Journal of International Information Management

Volume 8 | Issue 1 Article 9

1999

The year 2000 problem

Marshal Whatley
California State University, San Bernardino

The year 2000 problem

Marshal Whatley
California State University, San Bernardino

ABSTRACT

The purpose of this article is to discuss the root causes of the Year 200 Problem, the effects of the problem and how a systematic approach can be used to solve this problem.

INTRODUCTION

The Year 2000 Problem (Y2K), also known as the 'millennium bug,' may well be one of the largest problems facing organizations as they enter the 21st century. Computers are an integral component of an organization, organically of equal importance as financial resources and human resources. An organization's information system is the base on which all strategic and informational processes are developed, defined, and redefined. Therefore, the Y2K problem is not just a computer problem, it is a systemic business problem. A systemic approach will be needed to solve this problem.

This is a business problem because information system (IS) managers must effectively plan for the reallocation of hundreds of IS workers and millions of dollars in each organization. The average cost to examine a line of COBOL code is $8.35, and the Gartner Group estimates that there are 180 billion lines of COBOL world wide (Freeman, 1998). This problem requires a commitment of business resources (humans, financial, and capital) and if left uncorrected puts the organization and its customers at risk.

DEFINITION OF THE Y2K PROBLEM

The Y2K problem is the failure of computers to correctly store, manipulate, and calculate dates from the 1900's to the year 2000, including leap years. The Y2K problem has existed since the beginning use of computers. Computers, like most technological products, are relatively expensive in the beginning of their product life cycle. They become less expensive as innovation and the evolution of technology produces superior products. Hard disk space in the beginning use of computers was very expensive and very precious, almost as precious as silver or gold. The cost

of a hard disk to store 1 million bytes (1 MB) in the 1960s was $761.000. Today it costs as little as $ .75.

Economics was the main reason why programmers and system analysts were scrambling to find ways to save disk space wherever possible. One method that was used was to eliminate the first 2 digits of the year out of the date. For example to represent the date August 4, 1960 it would appear as 080460 instead of 08041960. This is the MMDDYY format, and of course this method assumes that all dates fall within a particular century, in this case the 1900s.

The second reason for the problem is that nobody expected the sofware programs, mostly in COBOL that were written in the 1970s and early 1980s would still be around by the year 2000. And to compound the problem, not only are the original programs in place, but they have been revised, updated, and altered with little or no documentation in place, so today's IS managers have to look line by line, component by component to find where the Y2K bug exists.

To make matters worse most programmers used the two-digit year date as a standard for almost everything, e.g., BIOSs were still using the last two digits until 1997. All of these elements contribute to machines malfunctioning and being unable to accurately calculate the date above December 31, 1999 and will roll over to January 1, 1900.

RAMIFICATIONS

The ramifications of not fixing the Y2K problem can range from simple adverse service disruption to economic instability or to possible cause of human life. Most Fortune 500 companies have identified, and are on their way to solving the Y2K problem. The Federal Government has identified the affected systems and are working on the programming fixes needed. However, small and medium size businesses are behind, and many believe they will not have a solution by the year 2000. NYSE has completed its test, but what about the markets in Asia, Europe, and Mexico, the Third World nations are way behind, and some are not even looking at the Y2K problem!

We are in a global economy, and our economy might well be in danger if gridlock spreads across the globe because our computers cannot talk to the rest of the world's computers. What would happen to the Eurodollar if Finland's computers are Y2K compliant, but Germany's are not? Will they have electricity? Will their transportation system work? What about government services and administration, and military defense computers? Will they work?

It would be impossible to measure the ramifications of Y2K problems, simply because there would be heterogeneous makeup of computers that are Y2K compliant, and some that are not Y2K compliant. This would cause what is known as the "domino effect."

APPROACHING THE PROBLEM

IS managers have to be careful in their approach in trying to correct the Y2K problem. It can be a minefield full of surprises especially in a medium-sized to large-sized organization where there may be several local IS departments or decentralized command and control of IS in the organization. Below are the steps to approach and fix the problem. These steps should minimize risk and maximize efficiency as they are arranged in a systematic work flow process.

Evaluate the situation

Evaluate the possible solutions

Implement changes

Testing

Integrate corrected applications

Step One: Evaluate the Situation

The first step is to evaluate the situation, beginning with a comprehensive inventory of all software and hardware products including homegrown applications and embedded systems. A complete inventory is needed on PC hardware and software (operating systems and desktop applications), network software (NOSs such as NetWare, NT, UNIX, etc.) and hardware (switches, hubs, and routers), databases (SQL, Oracle, Informix, etc.). In this inventory it is essential to record platform versions, releases, updates, and patches. One of the positive aspects from the Y2K problem is if the organization does not have an asset management program in place, most of the work will be done when this step is finished.

Step Two: Evaluate Possible Solutions

Once the inventory is complete, a query of the individual vendors or OEMs is needed to find out what fixes, if any, are available; what the costs are; what the timeframe for implementation is; if correcting different systems within the same timeframe can be coordinated; and what changes or enhancements the corrections will bring to the organization. Also, IS managers will have to decide whether to use existing staff to work on the Y2K problem or to use outsourcing, or maybe a combination of the two.

Once the IS manager knows what he is facing he can then start to develop a strategy to attack the Y2K problem. There are basically two strategies to attack the Y2K problem, the rework strategy and the replace strategy. Each has its own pros and cons and depends heavily on the situation.

The rework strategy would be modifying existing applications, installing patches and rewriting programs to accept the rollover to the new century. This approach involves altering data fields in programs from 2 digits to four digits for the year, or rewriting applications to use the

appropriate century in calculations based on the data characteristics. There is also a technique known as windowing, in which if the year in the database is less than 30, the century is automatically changed to read 20 instead of 19. If it is above 30, then the century stays at 19. This strategy also involves upgrading BIOSs, firmware, service packs, and service releases to bring hardware and software to Y2K compliance.

However, the risk of this option is that one may not catch all of the problems, even after the test phase, because testing may not catch every possible circumstance. Especially when the organization's computer talks to another organization's computer with its computers, it would be almost impossible to find out what changes are needed until after the year 2000. It would take a lot of cooperation, scheduling, and coordinating from both IS departments to develop and test such a mock communication.

The replacement strategy offers an opportunity to enhance the functions of the legacy systems via replacement. This option is either to purchase or build a new system to fit the business needs of the organization or to include upgrading to versions that are compliant. This strategy is usually the most expensive, but provides two key benefits, one is the implementation is usually faster, and the second is the organization is benefitting with a new enhanced system.

Step Three: Implement Changes

Regardless of which strategy was used it is important that changes made to the applications are done before the middle of 1999 to allow for sufficient time for testing of the corrected applications.

In my opinion, the PCs and the embedded systems are going to be the hardest components to identify and correct the Y2K problem. On the PC side, most large organizations have decided to completely decentralize, letting the control and support come from the local areas, and brand and models were based upon personal preferences with no global standard. The embedded systems will be a far more daunting task of identifying what components, and what the effect of the components are if they are not Y2K compliant.

On the BIOS of PCs there are three options to know to fix the Y2K problem. First there are commercially available TSR (Terminate-Stay-Resident) programs that can poll the RTC at a certain interval and check the date in the RTC (Real Time Clock) and correct it if needed. The system clock is updated by the RTC, and most operating systems get the date from the system clock. The drawback to this approach is that TSRs can be used under MS-DOS and Windows 95 and such, but Windows NT, OS/2, Linux, and UNIX, and most proprietary systems do not allow TSRs to operate. There may also be a drag on performance with using the TSRs because they are always resident in memory.

The second option is to flash upgrade the BIOS on the machines that are found not to be capable of handling the Y2K problem. The newer BIOSs available today are Y2K compliant.

The third option is to completely replace the PC with a newer system, most 486 and below systems do not have the ability to be updated with a BIOS upgrade. This is the most expensive option for the PCs, but it is the most reliable.

Most networking equipment does not use a date in performing their tasks, but there are exceptions. Certain routers may use a time stamp for processing packets. SNMP consoles use the date heavily in organizing strings and polling data from network devices into databases so that historical records and trends can be analyzed.

Embedded systems will be the hardest to identify as having problems with processing the date correctly. Most embedded chips do not use the date at all, but they are used everywhere. Embedded systems can range from telephone switches, fax machines, energy management systems, door and vault locks, elevators, heart fibrillators, pacemaker monitors, ATM machines, and credit card processing machines. Those card locks on hotel rooms would be the prime example, could you get into your room on January 1, 2000 if you checked in on December 31, 1999.

Visa had to recall several thousand credit cards because retail point-of-sale systems could not read credit cards with "00" expiration dates. As consumer complaints increased, Visa also banned member banks from issuing credit cards with expiration dates beyond 1999. Visa lifted the ban last October, once most of the buggy card readers had been fixed (Hoffman & King, 1998).

DIOs will also have the dilemma of deciding which approach to take on the main information system for their organization, whether it is a MRP, EPR, financial or human resource system. The rework strategy is fine, as long as ample planning in resources is taken into account. Project management is a must. Imagine several outsource COBOL programmers searching line by line, millions of lines of code, and only being able to do so much in a day because the normal day to day operations are still in progress. There is only so much time available.

The replacement strategy would be faster, and maybe even more cost effective, if 2 million dollars were spent fixing the problem with no enhancements. The other option is to spend a little more and buy a brand new system. Buying the new system would probably be the better choice. The drawback is that there would be new end-user training and maybe a learning curve for the existing IS staff.

Step Four: Testing

We will need to unit test and system-test the corrected applications to ensure that they are indeed Y2K compliant. We also need to test the interoperability between applications and make sure they function as required. It will also be necessary to create a special time-based testing environment with time-sensitive scenarios to prevent the possibility of the loss of data or systems resources. A key point to remember is that the year 2000 is a leap year, and that testing should be inclusive of testing all leap years to be sure the date calculation is accurate.

Testing the Y2K fixes has to be very detailed. For example, Smith Kline Beecham, a pharmaceutical firm, had acquired to apparently identical weighing machines. One cleanly passed a test of millennium compliance. The other machine failed. It appears the manufacturer had bought the chips for these machines from two different vendors (*The Economist,* 1998).

Step Five: Integrate Corrected Applications

Finally, we will need to integrate each corrected or new application into the production environment, making conversions if needed and planning the type of cut-over method to use. Most vendors suggest having at least two good backups before the year 2000; if something does not work right at least there is a good copy of the data. Where functionality improvements have been made to the applications, it is important that adequate end user training is planned for and implemented.

CONCLUSION

If Y2K does in fact turn out to be a disaster, then the lesson will be clear. We must have IS managers who understand information technology, and those organizations have to realize the importance of understanding what their information systems are doing. Documentation is key in any IS endeavor.

REFERENCES

Freeman, L. (1997, October). Recover missing source code to overcome "leak-roof syndrome." *Enterprise Systems Journal,* 42-48.

Hoffman, T. & King, J. (August, 1998). Early year 2000 glitches provide sneak preview. CNN Web Site: http://cnn.com/TECH/computing?9808/28/y2k.preview/index.html

The Economist (1998, September). Small cause. Vol. 348, Issue 8086, 4.