The History and the Hype

Computer scientists may disarm the Y2K bomb in time, but that doesn't mean they didn't screw up along the way

By CHRIS TAYLOR

TWO DIGITS. THAT'S ALL. JUST TWO LOUSY digits. 1957, they should have written, not 57. 1970 rather than 70. Most important, 01-01-2000 would have been infinitely preferable to 01-01-00. Though most of the dire predictions connected with that date—the Year 2000 computer bug's moment of truth—are unlikely to come true, a little computer-generated chaos would provide a fitting conclusion to a 40-year story of human frailties: greed, shortsightedness and a tendency to rush into new technologies before thinking them through.

How did this happen? Who is responsible for the bug we call Y2K? Conventional wisdom goes something like this: back in the 1950s, when computers were the size of office cubicles and the most advanced data-storage system came on strips of punched cardboard, several scientists, including a Navy officer named Grace Murray Hopper, begat a standard programming language called COBOL (common business-oriented language). To save precious space on the 80-column punch cards, COBOL programmers used just six digits to render the day's date: two for the day, two for the month, two for the year. It was the middle of the century, and nobody cared much about what would happen at the next click of the cosmic odometer. But today the world runs on computers, and older machines run on jury-rigged versions of COBOL that may well crash or go senile when they hit a double-zero date. So the finger of blame for the approaching crisis should point at Hopper and her COBOL cohorts, right?

Wrong. Nothing, especially in the world of computing, is ever that simple. "It was the fault of everybody, just everybody," says Robert Bemer, the onetime IBM whiz kid who wrote much of COBOL. "If Grace Hopper and I were at fault, it was for making the language so easy that anybody could get in on the act." And anybody did, including a group of Mormons in the late '50s who wanted to enlist the newfangled machines in their massive genealogy project—clearly the kind of work that calls for thinking outside the 20th-century box. Bemer obliged by inventing the picture clause, which allowed for a four-digit year. From this point on, more than 40 years ahead of schedule, the technology was available for every computer in the world to become Y2K compliant.

Programmers ignored Bemer's fix. And so did his bosses at IBM, who unwittingly shipped the Y2K bug in their System/360 computers, an industry standard every bit as powerful in the '60s as Windows is today. By the end of the decade, Big Blue had effectively set the two-digit date in stone. Every machine, every manual, every maintenance guy would tell you the year was 69, not 1969. "The general consensus was that this was the way you programmed," says an IBM spokesman. "We recognize the potential for lawsuits on this issue."

No one in the computer industry wanted to rock the boat. And no one could alter the course IBM had set, not even the International Standards Orga-

TIME/CNN POLL

How concerned are you about the Y2K bug problem?	
Somewhat/very concerned	59%
Not very/not at all concerned	39%

Might you take any of these actions to protect yourself from possible problems associated with the Y2K bug?	
Take extra cash out of bank account	47%
Stockpile water and food	33%
Not fly on an airplane	26%
Keep family members at home	26%
Stockpile fuel for car or house	23%
Arm yourself with a shotgun	13%
Move to a rural area	12%

Are the following likely to happen when the year 2000 begins?	
Equipment with computers will fail	59%
Banking system will be disrupted	53%
Riots or other social unrest will occur	38%
The world as we know it will end	9%

Where the Y2K Bug Came From

1890 Herman Hollerith develops an electrically driven census system that reads punch cards. Six years later, he founds the company that eventually becomes IBM.

1957 Grace Murray Hopper creates Flow-matic, the first computer language written in plain English. Two years later, it forms the basis of COBOL—a compromise *lingua franca* for business computers. To save precious room on the cards, years are abbreviated to two digits (e.g., 1957 is represented as 57).

1960 Anticipating problems later on, Robert Bemer and 47 other computer scientists begin lobbying for the four-digit year.

1964 IBM introduces the spectacularly successful System/360 mainframe, which retains the two-digit year.

1967 The White House orders the National Bureau of Standards to settle the date debate. Pressured by the Pentagon, the bureau keeps the two-digit year.

1979 Bemer makes the first widely published prediction of the Y2K crisis, in the journal *Interface Age.* The reaction is underwhelming.

1993 Curious nuke watchers at NORAD turn their computer clocks forward to Jan. 1, 2000—and the ICBM alert system crashes.

1995 IBM finally acknowledges the Y2K bug—and announces plans to help its customers make "timely year 2000 transitions."

1998 Y2K merchandise (mugs, clocks and caps) goes on sale. The White House appoints John Koskinen as its Y2K czar. Most government agencies get a failing grade in Y2K readiness.

1999 MARCH 31 Koskinen's preliminary deadline for all computer-system fixes.

2000 JANUARY 1 The moment of truth.

nization, which adopted the four-digit date standard in the 1970s. The Pentagon promised to adopt century-friendly dates around 1974, then sat on its hands. Bemer himself wrote the earliest published Y2K warnings—first in 1971, then again in 1979. Greeted by nothing but derision, he retired in 1982. "How do you think I feel about this thing?" says Bemer, now an officer at his own Y2K software firm. "I made it possible to do four digits, and they screwed it up."

Meanwhile, the torch of Y2K awareness passed to a new generation. In the fall of 1977, a young Canadian named Peter de Jager signed on as a computer operator at IBM. His first task was to boot up a nationwide banking system run on an IBM 370. When the machine whirred into life, it asked for the date. As De Jager, a mathematics major straight out of college, entered the number 77, a thought occurred to him. Did this machine care what century it was? With the impetuousness of youth, he marched off to his manager and informed him the computer would not work in the year 2000. The manager laughed.

"We used to joke about this at conferences," De Jager says. "Irresponsible talk, like 'We won't be around then.'" But by 1991, De Jager, a self-described "nobody" in the industry, had decided he would be around. Four years later, he was giving more than 85 lectures a year on the topic and posting regular updates to his site, the Web's first for Y2K warnings, *www.year2000.com.*

And here's the curious thing. From 1995 on, Y2K awareness had a kind of critical mass. Congress, the White House and the media all got wind of the bug at about the same time. After making too little of the problem for so long, everybody began to make, if anything, too much of it.

By 1999 folly has compounded folly. In many cases, the original COBOL code has been rejiggered so many times that the date locations have been lost. And even when programmers find their quarry, they aren't sure which fixes will work. The amount of code that needs to be checked has grown to a staggering 1.2 trillion lines. Estimates for the cost of the fix in the U.S. alone range from $50 billion to $600 billion. As for Y2K compliance in Asian economies still struggling with recession? Forget about it.

The fact is that no one on the planet really knows what will happen when 01-01-00 rolls around. Whether we'll be glad we were panicked into action or we'll disown the doomsayers depends on how diligently the programmers do their job in the coming weeks. One thing is already clear. In a century in which man split the atom, spliced genes and turned silicon into data, the tale of Y2K—how we ignored it for 40 years, then flew into a tizzy—will not be remembered as our finest hour. ■

Questions

1. Who is responsible for creating the Y2K bug? Why is there concern about this problem?

2. What steps are programmers taking to address the Y2K phenomenon?