX, you 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.

You can customize and extend virtually every aspect of this programmable editor. You can extend LPEX through dynamic link libraries—there is no proprietary extension language to learn. With the LPEX application programming interface (API), you can write powerful extensions to the editor using C and C++. In addition, LPEX provides a rich command language that you can use to create or modify editor functions. You can:

- Define your 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 your own editor commands.

Note: LPEX is also included in the Windows 95 and Windows NT support.

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 that let you quickly and efficiently control execution and analyze data. You 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.

You can debug CICS for OS/2 Version 3 transactions built with IBM VisualAge for COBOL for OS/2 interactively. (CICS for OS/2 Version 3 is part of the IBM Transaction Server for OS/2 Warp Version 4.)

For PM application programming support, synchronous and asynchronous modes give you two ways to debug PM applications. You can manage the application windows concurrently with the Debug Tool windows.

Note: A debug tool is also included in the Windows 95 and Windows NT support.

Performance Tuning: (Windows version supplied with Windows 95/NT support) Execution trace analysis and performance tuning is provided through the IBM Performance Analyzer. It helps you tune and understand your programs by monitoring program execution and generating a function-by-function trace of the run. This trace can be examined subsequently 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.

Note: The Performance Analyzer is also included in the Windows 95 and Windows NT support.

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 (both local and remote), lets you 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.

Optimization (OPTIMIZE Compiler Option): When activated, the COBOL optimizer generates a more efficient program that helps reduce program run time and amount of main storage used.

DB2 Co-Processor Support: By using the DB2 co-processor, the DB2 pre-processing step has been eliminated which results in better optimization of EXEC SQL statements. The DB2 support is fully integrated with the compiler.

Redeveloper Tools: Use the Redeveloper tools, which are part of VisualAge for COBOL, Professional for OS/2, to re-engineer existing programs to produce better understood and documented applications. The tools aid in understanding, maintenance, and reuse of existing COBOL code inventory, along with, editing, compiling, and debugging new and existing applications. Three Redeveloper tools are included:

- **Application Understanding:** understand application structure and relationship of resources; populate facility for scanning JCL; and analysis facility for determining shared resources.
- **Program Understanding:** understand complex COBOL applications; analyze and extract information about an entire application; and OS/2 graphical user interface to view application physical design.
- **Program Conversion and Structuring:** automate conversion of old source levels to new ANSI 85 source standard; structure code to make it easier to

understand, debug, modify and reuse; generate conversion, analysis, and structuring reports.

IBM VisualAge for COBOL for OS/2 works with TeamConnection to store COBOL source files, build COBOL applications, and provide version control and change management for applications in production.

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 online help.

Year2000 Highlights of IBM COBOL Set for AIX

Many 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

Features of IBM COBOL Set for AIX

ODBC Support: The Open Database Connectivity (ODBC) support lets you use the ODBC interface.

STL File System: For COBOL I/O support, the STL (STandard Language) file system is supplied for local files. The STL file system enables the processing of files created on other systems.

Local and Remote Data Access: IBM COBOL Set for AIX provides local and remote data access including:

- ***IBM SMARTdata UTILITIES (SdU) that 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 that supports COBOL. You can use the LPEX Editor 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. You can quickly find items you are looking for in your source code. Using LPEX, you 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.

You can customize and extend virtually every aspect of this programmable editor. You can extend LPEX through dynamic link libraries—there is no proprietary extension language to learn. With the LPEX application programming interface (API), you can write powerful extensions to the editor using C and C++. In addition, LPEX provides a rich command language that you can use to create or modify editor functions. You can:

- Define your 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 your 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 AIX 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 you 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 you don't have to remember command line invocations of tools. In most cases when you double-click on a file, the CDE desktop automatically launches 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 lets you 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 lets you quickly and efficiently control execution and analyze data. With these core functions, you 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.

Optimization (OPTIMIZE Compiler Option): When activated, the COBOL optimizer generates a more efficient program that helps reduce program run time and amount of main storage used.

DB2 Co-Processor Support: By using the DB2 co-processor, the DB2 pre-processing step has been eliminated which results in better optimization of EXEC SQL statements. The DB2 support is fully integrated with the compiler.

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**
- **IBM PL/I Set for Windows**

Features

These PL/I compilers, IBM PL/I for OS/2, IBM PL/I for Windows, and IBM PL/I set for AIX, are available today and provide full Year2000 support. They provide a built-in function which supports 4-digit-year data.

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 to:

- generate statistics
- modularize, migrate, or restructure code
- search class libraries.

IBM PL/I for OS/2

IBM PL/I for OS/2 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 a Professional Edition.

IBM PL/I for OS/2 Professional Edition: IBM PL/I for OS/2 Professional Edition includes a full 32-bit compiler, run-time library, and a graphical, interactive debugging facility that supports new PL/I application development on stand-alone PCs or small LANs, and 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 the Professional 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 for Windows

IBM PL/I for Windows provides a PL/I application development environment for OS/2 and Windows 95/NT workstations including a complier, language-sensitive editor, debugger, and performance tuner.

IBM PL/I Set for AIX

IBM PL/I Set for AIX 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 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:

- Program Builder
- Debug Tool
- PL/I online documentation

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.
- 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

Refer to the “IBM Year 2000 Solution/Services Database” accessible through the IBM Year 2000 HomePage at URL:

```
http://www.ibm.com/year2000
```

for a list of Solution Developer tools, applications, and services offered by Solution Developers who have represented to IBM that they provide products that are Year2000 ready. These products might be useful to assist your Year2000 analysis, re-engineering, and development when reformatting your year-date notation.

Chapter 9. IBM Consulting and Services

This section includes descriptions of two IBM consulting and service offerings:

- IBM Global Services' (ISG) comprehensive set of software and hardware offerings to assist your Year2000 efforts
- Product Support Services (PPS) (a suite of migration services).

Additionally, for a more comprehensive list of IBM Year2000 offerings and services, you can access “IBM's Year2000 Offerings (all platforms)” at URL:

```
http://www/ibm.com/IBM/year2000/mkt/matrix.html
```

This site offers IBM's broad array of products, services, tools, and free resources to help customers with the major steps in the Year2000 transition process--assessment, planning, implementation, and testing (as well as project management and “clean management”). Within each of these “steps to readiness” categories, the IBM products which apply are listed. A brief description of the product, service, or resource is included.

Global Transformation 2000 Services: IBM's Century Date Change Solutions

IBM recognizes that the Year 2000 change poses a significant challenge for the Information Technology industry. To help with this change, IBM Global Services (IGS) 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 **Global Transformation 2000 Services**.

IBM can help control costs and reduce risks of a Year 2000 conversion with a prescriptive methodology, effective technology, and extensive Year 2000 transformation skills and experience.

Global Transformation 2000 Services' Methodology

Global Transformation 2000 Services' methodology is founded on the legacy system transformation principles that have proven to be effective in more than ten years of IBM customer engagements. To meet the needs of complex Year 2000 conversions, IBM has leveraged and significantly expanded upon this legacy methodology.

Managing the Conversion Process

IGS' Year2000 Offering methodology initially calls for date impact analysis of many areas of a business, including the IT infrastructure, host environments, distributed midrange systems and desktop PCs. This review can also expand beyond the IT infrastructure to include non-IS assets.

Global Services analyzes a company's operations to determine the size and scope of the Year 2000 challenge, then proposes solutions for resolving those needs. The review spans the complete environment with respect to the current IT strategy.

In addition, Global Services can identify instances where complex, outdated systems can be replaced or modified to reduce IT costs. This implementation uses tools that can simplify your maintenance workload long after the Year 2000 conversion is complete.

Next, Global Services identifies date occurrences on a partition-by-partition basis. This involves locating primary, secondary, and file/database dates. Once the date fields in a partition have been identified, a detailed work plan is produced to take that partition through implementation. The work plan identifies the tasks associated with making the changes and identifies the testing strategy to be used. It can also identify the bridges that should be built as other partitions are reviewed throughout the process.

Changing and Testing Code

Global Services has developed a 'conversion factory' approach for changing and testing Year 2000 code and data. That is, tools and tasks are packaged into an integrated production line for implementing necessary Year 2000 changes. This provides you a mobile, customizable environment that is then established either on your facilities or off-site for cost-effective implementation.

The conversion factory approach streamlines the process and gets code back into production in a quality manner.

Clean Management

A unique set of Clean Management activities can help establish standards and procedures to keep recurring Year 2000 problems from affecting your ongoing system development and maintenance operations. Global Services can help you define standards to provide Year 2000-compliant date formats. Further, assistance is available to integrate concurrent production changes with Year 2000 changes. This process, from readiness assessment to methodology education and compliance monitoring, is tailored to meet your individual needs.

Project Office

With Project Office activities, IBM can help set up an overall management support structure for efficiently implementing a multi-location or worldwide Year 2000 solution. The Project Office structure, tailored to your needs, helps resolve the technical and business issues related to a Year 2000 transformation.

Proven Project Management Techniques

All of these activities are grouped into a comprehensive set of Global Transformation 2000 Services phases with more than 1,000 inter-related steps. Proven project management techniques are incorporated in each phase, using a variety of automated tools to help manage the entire project.

Automated Technologies

Some elements of a Year 2000 transformation are critical and complex. Others are highly labor-intensive. IBM provides automation tools that can increase accuracy and shorten the cycle time for both. IBM provides effective Year 2000 automation tools to improve overall productivity.

The process starts by implementing IBM's Redevelopment Assistant (RA) collection of tools. Initially created for legacy transformation impact analysis, these tools have been considerably enhanced to provide Year 2000 program inventory, partitioning, and code analysis functions. RA incorporates a repository that maintains a description of systems types and the relationships among them. Even with a very large application portfolio covering multiple sites and system types, it can usually be contained in a single RA repository.

Inventory Analysis Tools identify application components such as source programs, executable programs, jobs, transactions, files, databases, and data elements. RA tools can also build an extensive database showing the relationships among your system components, even if spread across multiple sites or multiple system types such as host system, midrange, and desktop PC.

Impact Analysis Tools employ powerful pattern matching and flow tracing technology to locate non-compliant dates and the components impacted by those dates. Impact Analysis also reveals which applications, sites or business units have the most date-sensitive impacts.

Partitioning Tools divide large, complex projects into manageable sub-projects that can be performed at multiple sites, across multiple platforms, and by multiple project teams. Partitioning schedules the most urgent work first, while using knowledge of component relationships to minimize bridge building.

Estimating Tools use information from Inventory Analysis, Impact Analysis, and Partitioning to generate detailed estimates for changing source code, converting production data, and testing date-impacted components for correct Year 2000 operation.

Other tools that help with the Year 2000 conversion include:

- Source Code and Date Change Tools
- Change Management Tools
- Testing Tools

IBM's Global Centers of Competency

IBM has created a unique organization dedicated to the management of Year 2000 solutions. This organization, the Year 2000 Center of Competency (CoC) - U.S., is responsible for continually evaluating new tools from around the world and from within IBM. A dedicated group of IBM technology professionals perform a review on tools entering the marketplace.

Managing the growth and refinement of our methodology is another responsibility of the CoC. As IBM engages customers (representing a wide variety of environments and special situations), specific input that can potentially enhance Global Services' methodology is directed to the CoC for evaluation. Professionals in the CoC then evaluate these processes to continually enrich the methodology.

IBM Centers of Competency are located throughout the world to deliver IBM's comprehensive IGS' Year2000 Offering solution worldwide. With this network of IBM professionals, we offer consistent, worldwide application of the Global Transformation 2000 Services' methodology. All of which helps IBM's multi-national clients to receive the global project management required for success.

Further information on Global Transformation 2000 Services' solution set is available from your IBM representative.

Global Services Test Support Services

Global Services' 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.

Global Services offers a testing service for 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. Global Services offers a testing package which provides a thorough Year2000 solution test.

Global Services will work with you during all phases of your Year2000 project. Global Services will:

- Develop a test plan with test objectives for each of your test levels:
  - Integration tests
  - System tests
  - User acceptance tests.
- Understand your total conversion environment and advise you how to create a test environment for it
- Design and execute test scripts and scenarios that 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

Global Service has experience on multiple platforms and environments. Global Services can help you define and meet your Year2000 conversion objectives relating to function, compatibility, and capacity from your host application needs to those of your distributed network.

Benefits to Your Organization

You can:

- Continue to use your Year2000 modifications, allowing Global Services 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

Global Services provides a range of services to support the testing environment including:

- Test process evaluation
- Test management consulting
- Stress testing
- Usability testing.

Contacts

For more information on Global Service's Year2000 testing services, contact your Global Service service representative or Global Services at:

Figure 9-1 (Page 1 of 3). IBM's Year2000 Services - International Addresses and Contacts

Country	Mailing Address	Electronic Address
AUSTRIA	IBM Austria Ing. Ewald Rotter Application Development Consulting Obere Donaustrasse 95 A-1020 Vienna AUSTRIA	Tele: 43 + 1 + 21145-6818 Fax: 43 + 1 + 21145-2393 Internet: Ewald_Rotter@at.ibm.com
BELGIUM	IBM Belgium Mr. Peter Quick Square Victoria Regina 1 1210 Brussels BELGIUM	Tele: 32-2-225.25.59 Fax: 32-2-225 22 40 Internet: PETER_QUICK@BE.IBM.COM
DENMARK	IBM Denmark A/S Mr. Carsten Brogger Andersen Nymollevej 91 DK-2800 Lyngby DENMARK	Tele: +45 45233000 Fax: +45 45874438 Internet: CBA@VNET.IBM.COM
DUBAI, United Arab Emirates	IBM Dubai Nabil Ghorayeb c/o UCMC P.O. Box 9226 DUBAI, UAE	Tele: (9716)-535333 Fax: (9716)-535363 Internet: nabilg@vnet.ibm.com
FINLAND	Oy International Business Machines Ab Tero Lietsala P.O. Box 265 FIN-00101 Helsinki FINLAND	Tele: +358-9-459 4690 Fax: +358-9-459 5303 Internet: tero_lietsala@fi.ibm.com
FRANCE	IBM France Jean-Louis Combeau 1 Place Jean Baptiste Clement 93881 Noisy le Grand Cedex FRANCE	Tele: 01 49 31 43 97 Fax: 01 45 92 90 89 Internet: JLCOMBEAU@vnet.ibm.com

Figure 9-1 (Page 2 of 3). IBM's Year2000 Services - International Addresses and Contacts

Country	Mailing Address	Electronic Address
FRANCE	IBM France Jean-Charles Andre Tour Septentrion 92066 Paris La Defense FRANCE	Tele: (33) 01.49.05.58.38 Fax: (33) 01.47.68.42.50 Internet: none
GERMANY	IBM Germany Mr. Guenter Holzmueller Anzingerstrasse 29 1671 Muenchen GERMANY	Tele: (49) 89 - 4504 - 2706 Fax: (49) 89 - 4504 - 2670 Internet: hlz@vnet.ibm.com
GREECE	IBM Hellas Mr. Vasilis G. Dimopoulos 284 Kifisias Ave. Halandri 152 32, GREECE	Tele: +30.1.6881295/278 Fax: +30.1.6801302 Internet: GRIBMHGM@IBMMAIL.COM
ISRAEL	IBM Udi Chen IBM House 2 Weizmann st. Tel-Aviv 61336 P.O.B. 33666 ISRAEL	Tele: +972-3-697-8365 Fax: +972-3-697-8838 Internet: udi@il.ibm.com
ITALY	IBM SEMEA Circonvallazione Idroscalo 20090 Segrate (Milano) ITALY	• Angelo Crippa Tele: +39/2/59627075 Internet: angelo_crippa@it.ibm.com • Alberto Gracchi Tele: +39/2/59624790 Internet: alberto_gracchi@it.ibm.com Fax: +39/2/59629192
THE NETHER-LANDS	IBM Netherlands NV Year 2000 1423ND Uithoorn Watsonweg 2 -- or -- Postbus 24 1420AA Uithoorn THE NETHERLANDS	Tele: 00 31 (0)79 3227601 Fax: 00 31 (0)79 3228639 Internet: • Engagement Manager: PvdGiesen@ibm.com.nl • Delivery Manager: Jan_Koster@ibm.com.nl
NORWAY	IBM Norway AS Mr. Geir Boen Rosenholmvn. 25 P.O. Box 500 1411 Kolbotn NORWAY	Tele: +47-66999505 Fax: +47-66999333 Internet: GEIR_BOEEN@VNET.IBM.COM
SAUDI ARABIA	IBM Saudi Arabia Mohammed Thiab, SRM P.O. Box 476 Al-Khobar 31952 SAUDI ARABIA	Tele: +966-3-857-1172 X 246 Fax: +966-3-857-0882 Internet: mt@vnet.ibm.com

Figure 9-1 (Page 3 of 3). IBM's Year2000 Services - International Addresses and Contacts

Country	Mailing Address	Electronic Address
SLOVENIA	IBM Slovenia TR/3, 1000 Ljubljana SLOVENIA	• Spela Urh Tele: +386 61 176 3652 Internet: SPELA_UP@IBM.COM • Vesna Eibel +386 61 176 3674 VESNA@VNET.IBM.COM • Saso Prek +386 61 176 3644 SPREK@VNET.IBM.COM Fax: +386 61 125 53 38 +386 61 125 21 19
SPAIN	IBM Spain Jose Martin Faba Santa Hortensia, 26-28 28002-Madrid SPAIN	Tele: 34-1-3975797 Fax: 34-1-5193987 Internet: JMFABA@ES.IBM.COM
SWEDEN	IBM Sweden Mr. Peter Johnson S-164 92 Stockholm SWEDEN	Tele: +46 8 7932230 Fax: +46 8 7932403 Internet: peter_johnson@se.ibm.com
SWITZER-LAND	IBM Switzerland Theodor Gomringer Baendliweg 21 (BW2W) CH-8010 Zuerich SWITZERLAND	Tele: 01 643-6028 Fax: 01 643-6494 Internet: go2000@ch.ibm.com
TURKEY	IBM Turk Ltd. Ozcan Kutluata Buyukdere Cad. Levent 80613 - Istanbul TURKEY	Tele: 90 - 212 - 2800900 (X 1358) Fax: 90 - 212 - 2780437 Internet: none
UNITED KINGDOM	EMEA Year 2000 Centre of Competence Mr. Ian Baker G2NE, 144, Hursley UNITED KINGDOM	Tele: 01962-815014 Internet: ian_baker@uk.ibm.com
UNITED STATES of AMERICA	IBM Global Services (Testing Services) 1510 Page Mill Road Palo Alto, CA 94304 UNITED STATES OF AMERICA	Tele: 1-800-690-4772 Fax: (415) 855-3215 Internet: ISSCTEST@VNET.IBM.COM

IBM Product Support Services (United States only)

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 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 9-2 on page 9-9 lists several priced, platform-specific Product Support 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 9-2. IBM Product Support Services and Their Supported Platform(s)

Platform	Product Support Service	Page
AIX	IBM AIX Upgrade Services	9-10
AS/400	IBM AS/400 SoftInstall	9-11
AS/400	IBM AS/400 SysMigration	9-12
CICS	IBM CICS Application Rehosting Services	9-13
OS/390 MVS MVS/ESA VM/ESA VSE/ESA	SoftwareXcel Installation Express	9-14
AIX Lotus Notes OS/2 RISC ⋮	IBM SmoothStart & Migration	9-15
General Support	IBM House Call	9-17
IBM and non-IBM	IBM Business Recovery Services	9-18

IBM AIX Upgrade Services

IBM AIX Upgrade Services provides a range of AIX program support to meet your 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

Benefits

- IBM specialists to help determine your AIX upgrade requirements
- Assistance with installation and configuration considerations
- Multiple upgrade options to address your unique needs

Contacts

- IBM Direct: 1-800-IBM-CALL (1-800-426-2255)
- USA IBMLink users can receive more information on Year2000 by contacting their client representative or calling: 1-800-IBM-4YOU (1-800-426-4968)

IBM AS/400 SoftInstall

The AS/400 SoftInstall program provides 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
- 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 (1-800-426-4968)
- 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 (1-800-426-4968)

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

- Experienced 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 (1-800-426-4968)

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

- Reduces your system programming staffs' time spent in software upgrading tasks, and relieves your programming staff of the time spent researching and applying services recommended with an upgrade.
- Reduces dedicated computer time and system disruption because a significant amount of the upgrade work is performed by IBM.
- Permits your programming staffs to dedicate their skills to maintaining and enhancing the business functions of the I/S systems.
- Offloads the installation and integration of software products and services (IBM and non-IBM) 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 Product Support 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
- 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 (1-800-426-4968)

IBM Business Recovery Services (BRS)

Responding to the Year2000 challenge might require changes to your application portfolio. Follow-on testing of those applications is required to determine that the systems will run properly and handle both 19xx and 20xx data correctly. IBM's Business Recovery Services provides Year2000 Test Services for this critical part of your Year2000 work effort. You can use IBM's BRS extensive computing facilities to conduct off-site testing, thereby minimizing the disruption to your existing production environments. Year2000 Test Services are conducted on IBM's Year2000-ready platforms that employ advanced system clocks for both the required hardware and software.

BRS can also provide support services to load your tapes and operating systems at IBM's site to assist in your test efforts. Test time can be scheduled on a short-term basis (for example, a 24-hour block) or over an extended period of time as required.

BRS test facilities are located at IBM's 14 geographically dispersed BRS Recovery Centers across the United States. (The largest centers are at Sterling Forest, New York and Boulder, Colorado.) All of these facilities provide Year2000 Test Services. You can request reservations to meet your test requirements.

Platforms

IBM and non-IBM platforms including: mainframes, Tandem, Digital, SP2, AS/400, RS/6000, and PCs.

Benefits

- A convenient and affordable environment to test your Year2000 compliant applications.
- An entirely separate test environment for your production system. No compromise of your production computing environment and/or related expensive downtime for testing.

Contacts

For more information and to schedule test time, call: 1-800-599-9950

IBM Product Support Services (Europe, Middle East, and Africa Only)

SystemCheck 2000

SystemCheck 2000 is a unique service that takes a 'snapshot' of your currently installed IBM System/390 software and processors. This snapshot can then be used to access your overall IS environment in relation to Year2000 readiness.

This services allows you to identify your unique IBM software system environment (**excluding applications**) on an IBM-provided input form. IBM researches your software and processor components and responds with a hardcopy analysis of your environment that contains:

- Solution Developer software list of currently-installed versions and releases, their Year2000 readiness, or the version/release you require.
- IBM software list of currently installed versions and releases, their Year2000 readiness, or the version/release you require.
- IBM & non-IBM processor model Year2000-ready assessment.

SystemCheck 2000 is an IBM fee service.

Platform

- MVS
- VM
- VSE

Benefits

- Understanding the implications of your Year2000 Challenge
- Detailed Report on IBM software system Year2000-ready status
- Offload of a cumbersome, administrative task to IBM
- Savings of time and human resource.

This service is a core component of the Entry Service Offering, other S/390 packed offerings, available standalone, or in a Phase I engagement.

Contacts

- Your local IBM representative.

CICS for MVS/ESA Time Machine

CICS Time Machine is a packaged service offering from IBM Hursley Services and Technology that brings new advantages to CICS V3, CICS V4, and CICS Transaction Server for OS/390 customers. CICS Time Machine provides an easy way to alter time as it appears to on-line transactions; thus opening up a range of benefits to test environments.

Possible Uses

Changes resulting from the use or integration of new technology such as EDI make new demands on critical on-line transactions. To ensure that code alterations made in an critical on-line transaction are fully tested before deployment, you will find the need to test using future dates for Year2000 testing.

As the year 2000 approaches, many IT operations plan to 'bring the future forward' in their CICS test regions to ensure critical transactions function properly after 1999-December-31. CICS Time Machine simulates future or past dates and times presented to a transactions at the CICS API level and allows CICS transactions to experience the future or past by forwarding or reversing the date. This allows testing of date-dependent logic routines to reveal discrete faults which otherwise remain hidden. The ability to 'alter time' provides you the flexibility to perform Year2000 testing at convenient and non-disruptive occasions.

Contacts

- A fee-based Service Offering based on this SupportPac is available from IBM Hursley Services and Technology. Their contact address is:
  - IBM Hursley Services and Technology
    IBM United Kingdom Ltd.
    Hursley Park
    Winchester
    Hampshire SO21 2JN
    England

    Tele: (44) 1962-816211
    IBM Network: INNOVATE at WINVMD
    IBM Mail: GBIBIQ3K at IBMMAIL
    E-mail: innovate@uk.ibm.com
    URL: http://www.hursley.ibm.com/cics/txppacs/ch15.html
- The CICS Time Machine time/date adjustment test tool is available by IBM as a Freeware SupportPac on the Internet. The SupportPac can be downloaded from the following WWW page:

  `http://www.hursley.ibm.com/cics/txppacs/ch15.html`

Appendix A. Year2000-Readiness Status of Selected IBM Program Products and Hardware

Appendix A

The Appendix A listing has been obsoleted by the "IBM Year 2000 Product Readiness Database" accessible through the IBM Year 2000 HomePage at URL:

`http://www.ibm.com/year2000`

The database allows you to select software and hardware products by platform, operating system, and industry, and limit the search to a specific product name or machine type/model or product number. You can request and receive a readiness report on the status of products that match your search criteria. "Readiness Reports" provide information such as the Year2000 readiness status of specific products, the principal technique used to achieve readiness in a given Year2000-ready product, recommended upgrades or replacements for not ready products, and other useful information. This report can be mailed to your e-mail address, often within minutes.

If you do **not** have World-Wide Web access and have IBM product-specific Year2000 readiness questions, you can get help in obtaining a "Readiness Report" by contacting an IBM Technical Support Center. Refer to "About This Book" on page ix for how to contact the IBM Technical Support Center supporting your geographic area.

Appendix B. Year2000-Ready Solution Developer Products

Editor's Note

This appendix has been obsoleted by the "Non-IBM Year 2000 Ready Vendor Applications, Tools, and Services" database accessible through the IBM Year2000 HomePage at URL:

`http://www.ibm.com/year2000`

This database of non-IBM Year2000-ready applications, tools, and services provides information on the readiness of IBM Business Partners, independent software vendors, systems integrators, tools vendors, and other information technology companies that have skills and offerings for customers to address the Year2000 challenge.

The database is designed to help customers and the IBM field force find vendors having Year2000 skills and offerings. It is part of IBM's effort to urge customers to accelerate their transitions and to help them migrate to Year2000-ready products. Customers should contact vendors directly to determine the appropriateness of a specific solution. For your convenience, many solution developer product entries are also provided with a 'hot link' to the solution developer's own HomePage.

The information in the database has been provided to IBM by Business Partners and other companies. It is provided by IBM as a service to customers on an as-is basis. IBM does not make any representations or warranties with respect to the solutions described in the database or the accuracy of the information provide by the listed companies.

Appendix C. Bibliography

Non-IBM Publications

By Author

1. Adam, Gerhard, *Year 2000: Starting the Assessment Process* Technical Support, 1996-Jun, p. 44.
2. Alter, Allan E., *The 2,000-year-old IS manager*, Computerworld, 1996-Apr-01, p. 37.
3. Appleton, Elaine L., *Call in the Cavalry Before 2000*, Datamation, 1996-Jan-01, p. 42.
4. Arnold, Robert S., *Millennium Now: Solutions for Century Date Change Impact*, Application Development Trends, 1995-Jan, pp. 60-67.
5. Arnold, Robert S., *On Assurances of Year 2000 Compliance*, Inside DPMA, Data Processing Management Association, 1996-Jan.
6. Arnold, Dr. Robert S., *Resolving Year 2000 Problems in Legacy Software*, presentation at Software Quality Week, San Francisco, CA, 1995-Jun-01.
7. Arnold, Robert S., *Tips for Performing A Software Inventory to Resolve the Year 2000 Problem*, Inside DPMA, Data Processing Management Association, 1996-Jan.
8. Barker, Paul, *End of Century Problem Looms As IS Remains Idle*, Computing Canada, Vol. 19, No. 23, 1994-Nov-08, pp. 1-4.
9. Barker, Paul, *Study Shows Impact of Year 2000*, Computing Canada, 94-47112, Vol. 19, No. 26, 1993-Dec-20, pp. 1-7.
10. Bartholomew, Doug, *The Year 2000 Problem: Time's Running Out*, InformationWeek, 1996-Feb-05, p. 30.
11. Baum, David, *Tool Up for 2000*, Datamation, 1996-Jan-01, p. 49.
12. Baum, David, *Union Pacific Stays on Track for 2000*, Datamation, 1996-Jan-01, 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, 1994-Jul-11, pp. 1-28.
15. Betts, Mitch, *IBM Pledges Year 2000 Update in '96*, Computerworld, Vol. 29, No. 46, 1995-Nov-13, p. 79,87.
16. Betts, Mitch, *Users Slow to Face Year 2000 Conversion*, Computerworld, Vol. 29, No. 15, 1995-Apr-10, p. 73.
17. Bloom, Al, Managing System Development, 1996-Jan.
18. Bohner, Shawn A. and Robert S. Arnold, *Software Change Impact Analysis*, First Edition, 1996-Jun IEEE Computer Society Press, ISBN 0-8186-7384-2, pp 378.

19. Butler, Ken, *Can Your Computer System Handle the Change of the Century?*, Rough Notes, Vol.138, No. 9, 1995-Sep, p. 28.

20. Celko, Joe, *Start Fixing Year 2000 Problems Now*, Datamation, 1996-Jan-01, p. 36.

21. Chandler, David L., *Century mark is system downer*, Boston Globe 1996-Jun-23, p. 1. The Boston Globe is available on the Internet at: *http://globe.com*

22. Chandrasekaran, Rajiv and Fern Shen, *In Government. Computer Date Crisis is a Byte in the Budget*, The Washington Post, 1997-April-15, p. C01.

23. Cini, Al, *System Bug of the Apocalypse*, Internetwork, 1995-Jan.

24. Cohn, Michael B., *Dateline 1999: IS Pros Retire in Droves*, Computerworld, Vol. 29, No. 44, 1995-Oct-30, p. 37.

25. Cohn, Michael B., *No Need to Fear the Year 2000*, Computerworld, Vol. 28, No. 47, 1994-Nov-21, p. 35.

26. Cohn, Mike, *Year 2000? Shut Up Already!*, Midrange Systems, 1996-Jun-14, p. 70.

27. Cox, Brian, *Year 2000 is Problematic for Systems*, National Underwriter (Life/Health/Financial Services), Vol. 98, No. 42, 1994-Oct-17, pp.7, 28.

28. Cox, Brian, *Year 2000 is A Big Problem for Systems*, National Underwriter (Life/Health/Financial Services), Vol. 98, No. 47, 1994-Nov-21, pp.9, 18. Davis, Lanny J. *A Look At the Computer Calendar Crisis, Countdown to a Meltdown*, Sunday Washington Post (Outlook section), 1996-Sep-15, p c3.

29. de Jager, Peter, *Collateral Damage Will Surface*, Computer World Canada, Vol 12, No. 1, 1995-Jan-19, p. 6.

30. de Jager, Peter, *Companies come clean*, Computerworld, Vol 29, No. 47, 1995-Nov-20, pp. 97-100.

31. de Jager, Peter, *Computer D-Day: Jan 1, 2000*, Toronto Star, 1994-Jun-16.

32. de Jager, Peter, *Do your Systems have only 5 years to live?*, Best's Review (Life/Health), Vol. 96, No. 2, 1995-May, pp. 88-90.

33. de Jager, Peter, *Doomsday 2000*, Computerworld Vol. 27, No. 36, 1993-Sep-06, pp. 105-109.

34. de Jager, Peter, *Embrace the Future, let go of the Past*, Information Canada, Vol. 19, No. 12, 1994-Dec, pp. 5.

35. de Jager, Peter, *Feedback Highlights bad BIOS Blues*, Information Canada, Vol. 20, No. 2, 1995-Feb.

36. de Jager, Peter, *If you start now ... you just might make it* Computerworld, Vol. 29, No. 47, 1995-Nov-20, pp. 97-100.

37. de Jager, Peter, *Only you can Decide*, Information Canada, 1995-Jun.

38. de Jager, Peter, *Start now or face 2000 Doom*, Datamation 96-73255, Vol. 41, No. 9, 1995-May-15, p.92, 11.

39. de Jager, Peter, *Take a reporter out to lunch*, Datamation, 1996-Jan-01, p. 76.

40. de Jager, Peter, *Time's a Wasting and The Clocks Keep on Tickin'*, Info Canada, Vol. 18, No. 5, 1993-May, p. 7.

41. de Jager, Peter, *2000 or Bust*, CA Magazine, Vol. 127, No. 6, 1994-Aug, pp.32-33.

42. DeVoe, Deborah, *Still No Quick Fix to Year 2000 Glitch*, InfoWorld Magazine, Vol. 17, No. 48, 1995-Nov-27, p. 46.

43. Dunn, Robert H., *Software Defect Removal*, McGraw-Hill, Inc., 1984.

44. Edwards, John, *New Year's Evil*, CIO Magazine, 15, 1995-Dec, p. 82.

45. Elms, Teresa, *Global Trends Shape Midrange for 2000*, System 3X/400, Vol. 20, No. 12, 1992-Dec, pp. 44-50.

46. Farber, Arnold and Rosemary LaChance, *Time Slips Away: Year 2000 is Closer Than You Think*, Enterprise Systems Journal, 1996-Feb, p. 42.

47. Fine, Doug, *Companies Brace for Millennium*, Infoworld, 1995-Apr-10.

48. Furman, Jeff, Albert Marotta, and Cliff Candiotti, *Party When It's 1999*, Software Magazine, 1995-Apr, pp. 6-8.

49. Furman, Jeff and Albert Marotta, *Year 2000 Denial*, Computerworld, Vol. 28, No. 43, 1994-Oct-24, pp. 70.

50. 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, 1993-Feb-08.

    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 1995-Dec-28.

    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 1995-Nov-29.

    d. Jones, N., *PCs and the Year 2000*, Gartner Group, Applications Development and Management Strategies (ADM), Research Note, Strategic Planning, SPA-980-1278, 1996-Jan-30.

    e. Phelps, John, *Year 2000 - Known but Not Understood?*, Gartner Group, Large Computer Research Note, Strategic Planning, SPA-800-1734, 1995-Jul-26.

    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, 1995-Apr-12.

    g. Schick, K., *INSPECT Legacy Applications for the Year 2000*, Garter Group, Application Development & Management Strategies (ADM), Research Note, Strategic Planning SPA-215-1146, 1995-Apr-12.

    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, 1995-Jan-25.

    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, 1995-Nov-29.

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, 1995-Jul-25.

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, 1995-Aug-07.

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, 1996-Feb-09.

51. Glass, Brett, *Turn of the Century Will Bring Out Flaws in Many Programs*, InfoWorld, Vol. 17, No. 25, 1995-Jun-19, p 42.

52. Glass, Robert L., *The Date Wars: Learning Management by Issue*, Information Systems Management, Vol. 12, No. 2, 1995 Spring, pp. 67-69.

53. Gleick, James, *Oh-Oh*, New York Times Magazine, 1996-Jun-02, p. 19. Gonick, Larry, *The Birth of a Computer Catastrophe*, Newsweek, 1997-June-2, p. 55.

54. Goodwin, Bill, *Tick, Tick, Tick...*, (newsletter), 2000AD, Inc., Brooklyn, NY.

55. Governor, James, *UK Post Office Sorts Out The Millennium*, Computing (UK), 1996-Jun-27.

56. Greiner, Lynn, *Year 2000 a Time Bomb Ready to Explode*, Computing Canada, Vol. 21, No. 14, 1996-Jul-05, p. 11.

57. Hafner, Katie, and Deborah Branscum, *Will My Home PC Die*, Newsweek, 1997-June-2, p. 59.

58. Hoffman, Thomas, *1,418 days and counting - CA offers year 2000 date-change products*, Computerworld, 1996-Feb-12.

59. Hayes, Brian, *Waiting for 01-01-00*, American Scientist, Volume 83, 1995-Jan/Feb.

60. Jones, Capers, *Patterns of Software Systems Failure and Success*, International Thomson Computer Press, 1996. ISBN:1-850-32804-8.

61. Jones, David C., *Solving Year 2000 Problem Long, Costly Project*, National Underwriter (Life/Health/Financial Services), Vol. 99, No. 26, 1995-Jun-26, pp.9-24.

62. Kappelman, Leon, and James Cappel, Journal of Systems Management, 1996-Aug-22, p 4.

63. Levy, Stephen, *The 1,000 Year Glitch*, Newsweek, 1996-Jun-24, p. 92.

64. Keuffel, Warren, *Coping with the Year 2000 Rollover* Software Development Magazine, Vol. 4, No. 8, 1996-Aug.

65. Kilmar, and Wasserman, *Pros/Cons Procedure vs. Data*, American Programmer, 1996-Feb.

66. Lefkon, Dick, *YEAR 2000: Best Practices for Y2K Millenium Computing: Panic in Year 2000*, AITP SIG-Mainframe, 1996, 1997.

67. Lefkon, Richard G., *Nuclear Disaster and the Millennium Trojan Horse*, 14th National Computer Security Conference Proceedings Vol II, Information

Systems Security Requirements and Practices, Washington, D.C., 1991-Oct-1/4.

68. Levy, Steven, and Katie Hafner, *The Day the World Shuts Down*, Newsweek, 1997-June-2. pp. 53-59.
69. Lips, Michael, *Six-Digit Dates And The Century Change: Complex Problem. Simple Solution?*, Enterprise System Journal, 1993-Oct, pp.68-69.
70. Marcoccia, Lou, Managing System Development, 1996-Jan.
71. Martin, James, *Information Engineering, Book I Introduction,*, Prentice Hall, 1990.
72. Martin, James, *Information Engineering, Book II Planning and Analysis*, Prentice Hall, 1990.
73. Martin, James, *Information Engineering, Book III Design and Construction*, Prentice Hall, 1990.
74. Martin, James and Joe Leben, *Strategic Information Planning Methodologies*, Second Edition, Prentice Hall, 1989.
75. Martin, James and Carma McClure, *Structured Techniques The Basis for CASE*, Revised Edition, Prentice Hall, 1988.
76. Martin, Larry W., *Millennium Preparation*, Software Magazine, Vol. 13, No. 10, 1993-Jul, pp.6-8.
77. McCarthy, Vance, *Keep the Millennium Virus off Your Net*, Datamation, 1996-Jan-01, p. 55.
78. McColgan, Declan, *Countdown to 2000*, Irish Computer, 1996-Feb.
79. McKendrick, Joseph, *A Once in a Century Crisis*, Midrange Systems, Vol. 8, No. 12, 1995-Jun-30, pp. 17-18.
80. McKendrick, Joseph, *The Year 2000: Still Way Off*, Midrange Systems, Vol. 8, No. 19, 1995-Oct-13, p. 3.
81. Meador, C. Lawrence, *Solving the Year 2000 Problem*, InformationWeek, 1996-Feb-05, p. 44.
82. Meall, Lesley., *The Century's Time Bomb*, Accountancy, Vol. 116, No 1228, 1995-Dec, p. 52.
83. Miller, Edward and William E. Howden, *Tutorial: Software Testing & Validation Techniques*, Second Edition
84. Murray, Jerome. T. and Marilyn. J. Murray, *The Year 2000 Computing Crisis: A Millenium Date Conversion Plan* (formerly titled *Computers in Crisis*), McGraw-Hill, 1996.
85. Neumann, Peter, G., *Computer Related Risks*, Addison-Wesley, ISBN 0-201-55805-X.
86. Nocera, Joseph, *The Story of '00*, Fortune, 1996-Aug-19, pp 51-52.
87. Olsen, Florence, *Govt. Urged to Perform Triage in Two-Digit Field Repairs*, Government Computer News, 1996-Mar-18.
88. Ouelette, Tim, *Insurer 'lies' to avoid year 2000 costs*, Computerworld, 1997-July-28, p. 16.
89. Perlman, Ellen, *Techno-Terror 2000* Governing - The Magazine of States and Localities, 1996-Sep.

http://web.governing.com/governing/92000.html

90. Perry, William E., *A Structured Approach to Systems Testing*, QED Technical Publishing Group, 1983.

91. Perry, William E., *Quality Assurance for Information Systems: Methods Tools, and Techniques*, QED Technical Publishing Group, 1991.

92. Peterson, DuWayne, *The Year 2000: Close Enough to be Dealt With*, SIM Network (Society for Information Management), 1996-Feb.

93. Peterson, Ivars, *Fatal Defect*, Times Books, ISBN 0-8129-2023-6.

94. Poland, Tom, *Year 2000 Dilemma Creating Havoc with Software*, National Underwriter (Life/Health/Financial Services), Vol. 98, No. 49, 1994-Dec-05, pp. 38-39.

95. 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.

96. Ross, Noah, *The End of the Century is Nearer Than You Think*, Application Development Trends, 1995-Apr.

97. Sager, Ira, *Glitch of the Millennium*, Business Week, No. 3450, No. 3450, 1995-Nov-13, p. 54.

98. Scheier, Robert L., *Face up to it*, Computerworld, 1996-Mar-25, pp. 83-84.

99. Scheier, Robert L., *Millennium legal costs could top $100 billion*, Computerworld, 1997-July-28, p. 16.

100. Schick, Kevin, *The Year 2000 Date Crisis*, Government Finance Review, Vol. 11, No. 4, 1995-Aug, pp. 32-33.

101. Schulmeyer, Gordon G., *Zero Defect Software*, McGraw-Hill, Inc., 1990.

102. Schwartz, Susana, *Are Insurers Stuck in a Time Warp?*, Insurance & Technology, Vol. 20, No. 5, 1995-May, p. 56.

103. Spiro, Leah Nathans, *Panic in the Year Zero Zero*, Business Week, 1996-Aug-12, pp 72-73.

104. Stedman, Craig, *Utility charges to 2000*, Computerworld, 1996-Feb-18, p. 71.

105. Steward, Donald V., *Software Engineering with System Analysis and Design*, Brooks/Cole Publishing Company, 1987, IEEE Computer Society Press.

106. Stitt, Martin, *Debugging: Creative Techniques and Tools for Software Repair*, John Wiley & Sons, Inc., 1992.

107. Sullivan, R. Lee, *Ghosts in the Machines*, Forbes, 1995-Jun-19.

108. Ulrich, William R., *10 pitfalls to avoid in year 2000 initiatives*, Application Development Trends, 1996-Feb.

109. *What's the ROI on Objects*, Software Magazine, 1996-May.

110. Vandercook, R. Gibbs, *Now Is the Time*, Systems Management, Vol. 23, No. 3, 1995-Mar, p. 66.

111. Violino, Bob, *The Trial of the Millennium* InformationWeek, 1997-April-21, p. 44.

112. Violino, Bob, and Bruce Caldwell, *Ripple Effect*, InformationWeek, 1997-April-21, p. 38.

113. Xenakis, John J, *The Year 2000 Surprise*, CFO: The Magazine for Senior Financial Executives, Vol. 11, No. 8, 1995-Aug, p. 22.

114. *Year 2000 Testing Task Ahead*, Software Magazine, 1996-May.

115. Yourdon, Edward, *Modern Structured Analysis*, Yourdon Press, Prentice Hall, 1989.

116. Zvegintzov, Nicholas, *The Year 2000 as racket and ruse*, American Programmmer, 1996-Feb.

117. Zvegintzov, Nicholas, Managing System Development, 1996-Jan.

By Title

1. *Adpac Unveil Date Conversion Tool*, Computerworld, 1994-Oct-10, p. 81.
2. *ANSI Cobol Standard Modified For The Year 2000*, Information Week, 1990-Feb-26, 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*.
5. *Are You Ready for the 21st Century?*, CrossTalk: The Journal of Defense Software Engineering, 1996-Mar.
6. *Biting the Bullet*, Computerworld, 1996-Mar-18, p. 16.
7. *Cashing in on Year 2000: Reprogramming work, consulting jobs may spell rising stocks for some vendors*, Computerworld, 1999-Sep-09, p 118.
8. *CAP Gemini Seeks To Solve Year 2000 Dilemma*, News 3x/400, 1995-Jun, p. 30.
9. *Computer Horizons Establishes Mini-homepage on de Jager's Year 2000 Information Center Internet Home Page*, PR Newswire Association, 1995-Nov-06, p. 1106NY062.
10. *Computer snafus over the year 2000*, The Kiplinger Washington Letter (Kiplinger Report), 1996-Sep-20.
11. *Debate of the Century, when does the next millennium begin?*, MoneyWorld, 1997-Mar, p 7.
12. *Defusing Millenium Time Bomb*, 1996-Jun. At WWW site: *http://www.us.net/signal*
13. *Firms Race Around the Clock to Avert "Millennium Bug" Bite*, Detroit News, 1995-Dec-03, p. C1.
14. *For better grade in final exam on 2000 conversion, study up*, Government Computer News, 1996-Aug-12. p 1.
15. *For The Unwary, The New Millennium Could Bring Headaches*, Newsbytes News Network, Seattle, WA, 1995-Mar-30.
16. *IBM Counts on Developers, VARs to Ring in Year 2000*, Computer Reseller News, CMP Publications, Inc., 1995-Nov-13, p. 96.

17. *IBM Plans Year 2000 Upgrades*, InformationWeek, CMP Publications, Inc., 1995-Nov-13, p. 30.

18. *IBM Prepares for Year 2000 Date*, Newsbytes News Network, 1995-Nov-02.

19. *Insurer Techies Scrambling For 2000*, National Underwriter, 1996-Sep-09.

20. *IT Begins to Face Year 2000 PC Data Problem*, Newbytes News Network, 1995-Oct-31.

21. ISO 8601:1988 *Data elements and interchange formats -- Information interchange -- Representation of dates and times*, Technical Committee ISO/TC 154.

22. *Keane, VIASOFT Announce Agreement to Provide Complete Year 2000 Solution*, Business Wire, 1995-Nov-04, p11061123.

23. *Mainframe Time Bomb: 01-01-00*, Fortune Magazine, 1995-Mar-06, p. 24.

24. *Micro Focus Offers Year 2000 Fixer*, Computerworld, 1995-Jul-31, p. 8.

25. *Millennium brings a problem* Government Computer News, 1996-Aug-12. pp 67-69.

26. *Millennium could bring computer chaos*, Southam News organization as reported in The Daily News, Halifax, N.S., Canada, 1996-Feb.

27. *The Millennium: Problems and Solutions*, Xephon, p. 173.

28. *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.

29. *New OS/390 handles year 2000*, Computerworld, 1996-Feb-19.

30. *Ottawa Confident of Escaping Year 2000 Rollover Debacle*, Technology in Government, 1996-Feb.

31. *Pain or Gain in the Year 2000?*, Computer Business Review, 1996-Mar.

32. *SEEC Ship COBOL Tool Upgrade*, PC Week, 1994-Nov-21, p. 31.

33. *Seec Upgrades Analysis & Maintenance*, Software Magazine, 1995-Feb, p. 93.

34. *Service for Year 2000 Year Date Processing of Programs Software*, New Technology Japan, 1995-Feb, p. 24.

35. *Something's Got to Give*, New York Times Magazine, 1996-Mar-24.

36. *Surviving the Year 2000*, Cutter Information Corp. (a compilation of recently published writings by: Capers Jones, Peter de Jager, Nicholas Zvegintzov, Gregory Nelson, Thomas S. Klimuc and Roxane M. Wasserman, Michael D. Lips, William M. Ulrich). Information at:

    `http://www.cutter.com`

37. *Le syndrome de l'an 2000*, Editor: Editions Eyrolles, 61 boulevard Saint-Germain, 75240 Paris Cedex 05, FRANCE, ISBN: 2-212-08913-9, pp. 180.

38. *Take a Year 2000 Inventory*, Datamation, 1996-Aug p 14.

39. *Technology Problem Of The Century*, Bank Technology News, Faulkner & Gray, Inc., 1995-Jun, p. 10.

40. *There Is No Feb 29, 2000!*, Datamation, 1996-Apr-01, pp 8-9.

41. *Tick Tock...*, Midrange Systems, 1996-May-24.

42. *Tick, tick, tick. 2000 looms, and the turf battles begin*, Datamation, 1996-Aug, p 14.

43. *Time running out for Year 2000 transition*, Network World Canada, Vol. 6, No. 2, 1996-Feb-02.

44. *Troubled Time*, Associated Press, Denver Post, (and various other USA newspapers), 1995-May-25.

45. *Users Invest in Millennium Changes*, Xephon 1995, Insight IS, pp. 12-13.

46. *Vendors Focus On Year 2000 Conversion Issue*, Computerworld, 1995-Apr-10, p. 73.

47. *VIASOFT and Litton Computer Services Enter into a Strategic Alliance*, Business Wire, 1995-Dec-01.

48. *Viasoft's Unveil Code Decipher Software*, Computerworld, 1994-Aug-15, p. 16.

49. *Worried About 2000? Group Shares The Fretting*, Investor's Business Daily, 1996-Feb-07, p.A6.

50. *Year 2000 causes double trouble for feds (Countdown to Calamity)*, Federal Computer Week, 1996-Apr-01.

51. *Year 2000: Keeping Doomsday Costs Down*, Enterprise Data Center Strategies, Meta Group, Inc., File No. 431, 1994-Dec-29.

52. *The Year 2000 Doesn't Compute*, Electronic Buyers' News, CMP Publications, Inc., 1996-Jan-02, 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.)

2. The Boston Globe Home Page:

   `http://globe.com`

3. Cable News Network - *The year 2000 does not compute*, CNN's Headline News, 1996-Jan-07:

   `http://www.cnn.com/TECH/9601/2000/index.html`

   `http://www.itpolicy.gov.80/library/yr2000/y201toc1.htm`

4. Compliance list links (US Air Force, University of Florida, and AuditServe) at:

   `http://www.deweerd.org/year2000/compliance.html`

5. *Component Test Standard* prepared by the British Computer Society Special Interest Group in Software Testing (BCS SIGIST) at:

   `http://www.rmcs.cranfield.ac.uk/~cised/sreid/BCS_SIG/index.htm`

6. Datamation Home Page (*Year 2000: Fix It Now!*, Special Report: Year 2000, 1996-Jan-01)

   `http://www.datamation.com/PlugIn/issues/1996/jan1/FEATURES.html`

7. Federal Year 2000 Interagency Task Force (draft proposal for “Recommended Year 2000 Contract Language”) at:

8. *The Globe and Mail* (Canada's National Newspaper):

   `http://www.globeandmail.ca/Cyber-17.html`

9. 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 Year 2000 Technical Support Center website at:

   `http://www.software.ibm.com/year2000`

   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
   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:

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

   where **`format_type`** is specific to one of the print formats listed above.

10. IBM DB2 (DATABASE 2) and the year 2000 on the web at:

    http://www.software.ibm.com/year2000/db2.html

11. 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.

12. IBM VisualAge 2000: Year 2000 Methodology and Tools Implementation Web site at:

    http://www.software.ibm.com/ad/va2000/y2k

13. IBM VSE and Year 2000 Home Page

    Access VSE/ESA and Year 2000 at:

    http://www.s390.ibm.com/vse/vsehtmls/vse2000.htm

14. InfoWorld, *Enterprise Computing: Client-Server Year 2000 problems*, 1996-Jun-03, pp.69-71.

15. Information Technology Association of America (ITAA). White paper available at:

    http://www.itaa.org

16. InformationWeek Year 2000 chat session:

    http://www.cmp.com/cgi-bin/techtalk/2000

17. InformationWeek web site:

    http://techweb.cmp.com/techweb/iw/current/

18. 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.)

19. Microsoft website at:

    http://www.microsoft.com/corpinfo

20. Microsoft's Technet (Year2000 article) at URL:

    http:/www.microsoft.com/technet/tnnews/features/2000.htm

21. 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

22. National Bulletin Board for Year 2000 Home Page:

http://www.it2000.com

23. Project 21st. Century Home Page at:

    http://www.webhelp.com/future.htm

24. Questicon Home Page (to include a year 2000 checklist (QCON2000.PPT)).

    http://questicon.com/products.htm.

25. Royal Greenwich Observatory (RGO) (2000 AD, Leap Years,...):

    http://www.ast.cam.ac.uk/RGO/leaflets/

26. The Software Technology Support Center web site (list of over 120 Year 2000 tools and vendors at:

    http://www.stsc.hill.af.mil/~red/index.html

27. *Software Testing On-Line Resource* (list of resources and references to testing-related web sites) at:

    http://www.mtsu.edu/~storm

28. The State of Minnesota (standards for Year 2000 compliance - *Information Resource Performance Standards*)

    http://www.state.mn.us/ebranch/admin/ipo/hb/document/std14-1.html
    http://www.state.mn.us/ebranch/admin/ipo/hb/document/std14-1.html

29. The State of Texas, Department of Information Resources (downloadable publication on the Web concerning the Year 2000) at:

    ftp://sol.stac.dir.state.tx.us/pub/srrpub9.txt

30. *Summary of the International Standard Date and Time Notation*. WWW site at:

    http://www.ft.uni-erlangen.de/~mskuhn/iso-time.html.

31. *TechLink Newsletter* from CMP's TechWeb - The Technology Information Source, 1996-Feb-08.

    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

32. Tick! Tick! Tick! Home Page (to include a list of Year2000 user groups):

    http://www.henterprises.com/tick3/

33. 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

34. United Kingdom database year2000-ready products. Developed by the law firm of Halberstam Elias) in conjunction the CSSA and the IBM Computer Users' Association)

    http://www.weblaw.co.uk.

35. USA Today Web site at:

http://www.usatoday.com/news/comment/colmane.htm

36. U.S. Dept. of Commerce, National Institute of Standards and Technology, 1996-Mar-25. (Change to FIPS Publication 4-1 concerning the format of year information.) NIST WWW site at:

    http://www.itpolicy.gsa.gov:80/library/yr2000/

37. Washington State's Year 2000 Home Page:

    http://www.wa.gov/dis/2000/y2000.htm

38. XPRESS Software, Inc. Home Page:

    http://www.xpsoft.com/

39. Year2000 compliancy statements from Dell, Gateway, Compaq, et al. at:

    http://www.wa.gov/dis/2000/survey/dt_hard

40. Year 2000 Frequently Asked Questions (FAQ) access

    ftp://www.year2000.com/pub/year2000/y2kfaq.txt

41. 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

42. *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

43. Year2000 research report (400 pages) and 42 year2000 vendors (over 100 products). Web site describes the report; lists a table of contents; and lists the vendors profiled at:

    http://www.mstnet.com/year2000/yr2000.htm

44. Year 2000 Software Control

    http://www.iac.honeywell.com/Pub/Tech/CM/CMTools.html

45. Year 2000 Technical Audit Center

    http://www.auditserve.com/yr2000/countdown.cgi?2000,1,0,XX,XX,XX

46. Year 2000 Technical Support Center (IBM)

    http://www.software.ibm.com/year2000

47. Year 2000 Web Links

    http://www.club.innet.be/~janjedsp/y2k.html

48. Yorktown Technologies Inc. (YTI) Home Page (a layman's analogy - “Year 2000 Parable - Explaining The Problem” by Keith Cowan) at:

    http://OurWorld.CompuServe.Com/HomePages/YTI/y2kparbl.htm

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, 1995-May. 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
   ```

4. *Year 2000 Executive Awareness*, Federal Reserve Bank, 1997. A 10-minute video introduced by Edward J. Kelley, Jr., FRB Board member, Board of Governors of the Federal Reserve System. To order, visit the FRB Web site at:

   ```
   http://www.bog.frb.fed.us/u2k/video.cfm
   ```

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. *COBOL for MVS & VM and COBOL 370 Run Time Migration Guide*, GC26-4764
3. *COBOL for VSE/ESA Migration Guide Version 1 Release 1* GC26-8070
4. *DB2 SQL Reference*, SC26-4890
5. *ES/9000 Multi-Image Processing Volume 1*, GG24-3920
6. *ES/9000 Multi-Image Processing Volume 2*, GG24-3921
7. *IBM High Level Assembler/MVS & VM & VSE: Language Reference (Release 1)*, SC26-4940
8. *IBM VSE/Enterprise Systems Architecture (Version 1 Release 3): System Macros Reference*, SC33-6516
9. *IMS/ESA Version 4: Utilities Reference: Database*, SC26-4627
10. Migrating from VSE/ESA V1 to V2 - Why and How?, SG24-4773
11. *National Language Design Guide Volume 2: National Language Support Reference Manual*, Fourth Edition, IBM Corp., 1994.
12. *OS/390 MVS Assembler Services Guide*, GC28-1762
13. *OS/390 MVS Assembler Services Reference*, GC28-1910
14. *OS/390 Planning for Installation*
15. *OS/390 MVS System Commands*, GC28-1781
16. *Preparing Your VSE System for the Year 2000*, SG24-4932
17. *ESA/390 Principles of Operations*, SA22-7201
18. *Systems Application Architecture: Common Programming Interface C Reference - Level 2*, SC09-1308
19. *Systems Application Architecture: Common Programming Interface COBOL Level 2 Reference*, SC26-4726
20. *Systems Application Architecture: Common Programming Interface Language Environment Reference*, SC26-4970
21. *Systems Application Architecture: Common Programming Interface PL/I Reference*, SC26-4381

22. *Systems Application Architecture: Common Programming Interface REXX Level 2 Reference*, SC24-5549

23. *S/390 Time Management and IBM 9037 Sysplex Timer*, SG24-2070

    Also available at URL:

    `http://www.redbooks.ibm.com/redbooks`

24. Taking Advantage of Language Environment for VSE, SG24-4798

25. *Virtual Machine/ Enterprise Systems Architecture: CP Command CP Command and Utility (Release 2)*, SC24-5519

26. *VM/ESA Year 2000 Migration - A Case Study*, SG24-2024

27. *VS COBOL II: Application Programming Language Reference*, GC26-4047

28. *VS FORTRAN (Version 2 Release 5): Language and Library Reference*, SC26-4221

29. *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`.

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`.

20th century. The period of time 0000.00 hrs 1901-January-1 through 2400.00 hrs 2000-December-31.

21st century. The period of time 0000.00 hrs 2001-January-1 through 2400.00 hrs 2100-December-31.

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 an 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. Although IBM recognizes that the 21st century begins at 0000 hrs, 2001-January-01, for purposes of this document, we are defining the 20th—21st century boundary to be between 2400 hrs, 1999-December-31 and 0000 hrs, 2000-January-1. 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. 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 ≥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

integer date. A count of days since a specified date. Various IBM software products have defined integer dates as follows:

Language/Product	Days Since
C	1969-Dec-31
COBOL	1600-Dec-31

Language Environment 1582-Oct-14

MVS/CICS/DB2 1899-Dec-31

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. As a general term used widely in computer programming and this document: 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. For example, 1996-August-29 is 96242.

However, the above definition is accurately referred to as the **Ordinal Day of Year** date, and an accurate definition of **Julian Day Number** is as follows:

The astronomal system that counts the days since the First of January in the year 4713 BCE (the year –4712 before the common era). This scheme was invented by the astronomer Joseph Scaliger in the 16th century and named by him for his father Julius. The leap year reforms implicit in the new scheme were adopted at the command of Pope Gregory XIII in the year 1582 - hence the Gregorian Calendar. The Gregorian, or New Style, calendar was adopted in Britain and their colonies in September of 1752. (September of that year was missing 11 days. The 14th followed the 2nd.)

The remainder left when dividing the Julian Day Number by 7 indicates the day of week of the specified date. Zero corresponds to Monday, 1 to Tuesday, up through 6 for Sunday.

For example, 1996-Aug-29 is equivalent to Julian Day 2450325. Further, the hour of the day is expressed as a decimal such that 2450325.5 is midnight 1996-Aug-29, based on the fact that a Julian Day begins at mid-day (noon).

L

Leap year. A year either evenly divisible by 400 **or** evenly divisible by 4 and not evenly divisible by 100. For example, the years 1700, 1800, 1900, and 1995 are not leap years, but the years 1600, 1996, and 2000 are leap years.

Lilian date. The number of days since 1582-October-14. 1582-October-15 is Lilian day 1, 1582-October-16 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.)

O

Ordinal Day of Year. See **Julian Date**

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.

Solution Developers. Also known as **third-party vendors** and **independent software vendors**. Marketers of computer hardware and/or software products, tools, or services to complement IBM's suite of products and services.

Y

Year2000 challenge. 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 caused by the transition to the year 2000.

Year2000 ready. The capability of a Product, when used in accordance with its associated documentation, to correctly process, provide and/or receive date data within and between the 20th and 21st centuries, provided that all products (for example, hardware, software, and firmware) used with the Product properly exchange accurate date data with it.

Year2000 support. The ability to provide **Year2000 readiness**.

Year2000 transition. The process of revising systems and databases) to correctly process date data both within and between the 20th and 21st centuries.

YY format. Synonymous with **2-digit-year format**.

YYYY format. Synonymous with **4-digit-year format** and a subset of **CCYY format**.

Appendix E. Calendars

To assist your future date testing (day-of-week, week-of-year, and so on), this appendix contains calendars for 1998, 1999, and 2000.

Note: ISO Standard 8601 specifies that:

- A week begins on a Monday
- The first week of a year is the first week containing four of more days (that is, the first week containing a Thursday).

Year 1998

Figure E-1. 1998 Calendar

JANUARY

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

FEBRUARY

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28

MARCH

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

APRIL

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

MAY

S	M	T	W	T	F	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

JUNE

S	M	T	W	T	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

JULY

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

AUGUST

S	M	T	W	T	F	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

SEPTEMBER

S	M	T	W	T	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

OCTOBER

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

NOVEMBER

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30					

DECEMBER

S	M	T	W	T	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

Year 1999

Figure E-2. 1999 Calendar

JANUARY

S	M	T	W	T	F	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

FEBRUARY

S	M	T	W	T	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28						

MARCH

S	M	T	W	T	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

APRIL

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

MAY

S	M	T	W	T	F	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

JUNE

S	M	T	W	T	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

JULY

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

AUGUST

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

SEPTEMBER

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

OCTOBER

S	M	T	W	T	F	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

NOVEMBER

S	M	T	W	T	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

DECEMBER

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

Year 2000

Figure E-3. 2000 Calendar

JANUARY

S	M	T	W	T	F	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

FEBRUARY

S	M	T	W	T	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	**29**				

MARCH

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

APRIL

S	M	T	W	T	F	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30						

MAY

S	M	T	W	T	F	S
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

JUNE

S	M	T	W	T	F	S
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

JULY

S	M	T	W	T	F	S
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

AUGUST

S	M	T	W	T	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30	31		

SEPTEMBER

S	M	T	W	T	F	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

OCTOBER

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

NOVEMBER

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

DECEMBER

S	M	T	W	T	F	S
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

Index

Special Characters

Numerics

A

B

C

D

E

F

G

I

J

L

M

O

P

R

S

T

U

V

W

Y

IBM Year 2000 Solution/Services Survey

Survey form for IBM Business Partners, Non-IBM Systems Integrators, Independent Software Vendors (ISVs), Tools Vendors, and Service Providers.

Is your product ready for the Year2000? Do you market tools designed to assist others in attaining Year 2000 readiness? Do you provide Year2000-related consulting services? If the answers to any of these questions is **YES!** and you would like to nominate your product or service for inclusion in IBM Year 2000 publications and directories, please supply the following information about your company. For each Year 2000-ready product, please provide the product information requested below. If you are an IBM Solution Provider or IBM Systems Integrator and your products and/or deliverables are in-process for Year 2000 enablement, please specify your current plans to make the products and/or deliverables Year 2000 Ready and indicate the anticipated date by which they will be Year 2000 Ready.

By completing the First name, Surname, Title and Company fields below and submitting this form, your organization certifies that all of the information provided in this survey is true and complete. You also authorize IBM to reference your company's name, address, and phone number, along with the Year 2000-ready products and registered trademarks in IBM publications and directories, including but not limited to, *The IBM Year 2000 Solution and Services Directory* and the *Year2000 and 2-Digit Dates: Guide for Planning and Implementation*.

Electronic Form Submission

Please access and submit this form electronically through the IBM Web Site at URL:

`http://www.developer.ibm.com/products/y2kform_mail.html`

Submission in that media is more efficient and will facilitate its processing.

Alternatively, you can return this hardcopy form to:

Year2000 Solution Developer Liaison
IBM Corporation
3200 Windy Hill Rd., WG2A20
Atlanta, GA 30339
United States of America

FAX: (770) 835-8001

⋆=required fields

First name:⋆

Surname:⋆

Title:⋆

Company name:⋆

Address:

City:

State:

Zip/Postal codes:

Country:

Phone number:

Fax number:

E-mail address:

Would you like us to link to your home page from our database? □Yes □No

WWW home page URL:

http://

==========

For purposes of this survey, a product or deliverable resulting from services is Year 2000 Ready if, when used in accordance with its associated documentation, is capable of correctly processing, providing and/or receiving date data within or between the twentieth and twenty-first centuries, provided that all products (for example, hardware, software, and firmware) used with the product or deliverable properly exchange accurate date data with it.

Firm's core business:

□Agent □Distributor □IBM Subcontractor □ISV □Reseller
□Service Provider □Solution Provider □System Integrator □other

If “Other,” please describe

Are you an IBM business partner?⋆ □Yes □No

If you answered Yes to the question above and you are an authorized IBM Solution Provider or IBM System Integrator, please specify any products or deliverables you provide as part of an approved Value Added Enhancement (VAE) or solution which are **NOT** Year 2000 Ready (if none, check None). This information will not be included in any publication but is part of your Year 2000 readiness affirmation.

Product/ Deliverable	**Name/description of VAE or IBM approved solution**	**Current plans to be Year 2000 Ready (include date)**

□None

Industry expertise.

□Construction □Distribution □Education □Finance □Government □Health □Insurance □Manufacturing/Process □Media □Retail □Transportation □Other

If “Other,” please describe:

Geographies supported:

□Africa □Asia/Pacific □Canada □Central America □Europe □Latin America □Middle East □United States

If your support is regional within a geography, please specify:

If you will support Europe, please answer the following questions.

If you provide application software, are you offering or do you plan to offer Single European Currency (EMU) ready versions? Please describe:

If you are providing consulting or services, do you have specific offerings to assist with EMU preparations? Please describe the types of services you perform:

==========

Please provide the following information about your Year 2000 Ready products:

Your product's name

Your product's version/release:

IBM platform your product supports:

Hardware platform: □AS/400 □PC Server □RS/6000 □S36 □S/38 □S/390 □Other

If “Other,” please describe:

Software platform: □AIX □MVS □NT □OS/2 □OS/400 □ VM □VSE □Other

If “Other,” please describe:

Provide a brief description of your product or offering (Tools, Application, Services, etc.):

Year 2000 certified:

□ITAA (Information Technology Association of America) □Other

If “Other,” please indicate certifying organization:

(**Note: Please copy this section as necessary to enter additional products.**)

If you are a tool vendor or provide support for Year 2000 tools, please answer the following questions:

Tool category:

□Inventory □Assessment □Find □Fix □Test □Other

If “Other,” please describe:

Year 2000 tools supported:

If “Other,” please describe:

If you provide Year 2000 project or consulting services, please specify type and provide a brief description:

□Assessment □Code Conversion □Consulting services □Impact analysis
□Inventory □Project management □System upgrade □Testing □Other

If “Other,” please describe:

Description of project or testing services:

Will you provide references upon request for prior application development and/or migration projects your company has completed? □Yes □No

==========

If you have questions regarding this survey, you may contact the Solution Developer Hotline, extension Y2K, in Atlanta at:

- 1-800-627-8563 (US and Canada)
- 1-770-835-9902 (Worldwide)

Communicating Your Comments to IBM

1998 February

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

Publication No. GC28-1251-08

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

1998 February

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

Publication No. GC28-1251-08

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
GC28-1251-08

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 the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

GC28-1251-08