Preparing SAS Software Applications for the Year 2000

Sunil Kumar Gupta, Millennium Technologies Institute, Inc.
Kirk Paul Lafler, Millennium Technologies Institute, Inc.
Charles Edwin Shipp, Millennium Technologies Institute, Inc.

Introduction

Solving the Year 2000 century date problem will be the largest computing task ever performed in history . . . and the costliest. Recent estimates for fixing the problem range between $300 billion and $600 billion and are rapidly climbing. Due to the enormous amount of software code that needs to be converted and the limited amount of time before the next century, organizations around the world are rushing to become Year 2000 compliant.

Programmers code programs differently, so there is no automated solution or "silver bullet" to fix the problem. The Year 2000 is not a technological problem, but a business problem with strategic implications that have direct impact on an organization's profitability. Solving it will require a common sense approach that keeps applications working while controlling costs. This paper and presentation will describe the problem, issues and challenges, the SAS environment, what a SAS date is, and SAS System solutions to the Year 2000 problem.

Origin of the Problem

The Year 2000 problem can be traced to the way many computers and programs handle dates by using dates with two-digit year notation, for example 07/07/97. The two-digit year notation has come about from customary date usage, to save storage space, and to save the number of keystrokes incurred by data entry personnel. The two-digit year affects the way data is manipulated, particularly when performing arithmetic computations, date comparisons, and sorting.

For example, the year "1980" is generally stored as "80" in data files, and "2000" is stored as "00". If you were to compute the difference between 97 and 80 (1997 and 1980 respectively), the result of this operation is accurately computed as 17.

The problem immediately becomes more serious when computing two-digit years that span more than one century. Say you were to compute the difference between 1980 and 2000. The computed result would produce an incorrect value of -80, the difference between 80 and 00. Errors of this type could affect any calculation that derives or uses time spans, such as when performing simple interest calculations. The problem doesn't end there. Many online applications provide for two-digit years such as in data entry and reporting systems.

Issues and Challenges

The time to begin the Year 2000 process is now. As the saying goes, ***don't delay . . . start today*** couldn't be more true.

Most analysts agree that solving the year 2000 century date problem will be a daunting task. Some estimates suggest that more than 90% of an organization's software inventories will be affected by the date bug and about 50% of the technology budget (through the year 2000) dedicated to fixing the problem.

There are several issues and challenges facing organizations:

- limitation of time
- immovable deadline
- numerous applications
- limited people, computing, and budget resources
- software configuration management
- missing source code
- availability of system documentation
- identification of date-sensitive programs
- legal consequences for noncompliance
- $$$ costs for Year 2000 effort.

Frequently Asked Questions

The following Frequently Asked Questions (FAQs) address a few of the more interesting questions and attempt to shed some light for the reader. The reader is advised to look elsewhere (i.e., Internet) for a more detailed and comprehensive list of FAQs.

Q1. What will happen if the Year 2000 issue isn't corrected?

A1. Any computer calculation or comparison involving a date could yield an incorrect answer.

Q2. Is the problem with Year 2000 a hardware or software problem?

A2. It is primarily a software problem.

Q3. Is there anything that will need to be done to my hardware?

A3. Computer users should read their users manuals and/or contact their service representative or sales vendor to determine whether their computer hardware can handle the change of century.

Q4. Does this problem only occur on mainframe systems and/or legacy applications?

A4. No. Any system or programs can be affected if two-digit year notation is used.

Q5. Is 2000 a leap year?

A5. Yes. To help you determine whether a given year is a leap year, apply these simple rules.

① When the year is evenly divisible by 4 it is a leap year, except for years ending in 00.

② A year ending in 00 is a leap year if it is evenly divisible by 400.

Q6. How can one proceed in solving the various Year 2000 problems?

A6. Apply the following three step rule for every Year 2000 problem:

① Prepare by taking inventory of all application programs.

② Implement by identifying, correcting code and data, and verifying.

③ Deploy by performing system tests and moving testing systems into production.

Q7. Will legacy systems be able to handle dates in the twenty-first century without requiring changes?

A7. Not necessarily. Further assessment would need to be performed to determine if the legacy system contains any dates and/or date-dependent processing. Programs using of a four-digit century with year, may have little or no difficulty. If dates are represented as two-digits, then problems could arise. An inventory and impact assessment is advised to be certain.

The SAS Environment

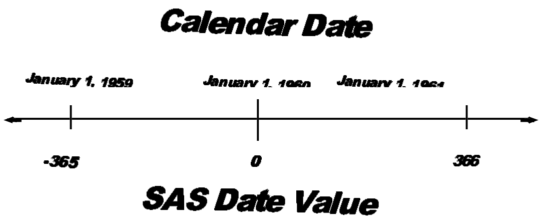
SAS Institute, as we might know, has been prepared to handle Year 2000 compliancy for many years now. They planned for the century date change by allowing:

- date variables to store values from 1582 to 10,000
- date values to be the same across operating platforms
- system option to control how dates are handled.

What is a SAS Date?

A SAS date variable is defined as numeric. It is an integer value representing the number of days between January 1, 1960 and a specified date. No matter how a date is displayed or written, the SAS System converts and stores a date as the difference between January 1, 1960 and the date being entered. Date values can be positive (values after January 1, 1960) or negative (values before January 1, 1960).

SAS date values represent dates from the year 1582 A.D. (after the adoption of the Gregorian calendar) to 10,000 A.D. Leap years, century, and fourth-century adjustments are automatically handled by the SAS System.



SAS dates can be used with formats (or output templates) to convert a date value to another form. Formats can be applied to these values by using either a FORMAT statement in a DATA or PROC step, or with a PUT statement in a DATA step.

Some of the most frequently used date formats include: DATEw., DDMMYYw., MMDDYYw., MONYYw., WEEKDATEw., WORKDATEw., and YYMMDDw.

Examples of date formats include the following:

Value	MMDDYY8.	MMDDYY10.
0	01/01/60	01/01/1960
-365	01/01/59	01/01/1959
366	01/01/61	01/01/1961

SAS dates can be used with informats (or input templates) to read a data value into a variable. Informats can be applied to these variables by using either a INFORMAT statement in a DATA or PROC step, or an INPUT statement.

Some of the most frequently used date informats include: DATEw., DDMMYYw., JULIANw., MMDDYYw., MONYYw., SMFSTAMPw., and YYMMDDw.

Examples of date informats include the following:

Value	MMDDYY8.	MMDDYY10.
0	01/01/60	01/01/1960
-365	01/01/59	01/01/1959
366	01/01/61	01/01/1961

SAS dates can be used with functions for the purpose of converting non-SAS dates to SAS dates when an informat will not work, obtaining the computer system date, converting SAS date values into printable values, or counting and incrementing time intervals.

Some of the most frequently used date functions include: DATE, DATEJUL, DAY, INTCK, JULDATE, MDY, MONTH, QTR, TODAY, YEAR, and YYQ.

Examples of date functions include the following:

SAS Statement	Result Value
DATE()	366
MDY(1,1,2000)	01/01/2000
YEAR(DOB)	1961

SAS System Solutions to the Year 2000

The Year 2000 becomes problematic for systems that perform processing on two-digit years such as in comparisons, arithmetic operations, and sorting. The SAS System has developed four solutions for addressing Year 2000 compliancy.

The first solution provides a SETINIT to SAS installations so they can perform testing activities by setting the system date into the Year 2000 (bypassing the license expiration message).

By working with SAS Institute, a proper Year 2000 test setinit can be initiated. This allows the SAS System date to be set within the range of December 15, 1999 to March 14, 2000 without causing an error in the license agreement.

A second solution involves using the YEARCUTOFF= system option for dates containing two-digit years. Its value represents a 100-year span and recognizes values from 1582 to 10,000.

The YEARCUTOFF OPTION was implemented in Release 6.06. In addition, this option automatically adjusts for leap years, century adjustment, and fourth-century adjustment. Once the option is defined, it specifies the first year of the 100-year span.

Examples of the interpretation of two-digit years with the YEARCUTOFF OPTION includes:

Value	Date Range
= 1900 (Default)	1900 - 1999
= 1950	1950 - 2049
= 1972	1972 - 2071

Specific examples of the default value YEARCUTOFF = 1900 includes:

MMDDYY8.	MMDDYY10.
10/31/98	10/31/1998
12/31/99	12/31/1999
01/01/00	01/01/1900

Specific examples of the value YEARCUTOFF = 1940 includes:

MMDDYY8.	MMDDYY10.
10/31/98	10/31/1998
12/31/99	12/31/1999
01/01/00	01/01/2000

Another solution allows the SAS System to interact with 3rd party vendor tools by simulating changing the system clock to the 21st century. This new option is called SVC11/NOSVC11, and permits products that trap and manipulate date and time information to benefit from this new option. Currently, this feature is available in release 6.09E and operates under MVS.

Finally, the SAS System offers a new feature for detecting two-digit years in date-oriented applications and program. This experimental option is called YEAR2000CHECK= and is specified as a parameter of the DEBUG= system options as follows:

```
OPTIONS  DEBUG='YEAR2000CHECK=n'  ;
```

where 'n' indicates the maximum number of occurrences of two-digit years you want reported on the SAS log.

The YEAR2000CHECK feature checks for two-digit years in dates, date functions such as DATEJUL, MDY, YYQ, and date informats such as DATE, DATETIME, DDMMYY, JULIAN, MMDDYY, MONYY, and YYMMDD.

Conversion Strategies

Legacy systems that store, display or print dates in two-character representation for year ('97, '98, '99, ...) will have an ambiguity problem when any of the year values go beyond the year 1999. Serious time-span calculations and other problems will then occur. Fourth generation language (4GL) date functions, such as the SAS function YEARCUTOFF provide a user-selected 100-year window and is good for many situations. Another solution is to read and write the storage using hex to retain the old record layout. Two hexadecimal bytes can represent year values from 0 to over 65,000, removing the 100-year limit. This method has the advantage of not needing to redesign old file structures, and the added advantage that other languages can also read the files. Several other options are presented.

Inventory of Files

There are several ways to solve the Year 2000 dilemma and what is best for one system will not be the best for other systems. Each system needs to be considered individually based on the complexity of 1. input/output screens; 2. database; 3. flat files (input/output); 4. program variables and logic; and 5. printed output reports. Assuming your system is valuable enough to continue (not be absorbed into another project, etc.) then you have several choices.

Four-Digit Year Expansion

Your first choice is to expand all dates to four-digit year values. This may be easier than you first think, especially if your dates are stored in SAS or Focus files. You could have a system with very little date values involved.

Fixed 100-Year Window

A good solution for systems that have dates that are rather close (within 25 years before and after) to the current date, then you can feel confident that a 100-year fixed window will include your dates. You can chose a 25/75, 50/50, or 75/25 year window depending on if you feel dates are more prevalent in the future, past, or about equally balanced. Then use the following SAS Options statement at the top of your program.

```
OPTIONS YEARCUTOFF=1950;
```

or

```
OPTIONS YEARCUTOFF=1925;
```

or

```
OPTIONS YEARCUTOFF=1975;
```

This allows you to avoid a potentially large programming effort and still have the services of your computer system. This method is very valuable when things are complex. A very simple 4GL statement in your program adjusts all of the standard date results. Note that non-standard handling of year values are not corrected. (SAS, and Focus and others are fourth generation languages, whereas Cobol, FORTRAN, and others are third generation languages.)

Sliding 100-Year Window

Suppose you want your window for dates to be four or five years into the future and 95 years in the past. Plus you want it to move each year with the current year. You can then program a "Sliding 100-Year Window" using the following statements:

```
*** ENTER WITH YR IN TWO-DIGITS ('97, '98, '99,
'00, '01),
*** EXIT WITH YEAR IN FOUR-DIGITS (1997, 1998,
1999, 2000, 2001);
        YRTEST1=YEAR(TODAY())+9-2000;
        IF YR <= YRTEST1  THEN YEAR = 2000 +
YR;
           ELSE YEAR = 1900 + YR;
```

This gives you a sliding window that covers 90 years in the past and 9 years in the future.

Hexadecimal Format

Larger problems arise when multiple systems rely on the same flat files, and they have two-digit year values. Making one simple change from two-digit to four-digit year values can have a domino effect, and in many instances, that approach must be avoided -- it is not an option. In these instances, retaining the relative position and length of variable fields is possible by changing the input/output year formats from numeric (2.) or (3.) to hexadecimal such as PK2. (packed.) One then needs to consider a planned migration in program version and data version, but it is certainly a viable solution to a difficult problem.

EXAMPLE CODE (BEFORE)

```
  INFILE DD1;
  INPUT   ID $3.    PART $CHAR12.   YR1 2.   YR2 2.;
```

EXAMPLE CODE (AFTER)

```
  INFILE DD1;
  INPUT   ID $3.    PART $CHAR12.   YR1 PK2.   YR2
PK2.;
```

It is a good idea to make two copies of all programs and data involved: one for backup and one for testing. To move entirely in this direction (using PK2. fields) you can have a utility program write over the TEST data sets so they have four-digit dates.

Do Nothing

There are cases where you will want to do nothing. If your system has two-digit dates but they are only for display and printing, with no calculations and no possibility of being ambiguous, then an option is to just let them appear as is. For example MM/DD/YY for Independence Day for the next few years would be: 07/04/97, 07/04/98, 07/04/99, 07/04/00, 07/04/01, 07/04/02, and 07/04/03. If the values are for display only, then nothing needs to be done. Be careful, however, that there is no sorting, comparisons, nor time-span calculations. In these cases choose another solution.

Select a Year 2000 Solution
(Written in "pseudo-code")

```
(1) Do ... Examine your System/Program/Data:

(2) If the representation of years is OK, and there is
     no:  1. time-span calculations,
   nor:  2. year-sorting,
   nor:  3. year-comparisons,

then you could consider leaving it ...  "As Is."
```

(3) If (Priority > Difficulty) then **EXPAND ALL DATE FIELDS**;
 * From two-digit to four-digit year fields;
 * in all pgms/screens/data/printed-reports;

(4) **Else If**
1. you use standard 4GL date functions, and
2. your dates are close together (< 100 yrs)

then consider the **FIXED WINDOW** in that language.

For SAS programs it is:
OPTION YEARCUTOFF= yyyy; (like 1950);

(5) **Else if** you prefer a **SLIDING WINDOW**: use system calls based on the system current year:

For SAS programs it can be:
%EVAL(%SYSFUNC(YEAR(%SYSFUNC(TODAY()))-90);

(6) With choices (4) and (5), remember also to still **check** output reports and input/output screens to be to your liking (and your customers) i.e. four digit years.

(7) **Else If** (Other languages use files) or (years spanned are too diverse) then use an **ENCAPSULATION METHOD**.
* This can pack four-digit values into two columns;
* This can temporarily time shift your data processing.

(8) Last Resort (sometimes best) is to **RE-ENGINEER**: to place program features into other systems -- possibly **Year 2000 compliant** "off the shelf" or turnkey software.

Table 1. Decision Tree

Best Alternative

Because there are so many data files and programs to fix, each company needs to prioritize their efforts. In addition, it is best to select the solution best suited for each situation. It will not be an option to expand all systems to four-digit years, nor is it the best solution for all systems. Past experience indicates that the YEARCUTOFF window (where the value sets the beginning edge of the 100-year window) is an excellent solution choice due to the fact that it will save 70% of the reprogramming costs and have the same end results. Further, a sliding window would provide a perpetual solution.

A Final Note - Conclusion

Working with SAS dates, as implemented in the SAS System, doesn't represent nearly the same challenge as working with dates used in other languages such as COBOL. Still, preparing SAS applications for the Year 2000 by using one or more of the available solutions as presented in this paper can enable you to smoothly transition your applications into the next century.

The most important thing to remember as you embark on this project is to just get started, and avoid getting bogged-down.

Contact Information

The authors welcome comments and suggestions. They can be reached at:

Millennium Technologies Institute, Inc.
P.O. Box 2027
Spring Valley, CA 91979-2027
Tel: (619) 660-2400
Email: SunilGupta@aol.com