

THE MILLENNIUMBUG

We Know It Can Hurt Us. We Know It Didn't Have To Be. What We Didn't Know, Until Today, Was Whom We Can Blame for Letting It Loose

July 18, 1999 More than **27 years ago**

By Gene Weingarten

We are knocking at the door of a high-rise apartment in Baileys Crossroads, with a question so awful we are afraid to ask it. We do not wish to cause a heart attack.

A woman invites us in and summons her husband, who shuffles in from another room. She is 78. He is 82. They met in the 1960s as middle-management civil servants, specialists in an aspect of data processing so technical, so nebbishy, that many computer professionals disdain it. He was her boss. Management interface blossomed into romance. Their marriage spans three decades. They are still in love.

"You know how we use Social Security numbers alone to identify everyone?" she says. She points proudly to her husband. "That all started with this kid!"

The kid has ice cube spectacles and neck wattles. He has been retired for years. Some of his former colleagues guessed he was deceased. His phone is unlisted. We located him through a mumbled tip from a man in a nursing home, followed up with an elaborate national computer search. Computers--they're magic.

It is still early. We have, alas, roused them from bed.

She is feisty. He is pleasantly grumpy. They are nice people.

Here is what we have to ask him: Are you the man who is responsible for the greatest technological disaster in the history of mankind? Did you cause a trillion-dollar mistake that some believe will end life as we know it six months from now, throwing the global economy into a tailspin, disrupting essential services, shutting down factories, darkening vast areas of rural America, closing banks, inciting civic unrest, rotting the meat in a million freezers, pulling the plug on life-sustaining medical equipment, blinding missile defense systems, leaving ships adrift on the high seas, snarling air traffic, causing passenger planes to plummet from the skies?

Stay informed with a subscription to The Washington Post.

Subscribe



He is wearing pajamas.

A Hot Date

By now, everyone knows that on Jan. 1, 2000, something dreadful will happen on a global scale. Or possibly it will not. Experts are divided. This much is indisputable: To prevent it, billions of dollars have already been expended not only by government, which is prone to squandering money on foolishness, but also by big business, which is not. This is no empty scare.

Technology has been the propulsive force behind civilization, but from time to time technology has loudly misfired. In the name of progress, there have been profound blunders: Filling zeppelins with hydrogen. Treating morning sickness with Thalidomide. Constructing aqueducts with lead pipes, poisoning half the population of ancient Rome. Still, there is nothing that quite compares with the so-called "Millennium Bug." It is potentially planetary in scope. It is potentially catastrophic in consequence. And it is, at its heart, stunningly stupid. It is not like losing a kingdom for want of a nail; it is like losing a kingdom because some idiot made the nails out of marshmallows.

On Jan. 1, 2000, huge numbers of computers worldwide are expected to fail because, despite the foreseeable folly of it, they have always been programmed to think of the year in two digits only.

The two-digit year is a convention as ancient as the feather pen--writing the date on a personal letter with an apostrophe in the year, implying a prefix of 17- or 18- or 19-. But reading an apostrophe requires sentience and judgment. Computers possess neither. They cannot distinguish an "00" meaning 1900 from an "00" meaning 2000. When asked, for example, to update a woman's age on Jan. 1, 2000, a computer might subtract her year of birth (say, '51) from the current year ('00), and conclude she will not be born for another 51 years. A human would instantly realize the nature of the error, adjust his parameters, and recalculate.

Computers aren't built that way. They require absolute, either-or, plus-or-minus, binary logic at every step of their operation, and if this process is stymied even momentarily, if there is a juncture at which neither plus nor minus yields a comprehensible response, a computer will react immaturely. Sometimes it will start acting out--doing petulant, antisocial things such as coughing out daffy data or obliterating files. More often, the computer will simply burst into tears. It will shut itself down.

The permutations of the Y2K problem are bewildering. If General Motors has fixed its computers, that's swell; but if the hydroelectric plant that sells power to the subcontractor who imports the rubber that is used to make tires for GM cars has not fixed its problem, the GM assembly line closes down anyway. Plus, the Y2K problem is hard-wired into millions of microprocessor chips, independent mini-brains that are embedded in things like automobiles, traffic control systems, medical equipment, factory control panels; some businesses aren't even certain where all their microprocessors are.

Never has a calamity been so predictable, and so inevitable, tied to a deadline that can be neither appealed nor postponed. Diplomacy is fruitless. Nuclear deterrence isn't a factor. This can't be

Stay informed with a subscription to The Washington Post.

Subscribe

Y2K has powerful, nearly mystical, themes. For some religious fundamentalists who have long been predicting a millennial apocalypse, the avenging instrument has finally loomed into view. For Luddites aghast at the excesses of the industrialized world, Y2K is the perfect comeuppance. For anyone who has ever read Vonnegut or Eliot, the ironies are lush.

This is the way the world ends. Not with a bang but a . . . crash.

Because society has been gamely focused on working together to forestall disaster, not much effort has so far been expended on senseless finger-pointing. The civility will end after the first of the year. Finger-pointing will no longer be senseless. One question will be asked repeatedly, mostly by attorneys gearing up for lawsuits:

Who screwed up?

The search for a culprit is an honored American tradition. It nourishes both law and journalism. When things go bad, we demand a fall guy. A scapegoat. A patsy.

Today we'll search for one, and find him.

The Unsquashable Bug

First, it isn't really a "bug."

The term "computer bug" was coined by Navy computer pioneer Grace Hopper in the 1950s after a moth got into one of her machines and it went haywire. A "bug" implies something unforeseeable.

The Y2K problem wasn't just foreseeable, it was foreseen.

Writing in February 1979 in an industry magazine called *Interface Age*, computer industry executive Robert Bemer warned that unless programmers stopped dropping the first two digits of the year, programs "may fail from ambiguity in the year 2000."

This is geekspeak for the Y2K problem.

Five years later, the husband-wife team of Jerome T. and Marilyn J. Murray wrote it much more plainly. In a book called "Computers in Crisis: How to Avoid the Coming Worldwide Computer Systems Collapse," they predicted Y2K with chilling specificity.

Few people read it. The year was 1984, and to many, the book seemed very 1984-ish: a paranoid Orwellian scenario. *ComputerWorld* magazine reviewed it thus:

"The book overdramatizes the date-digit problem. . . . Much of the book can be overlooked."

Stay informed with a subscription to The Washington Post.

Subscribe

Basically, we blinded ourselves, like Oedipus. It seemed like a good idea at the time.

Imagine you own a car that gets one mile to the gallon, and every additional ounce in the passenger compartment further reduces the gas efficiency. You would do anything you could to lighten your load. You might even drive naked, gawkers be damned.

That's pretty much what occurred back in the 1950s, in the early days of computers. Simple arithmetic calculations required a machine the dimensions of a minivan. Memory was contained not in chips the size of fingernails but in electrostatic vacuum tubes the size of cucumbers; small stores of memory cost tens of thousands of dollars. Data were entered by punching holes in stiff cards the size of airline tickets, each containing only 80 characters of information. Businesses needed warehouses to store tons of cards. Anything that reduced the amount of data, even slightly, saved money.

What followed was nearly inevitable. Programmers built a house of cards.

Most of them employed abbreviations, particularly to represent prosaic bits of recurring data, such as the date. They expressed the month, day and year in a total of six digits rather than eight.

Many programmers say today that they knew they were being sloppy. But there were greater priorities.

So they drove naked.

Why didn't people realize earlier the magnitude of the problem they were creating?

And when they did realize it, why was the problem so hard to solve?

Have Run, Will Travel

We sought the answer from the first man to ask the question.

Robert Bemer, the original Y2K whistleblower, lives in a spectacular home on a cliff overlooking a lake two hours west of a major American city. We are not being specific because Bemer has made this a condition of the interview. We can say the car ride to his town is unrelievedly horizontal. The retail stores most in evidence are fireworks stands and taxidermists.

In his driveway, Bemer's car carries the vanity tag "ASCII." He is the man who wrote the American Standard Code for Information Interchange, the language through which different computer systems talk to each other. He also popularized the use of the backslash, and invented the "escape" sequence in programming. You can thank him, or blaspheme him, for the ESC key.

In the weenieworld of data processing, he is a minor deity.

We had guessed Bemer would be reassuring about the Y2K problem.

Stay informed with a subscription to The Washington Post.

Subscribe

Our first question is why the heck he recently moved from a big city all the way out to East Bumbleflop, U.S.A.

It's a good place to be next New Year's Eve, he says. From a kitchen drawer he extracts two glass cylinders about the size of the pneumatic-tube capsules at a drive-through teller. Each is filled with what appears to be straw.

"They're Danish," he says. "They cost \$500. We ran water with cow[poop] through them and they passed with flying colors."

They're filters, to purify water. If Y2K is as bad as he fears, he says, cocking a thumb toward his backyard, "we can drain the lake."

Bemer is 79. He looks flinty, like an aging Richard Boone still playing Paladin.

He has started a company, Bigisoft, that sells businesses a software fix for the Y2K problem. So, for selfish reasons, he doesn't mind if there is widespread concern over Y2K, though he swears he really thinks it is going to be bad. That's why he has requested that we not mention the town in which he lives. He doesn't want nutballs descending on him in the hellish chaos of Jan. 1, somehow blaming him.

Who, then, is to blame?

Bemer rocks back in his chair and offers a commodious smile.

In one sense, he says, he is.

Binary Colors

In the late 1950s, Bemer helped write COBOL, the Esperanto of computer languages. It was designed to combine and universalize the various dialects of programming. It also was designed to open up the exploding field to the average person, allowing people who weren't mathematicians or engineers to communicate with machines and tell them what to do. COBOL's commands were in plain English. You could instruct a computer to MOVE, ADD, SEARCH or MULTIPLY, just like that.

It was a needed step, but it opened the field of programming, Bemer says, to "any jerk."

"I thought it would open up a tremendous source of energy," he says. "It did. But what we got was arson."

There was no licensing agency for programmers. No apprenticeship system. "Even in medieval times," Bemer notes dryly, "there were guilds." When he was an executive at IBM, he said, he sometimes hired people based on whether they could play chess.

Stay informed with a subscription to The Washington Post.

Subscribe

There was nothing in COBOL requiring or even encouraging a two-digit year. It was up to the programmers. If they had been better trained, Bemer says, they might have known it was unwise. He knew.

He blames the programmers, but he blames their bosses more, for caving in to shortsighted client demands for cost-saving.

"What can I say?" he laughs. "We're a lousy profession."

Some contend that the early programmers were unconcerned about the year 2000 because they expected their programs to last only a few years. If that is true, it was naive. Computers are forever becoming obsolete, replaced by faster, better technologies, but the programs they run can be nearly immortal. A good program is self-perpetuating, tested over time, wrinkles ironed out through updates, a solid foundation for all that follows. The house above it may be fancified, with spiffy new wings and porticoes, but the foundation remains. Which goes to the heart of the Y2K problem.

The longer a program is used, the larger the database and supporting material that grow around it. If, say, a program records and cross-references the personnel records in the military, and if the program itself abbreviates years with two digits, then all stored data, all files, all paper questionnaires that servicemen fill out, will have two-digit years. The cost of changing this system goes way beyond the cost of merely changing the computer program.

It's like losing your wallet. Replacing the money is no sweat. Replacing your credit cards and ATM card and driver's license and business-travel receipts can be a living nightmare.

And so, even after computer memory became cheaper, and data storage became less cumbersome, there was still a powerful cost incentive to retain a two-digit year. Some famously prudent people programmed with a two-digit date, including Federal Reserve Chairman Alan Greenspan, who did it when he was an economics consultant in the 1960s. Greenspan sheepishly confessed his complicity to a congressional committee last year. He said he considered himself very clever at the time.

In their omnibus 1997 manual for lawyers planning Y2K litigation--an excellent if unnerving document of 600-plus pages--attorneys Richard D. Williams and Bruce T. Smyth suggest that IBM and other computer manufacturers might be partially at fault for not addressing the problem in the early '60s by advising their customers of the wisdom of a four-digit year. In 1964, IBM came out with its System/360 computers, which revolutionized the industry. It built upon existing programs, yet required much new software. Should IBM have seized the moment to make things right?

"That would have been stupid," responds Frederick Brooks, a University of North Carolina computer science professor. In the 1960s, Brooks was IBM's project manager for the System/360.

The average 360, he says, had either 16 or 32 kilobytes of memory, 12 of which were needed to run the operating system. What was left was less memory than is available today in a hand-held personal

Stay informed with a subscription to The Washington Post.

Subscribe

"I never heard anyone seriously propose a four-digit year," he recalls. It is not as if a two-digit year was set in stone anywhere, he says. It just became a logical convention, across the industry.

So Y2K was inevitable?

No. As time passed and memory became cheaper and the end of the century got closer, Brooks says, "the cost of using four-digit years went down gradually, and the wisdom of using them went up gradually."

When did the two lines cross on the graph?

Around 1970, he says. But competitive pressures kept managers from making that expensive decision. By the mid-1980s, it was too late. Computers were everywhere, their programs hopelessly infected with the problem.

Could anything have changed corporate attitudes earlier?

The former IBM man ponders this.

"If we had adopted industry-wide standards by some standards group, standards everyone would have had to follow, there would be no competitive pressures for cost." But nothing like that ever happened, he says.

Actually, Brooks is wrong. Something very much like that happened. A group did adopt a written standard for how to express dates in computers.

We are looking at it now.

It is a six-page document. It is so stultifying that it is virtually impossible to read. It is titled "Federal Information Processing Standards Publication 4: Specifications for Calendar Date." It is dated Nov. 1, 1968, and took effect on Jan. 1, 1970, precisely when Brooks says the lines on the graph crossed, precisely when a guiding hand might have helped.

On Page 3, a new federal standard for dates is promulgated.

Sometimes, someone makes a reasonable-sounding statement that, in the merciless glare of history, seems dreadfully unwise: "Separate but equal" is one of these. Also: "I believe it is peace in our time," an opinion rendered by Neville Chamberlain weeks before the outbreak of World War II.

Federal Information Processing Standards Publication 4, Paragraph 4 and Subparagraph 4.1, is another of those statements. Here it is, in its entirety:

Stay informed with a subscription to The Washington Post.

Subscribe

Calendar Date is represented by a numeric code of six consecutive positions that represent (from left to right, in high to low order sequence) the Year, the Month and the Day, as identified by the Gregorian Calendar. The first two positions represent the units and tens identification of the Year. For example, the Year 1914 is represented as 14, and the Year 1915 is represented as 15.

Ah.

The Y2K problem.

Set in stone.

By the United States government.

FIPS 4, as it was called, was limited in scope. It applied only to U.S. government computers, and only when they were communicating from agency to agency. Still, it was the first national computer date standard ever adopted, and it influenced others that followed. It would have affected any private business that wanted to communicate with government computers. It might have been a seed for change, had it mandated a four-digit year.

It was a missed opportunity. Who screwed up?

The Standard Bearers

Harry S. White Jr., 64, places a briefcase on the table. It is heavy. He has documents.

We are meeting in a conference room at a Holiday Inn in Morgantown, W.Va., to plumb ancient history. White helped write FIPS 4; at the time he was with the National Bureau of Standards.

White says he is pleased to meet us. He holds out a hand. In it is a Bible.

"Be careful with that," he says mildly. "It's powerful. If you open it, it will have an impact on your life."

White is West Virginia chairman of The Gideons International, the gentlemen's organization that places Bibles in hotel rooms. He is now semi-retired, but for much of his life he was an expert on standardizing computer codes, a scientist whose field involved the proper sequencing of digits and symbols. God, they say, is in the details.

In the 1960s and '70s, White was one of a few dozen computer experts who met regularly on committees to try to get government and industry to use identical conventions in programming. It was an important job, but a thankless one. Programmers sometimes consider themselves as creative as novelists; to them, standards experts are squinty-eyed, pencil-necked editors--necessary, perhaps, but nit-picky and annoying.

Stay informed with a subscription to The Washington Post.

Subscribe

Harry White says that back in 1968, he was opposed to a two-digit year. He did not exactly foresee the extent of the Y2K problem but there was something about two digits that offended his sense of the rightness of things: "If it is four digits," he says, "it is everlasting."

But FIPS 4 was produced by a committee, White explains. A committee. When a committee tries to design a horse, it can come up with a jackass.

On the committee were representatives of several government agencies, among them the Office of Management and Budget, NASA, the General Services Administration and the Department of Defense. Defense was by far the biggest computer user in the federal government, probably in the world, White says, and its input was disproportionately influential. The Defense Department, he says, opposed the four-digit year because it would have meant rewriting all its programs, and all the supporting data. Defense had bigger worries. We were neck deep in Vietnam.

Besides, White says, there was a much larger issue on the table: the precise order in which the day, month and year would be written. DOD wanted to keep its system, familiar to Europeans and the American military: day/month/year. Others wanted the standard month/day/year sequence, the way Americans write it on personal correspondence. Whether years would be four digits or two seemed a minor matter. Even those people like Bemmer and White, who sensed a problem, had no real understanding of its potential scope: In the 1970s few people anticipated how thoroughly computers would come to dominate our lives.

Eventually, White says, Defense gave up on the issue of the order of the date, but it held fast on the two-digit year.

Three years later, the American National Standards Institute issued its own voluntary standard for expression of date in computer language. This was ANSI standard X3.30, which was drafted by, and for, both government and industry. Harry White was chairman of the subcommittee that addressed the issue of date. The Defense Department, White says, remained solidly opposed to change: It stuck to its guns, as it were.

The initial proposal was for a two-digit year, just like FIPS. But eventually, White said, he and others prevailed. The final standard was for a four-digit year, including the prefix 19- or 20-. But as a compromise with the Defense Department, White says, the Standards Institute added an option: Programmers could stick with a two-digit year if they wanted to.

That gave everyone an out. In essence, government and business programmers could choose to adopt the recommended standard, at the cost of many millions of dollars, or they could ignore it completely, without technically having committed a sin.

"That," says Robert Bemmer, "was devastating. It was an excuse to put it on the shelf."

Who screwed up? Was anyone in particular behind this?

Stay informed with a subscription to The Washington Post.

Subscribe

"The director of data standards for the Office of the Secretary of Defense. I used to work for him."

Who was he?

"I don't want to give the impression that I was a hero and he was a bad guy. There was just a difference in making judgments and decisions."

Give us a name, Harry.

"Bill Robertson. He married his assistant, Mildred Bailey. "

Harry and Bill

Bill and Mildred are amiable, despite being ambushed in their jammies in their Baileys Crossroads apartment. They are wearing socks and slippers. She is redheaded, lean and energetic. He is solidly built, a little deliberate afoot.

We tell them why we are there.

"Anyone who says the Department of Defense was against the four-digit year is full of crap," Bill Robertson says. "Harry White made that up out of his own imagination, whole cloth." The issue never came up, Robertson said, at least not exactly that way.

Robertson and Bailey both deny their office was ever even consulted on the FIPS 4 regulation, though it did have input into the ANSI standards. Robertson says he does not recall ever being asked to comment specifically on a four-digit year, though he agrees the Department of Defense did in general oppose major changes to its computer system. Change would have been costly. The various armed services would not have stood for it.

"We would have had to change every stinking file," Bailey says.

"We would have had a revolt," Robertson says. If someone had ordered them to change, "we would have said, 'Blow it out your airbag.' "

However, it was all moot, he claims. The Department of Defense already had a system for recording the date, a system Robertson helped develop and implement back when he was in the Air Force. Robertson wanted it to be a national system.

What was their system?

It had a two-digit year in it, he says.

Aha!

Stay informed with a subscription to The Washington Post.

Subscribe

But, Robertson says, his system included something else. A date was designated by "data elements." The month, year and day were only three elements of five. There was another element, for optional use, that would have indicated which century it was, and yet another indicating which millennium. If you chose to put those in, it would tell the computer to distinguish between centuries. It was the solution to the Y2K problem, but it was never adopted nationally.

Bill's system never would have worked, Harry replies: "See, this is where we ran into that kind of problem with him! This was his definition of data elements, but the rest of the world would not accept this definition!"

Harry says Bill was "a very narrow, bullheaded individual. When it came to matters of being able to compromise, he was totally inflexible."

Bill says Harry was the bullheaded one. He wouldn't listen to reason. Wouldn't join him in his data elements program. "We had the answer in 1964. Harry never tried to get on board!"

Once, Bill says, Harry got into a shouting match with one of Bill's deputies on a philosophical dispute about how to express the concept of midnight. It nearly came to blows.

Harry says Bill was envious of him because he eventually rose above Bill, his former boss, to a position of higher authority in the field of data standards: "He never got over it," Harry says.

Bill says Harry was the envious one, ever since the day Bill beat him out for the Department of Defense standards job.

"Harry and I interviewed for the same job. Has it occurred to you why I got it and he didn't? He didn't understand standards!"

Did too, Harry says.

Did not, Bill says.

James Gillespie was a computer standards man for the Navy. He worked with both White and Robertson, on ANSI deliberations. He liked them both, he said, but the two men could not get along.

"They had a personality conflict that impeded progress," Gillespie said.

For some danged reason, the negotiations over computer date lasted a very, very long time. And for some danged reason, nothing very handsome was accomplished.

In the end, what was produced was FIPS 4 and ANSI X3.30, neither of which protected the world against Y2K.

Stay informed with a subscription to The Washington Post.

Subscribe

It may be the only thing in the whole entire world they agree on.

File Not Found

We've tried to further research this Harry-Bill contretemps. Many of the participants are dead; others' memories are indistinct. Harry says there should be a paper trail showing the Defense Department's complicity in all this--but the official government file on the FIPS 4 document is as thin as leaf. There's no paper trail.

Harry suspects chicanery: He theorizes the records were either "shredded or placed where they are not in the public domain."

A spokesman for the National Institute of Standards and Technology, keeper of the FIPS files, confirms that other FIPS regulations have bulging folders, but not FIPS 4. He does not know why, but says there is no evidence any larger file ever existed. There is certainly no coverup, he said.

Ruth Davis is president of the Pymatuning Group, a technical management firm in Alexandria. In the 1970s, she was Harry White's boss at the National Bureau of Standards. She remembers Harry being apoplectic at the intransigence of the Department of Defense on the issue of the four-digit year. But she says she never really blamed DOD. The cost, she said, would have been huge.

Davis had once worked for Defense, and understood the necessity of saving space. At times, she said, it was a life-or-death priority. Back then, she said, Defense had to maintain control of rockets during their launches. Calculations had to be made in real time. This required quickness, which required computer memory. They couldn't screw around with four-digit dates.

She said it would be wrong to blame any one person at DOD. It was policy, top to bottom. Plus, it made sense.

So we can't blame Bill?

"You can't blame anyone."

Damn.

Tomorrow Is Another Day

Maybe we're looking at this thing all wrong. Maybe it isn't about people, at all. Maybe it is just about numbers.

Maybe, in the early days, there simply never was a good solution to a basic problem of space: A six-digit date was much more economical than eight. Maybe a problem at century's end was unavoidable, since you could not possibly express the date unambiguously in six digits alone.

Stay informed with a subscription to The Washington Post.

Subscribe

Except, you could. Astronomers do. They deal in distances so vast that light takes millions of years to traverse them. So astronomers are forever having to add and subtract time periods that span centuries. Since the 1700s, they have found a simple way to do this, with a minimum of figuring or adjustment for leap years and the like: They use something called the Julian day number, adapted from the ancient Julian calendar.

In this system, the day Jan. 1, 4713 B.C., is arbitrarily taken as Day 1, the beginning of time. And every day thereafter is numbered sequentially, as a single number. For example, Jan. 1, 2000, the day of the presumed Y2K Armageddon, would be Julian Day 2,451,545.

In Julian day calculations, there is never a need for Month, Day, or Year. There is no ambiguity about centuries, because there is no century. Julian day numbers are, at least theoretically, the perfect solution to the Y2K problem.

The modern Julian day number is seven digits long. But, if you used it in computers, you could safely drop the first one. That abbreviation would eventually create a Y2K-type ambiguity, but that ambiguity would not occur until A.D. 3501, when the Julian date would hit 3,000,000. By then we might all have big, bald heads and no teeth and do our computing telepathically.

If the Julian day had been used in computers--it could have been since 1963, when an algorithm was written to perform the conversion automatically--it would have conserved memory. For microprocessor chips, no conversion would even have been necessary; they could have been programmed directly with the Julian date.

This was actually considered.

Thomas Van Flandern, an astronomer at the University of Maryland, believes that if data processors had adopted the Julian date in 1963, the Y2K problem would not exist.

In fact, he says, this was once a hot topic among astronomers. They wanted to recommend it: "There was a lot of discussion about it at the Jet Propulsion Laboratory," he recalls. "But it broke up into small pools."

Astronomers simply couldn't get together on it, Van Flandern says. Basic philosophical disputes arose. The movement disintegrated, he says, because it became polarized. On one side were those who wanted to change all calculations, such as expressing all angles not in degrees but in radians. On the other side were people who wanted to change nothing. They fought. Those simply advocating a Julian date for computers were lost in the din. Nothing got done.

And the Julian date issue died.

So maybe the Y2K problem is about people after all.

Stay informed with a subscription to The Washington Post.

Subscribe

We had one more lead on someone to blame. A last-ditch theory. It was a good theory. It promised us a fabulous villain. We were excited.

In the early 1970s, Robert Bemer remained bugged, as it were, by the problem of the two-digit year. He discussed it with acquaintances. One of these was Edward David, the science adviser to the president of the United States.

Bemer says he urged David to take the matter to the White House. To the president himself, if need be.

The president was . . . Richard Nixon.

Clearly, this merited further investigation.

Edward David is 74. He is president of EED Inc., a computer consulting firm in Bedminster, N.J.

Yes, he recalls, Bemer did discuss the two-digit year with him. And yes, David agreed with Bemer that it might be a problem. "I know computers," David says. "I know how stupid computers are."

And yes, David says, Bemer urged him to take it up at the highest levels.

Did he talk about it with, y'know . . . Nixon?

"No."

So much for the perfect villain.

"I discussed it with my staff," David says. "I discussed it with some other agencies." He certainly talked to people in the Office of Management and Budget, he says, and possibly in John Ehrlichman's office, or George Shultz's. David does not recall names, but he recalls the reaction. People, he says, "wagged their head sagely and said this problem is simply not on the radar screen."

In particular, he remembers this fairly universal response:

"It's 30 years in the future. We'll be out of office. Leave it to the civil servants. They'll still be here."

The Sting of the Bug

It's not my problem. It's not on my watch. He's full of crap. They're jerks. He won't listen to reason. She's jealous. What's he trying to pull? Blow it out your airbag.

A people problem.

No one wanted the Millennium Bug. No one hatched it. But no one bottled it up when they had the chance. **Stay informed with a subscription to The Washington Post.**

Subscribe

It's the same way with warfare: No one wants it. Everyone tries to avoid it. And here it is.

The Y2K problem is not a computer problem, after all. It was not hard-wired into the mechanical brains themselves, as some have contended. It was hard-wired into the human brain. We want to be enlightened. But our wisdom falls victim to greed and hostility and covetousness and expedience. It's human nature.

A people problem.

We didn't want a people problem. We wanted a person problem. Someone to blame.

With Y2K, there is only one fact about which most everyone agrees: It happened in large measure because computers were invented in the center of the century. It was an accident of timing.

The first electronic digital computer, ENIAC, was unveiled in 1946. Let's say this had occurred in 1996. The next century would have been right around the corner, barreling at us. Yes, some programs would have been able to ignore it, but the majority would not. Simple mortgages would have had to accommodate the new century. The balance would be tilted. The state of the art would have to be the four-digit date, despite the cost. Few computer experts doubt this.

And if computers had been invented in, say, 1912, the same thing would have happened in reverse. The birthdays of 80 percent of the American population would have had to be expressed as part of a previous century. Arithmetic involving ages, dates of employment, home purchases, anything that looked remotely into the past would have similarly had to account for the 1800s.

But where does this get us? It's impossible to second-guess the march of progress. Science proceeds at its own pace. Inventions beget other inventions. Computers happened when they were ready to happen. Not before or after.

But why did that moment fall at the center of the century? Can the calendar itself be second-guessed?

S. Thomas Parker is a professor of history at North Carolina State University. He is an expert on time measurement. We got his name from an Internet search. (Computers. They're magic.)

We explain our predicament. We need to find someone to blame for the fact that the year 2000 is arriving in six months, and not at some other time. In other words:

Why 2K?

Parker thinks about this.

He consults a book.

Stay informed with a subscription to The Washington Post.

Subscribe

Dennis the Menace?

Most likely, he dressed in coarse brown robes woven from hemp. He was a Scythian monk who lived in Rome in the 6th century A.D. His name was Dionysius Exiguus, which translates, roughly, into "Dennis the Short." Dennis may well have been a small man, but scholars suspect he took the moniker as a sign of humility.

Parker explains that before Dennis the Short, time was reckoned in various ways; some figured the date by the number of years since the election of the current pope. The most common system for counting time, however, was dating it from the founding of Rome in what is now considered 753 B.C.

Dennis the Short is widely credited with having created the modern calendar. In A.D. 525, he is said to have proposed dating the Christian era from the birth of Jesus, and persuaded the papacy this was a good idea. Dennis calculated this to be the year we now call A.D. 1. It took centuries, but eventually this system was adopted throughout the Christian world.

But Dennis was wrong, Parker says. He miscalculated. If the Scriptures are to be believed, Jesus was certainly born during the reign of Herod the First, the king who ordered the death of all male babies in Judea after hearing of the birth of a messiah. Herod died in 4 B.C. That means Jesus was born at least four years earlier than Dennis reckoned. Which means all dates should be four years later than we think.

Not good enough. It would not have mattered appreciably if computers had been invented in 1950 instead of 1946.

Parker considers this.

Well, he asks, why did Dennis the Short fix the start of the Christian era at the birth of Christ? "Resurrection is the true beginning," he says.

Good point. Christ died a Jew. His last supper was a seder. The Christian era should begin not with his birth but his death.

He is thought to have died around A.D. 34, during the latter years of the tenure of Pontius Pilate, Judea's Roman prefect.

Shorty turned over the hourglass 34 years too soon!

Let's recalculate time.

A.D. now means what schoolkids have always thought it meant: After Death. The U.S. was birthed in Philadelphia not in 1776 but 1742. The Civil War began in 1826. The stock market crashed in 1895.

Stay informed with a subscription to The Washington Post.

Subscribe

Pretty soon thereafter, the Department of Defense had a problem. It really, really, wanted to program its computers using a two-digit year. But gosh darn it, this just wasn't practical. Half of all servicemen were born in the previous century. Industry faced similar problems. When they could, programmers still used a two-digit date. But most could not. The four-digit year became the rule, not the exception.

Today is Sunday, July 18, 1965. The century will not end for 34 years. But computers will have been programmed correctly. There will be no millennium bug.

The Villain, Unmasked

It's not Dick Nixon. It's not Bob Bemer. It's not Ed David. It's not Alan Greenspan. It's not Bill. It's not Harry.

It's Shorty.

He's the one who screwed up.

Big time.

Special correspondent Bob Massey contributed to this report.

CAPTION: Above, Robert Bemer, the computer pioneer who helped write COBOL, an intentionally simple language. It opened the field of programming to "any jerk," Bemer says. "I thought it would open up a tremendous source of energy. It did. But what we got was arson." Bemer has moved from Arizona and lives near a lake he can drain for drinking water in case of Y2K emergencies. Right, Edward David, former science adviser to President Richard Nixon. He tried to bring up the Y2K problem with colleagues. But people turned a deaf ear. "It's 30 years in the future," people replied. "We'll be out of office. Leave it to the civil servants. They'll still be here."

CAPTION: Above, Mildred Bailey and Bill Robertson. Robertson says he had the solution to the Y2K problem back in 1964, but people turned a deaf ear. Particularly Harry White. Left, Harry White, who had a front-row seat in the 1960s bureaucratic skirmish over the two-digit year. He was opposed. "If it is four digits," he says, "it is everlasting." That might have been the solution to the Y2K problem, he said, and people turned a deaf ear. Particularly Bill Robertson.

Stay informed with a subscription to The Washington Post.

Subscribe

Stay informed with a subscription to The Washington Post.

Subscribe

Stay informed with a subscription to The Washington Post.

Subscribe