LETTER FROM THE EDITOR

RISKS TO THE PUBLIC IN COMPUTER SYSTEMS

Taming the Tiger

Witold Rybczynski's book, *Taming the Tiger -- the Struggle to Control Technology* (Viking 1983, Viking Penguin 1985) concludes with this paragraph:

> Whether we control technology by directing its evolution, by choosing when and how to use it, or by deciding what significance it should have in our lives, we shall succeed only if we are able to accept what at first appears to be an impossible shift in our point of view: different as people and machines are, they exist not in two different worlds, but at two ends of the same continuum. Just as we have discovered that we are a part of the natural environment, and not just surrounded by it, so also we will find that we are an intimate part of the environment of technology. The auxiliary *organs* that extend our sight, our hearing, and our thinking really are an extension of our physical bodies. When we are able to accept this, we shall discover that the struggle to control technology has all along been a struggle to control ourselves.

On Assessing Risks

The past quarter has been one of deep reflection for me, which among other things results in this rather pensive message on how difficult it is to predict the future based only on our (incomplete) knowledge of the past and the present.*

On 13 October 1985, around 10 PM in Florence, Italy, my seemingly very healthy 19-year-old son Chris -- who had never had a medical problem in his life -- suddenly had his heart stop, and died within moments. Resuscitation attempts failed. The autopsy found no discernible cause of death. A neurophysiologist friend who joined me in Florence suggested ventricular fibrillation as a likely cause. (The signals to the heart generally arrive sufficiently synchronously to trigger heart contraction. Under fibrillation, the signals arrive incoherently and cannot be integrated properly.) Other possible explanations might include anaphylactic shock, such as that due to a critical allergy. This sudden turn of events reminds me once again that even in an apparently completely healthy system there always exists some risk of spontaneous malfunction -- and that even afterwards it may in some cases be impossible to find out with certainty what caused the malfunction.

In making arguments about how hardware and software will operate correctly or will continue to operate correctly (if they ever did), we make all sorts of implicit underlying assumptions that may be true most of the time, but which may indeed break down -- particularly in times of stressed operation. The nastiest problems of all seem to involve unanticipated conditions in timing and sequencing, in both synchronous and asynchronous systems, and particularly in distributed systems. We have seen various problems in the past -- the ARPANET collapse (due to an accidentally propagated virus, after years of successful operation) and the first shuttle launch immediately come to mind as specific well-documented examples. In some cases there is also signal interference -- as in the pacemaker problems (see Nancy Leveson's contribution below). I think that in our lives as in our computer systems, we tend to make unjustified or oversimplified assumptions. In life, such assumptions make it possible for us to go on living without inordinate worry (paranoia). In computer systems, however, greater concern is often warranted -- especially if a system must meet its requirements under all possible operating conditions. Thus, it seems essential that we try to make our assumptions about computer systems and their use both more explicit and more realistic. Basing a system design on assumptions

---

*Apologies to our on-line RISKS Forum readers; because I was away during most of the quarter, the material here is culled largely from the on-line forum.

that are almost but not quite always true may seem like a close approximation, but may imply the presence of enormous unanticipated risks. In systems such as those involved in Strategic Defense, for example, many potentially critical assumptions have to be sufficiently correct.

We understand relatively little about long-term effects (whether they eventually manifest themselves as very obvious effects or remain undetected as invisible side-effects). However, in some cases potential risks have been recognized by people within an organization, but that knowledge has been willfully suppressed.

The front page of the NY Times on Sunday, 1 Dec 1985, has two articles on the Bhopal disaster, one year later. Stuart Diamond's article begins, "Medical studies conducted in the year since the chemical leak ... indicate that the chemical responsible for the accident causes serious long-term health problems that were unknown before the disaster." Furthermore, the Bhopal problems appear to have been due not just to the pesticide ingredient methyl isocynate, but to an unforeseen chemical reaction that transformed some of it to hydrogen cyanide. (An antidote to the latter chemical was available, but was not recognized to have been appropriate until months afterwards.)

In various past issues of ***SEN***, we have noted risks such as side-effects of pacemaker interference; auto microprocessor bugs; command and control computer problems; and so on -- perhaps ad nauseum to some of you -- and the dangers of making a small set of assumptions that underly proper system behavior. I have also alluded occasionally to lessons that we might learn from environmental risks such as toxic substances in our food, drink, and environments; some of those risks were known in advance but ignored -- e.g., for commercial reasons; others came as "surprises" (thalidomide, for example), but probably represented a lack of care and long-term testing. In some cases the risks were thought of, but considered minimal. In other cases, the risks were simply never considered.

This note merely adds a plaintive cry for greater humility. Science (especially computer science) does not have *all* the answers. Furthermore, the absence of any one answer (and indeed suppression of a question that should have been asked) can be damaging. But, as we see from the nature of the problems to date, some of us too often keep our heads in the sand -- even after being once (or multiply) burned. Eternal vigilance is required of all of us. Bureaucrats and technocrats who say "don't worry, nothing can go wrong" must be exposed. But technocrats who say "we can't do it at all"also need to be very careful in their statements; in the small, anything is possible. However, we must remember that systems often tend to break down when operating under conditions of stress -- particularly when global system integration is involved. Furthermore, it is under the same conditions among people when rational arguments also tend to break down.

Gloria in Exchange's Day-oh: Lack of a backup computer closes stock exchange

Marty Moore noted that when Hurricane Gloria was approaching the New York area, the New York and American Stock Exchanges did not open (in anticipation of the storm). The Midwest Exchange, located in Chicago, opened on schedule; unfortunately, it had to close 40 minutes later, when its nationwide computer system failed. Where is the central computer of that system located? New York, of course. The Director of the Exchange was quoted as saying, "Well, this has got to change."

Gentlemen Prefer Platinum to Bonds -- $32 Billion Overdraft

A Computer Snafu Snarls the Handling of Treasury Issues
Phillip L. Zweig and Allanna Sullivan, staff reporters
Wall Street Journal, Monday 25 November 1985 [quoted without permission]

> NEW YORK- A computer malfunction at Bank of New York brought the Treasury bond market's deliveries and payments systems to a near-standstill for almost 28 hours Thursday and Friday. Although bond prices weren't affected, metal traders bid up the price of platinum futures Friday in the belief that a financial crisis had struck the Treasury bond

market. However, Bank of New York's problems appeared to be more electronic than financial.

The foul-up temporarily prevented the bank, the nation's largest clearer of government securities, from delivering securities to buyers and making payments to sellers -- a service it performs for scores of securities dealers and other banks. The malfunction was cleared up at 12:30 p.m. EST Friday, and an hour later the bank resumed delivery of securities.

But Thursday the bank, a unit of Bank of New York Co., had to borrow a record $20 billion from the Federal Reserve Bank of New York so it could pay for securities received. The borrowing is said to be the largest discount window borrowing ever from the Federal Reserve System. Bank of New York repaid the loan Friday, Martha Dinnerstein, a senior vice president, said. Although Bank of New York incurred an estimated $4 million interest expense on the borrowing, the bank said any impact on its net income "will not be material." For the first nine months this year, earnings totaled $96.7 million. Bank of New York stock closed Friday at $45.125, off 25 cents from the Thursday, as 16,500 shares changed hands in composite trading on the New York Stock Exchange. Bank of New York said that it had paid for the cost of carrying the securities so its customers wouldn't lose any interest. Bank of New York's inability to accept payments temporarily left other banks with $20 billion on their hands. This diminished the need of many banks to borrow from others in the federal funds market. Banks use the market for federal funds, which are reserves that banks lend each other, for short-term funding of certain operations. The cash glut caused the federal funds rate to plummet to 5.5% from 8.375% early Thursday.

The electronic snafu is by far the largest of computer problems that periodically have bedeviled the capital markets. Almost all government securities transactions are settled electronically through the New York Federal Reserve Bank. In this system, computers of clearing banks are linked to one another through a central computer, to enable banks to settle purchases and sales of securities by customers. According to Wall Street sources, the malfunction occurred at 10 a.m. Thursday as Bank of New York was preparing to change software in a computer system and begin the days operations. Until Friday afternoon, Bank of New York received billions of dollars in securities that it couldn't deliver to buyers. The Fed settlement system, which officially closes at 2:30 p.m., remained open until 1:30 a.m. Friday in the expectation that technicians would be able to solve the problem.

Rumors about bank problems often send commodity traders scurrying to buy precious metals. In the platinum pit at the New York Mercantile Exchange, the price for January delivery surged $12.40 an ounce to $351.20 Friday on volume of 11,929 contracts, a 29-year record. Reports that the Fed was investigating transfer problems at Bank of New York prompted the platinum buying. [WSJ]

Peter G. Trei contributed the above article, along with the following commentary:

I talked to a friend of mine who was peripherally involved in the recovery from this 'snafu', and it seems that the primary error occurred in a messaging system which buffered messages going in and out of the bank. The actual error was an overflow in a counter which was only 16 bits wide, instead of the usual 32. This caused a message database to become corrupted. The programmers and operators, working under tremendous pressure to solve the problem quickly, accidently copied the corrupt copy of the database over the backup, instead of the other way around.

One thing I have often noticed is that the 'normal run' code of software packages tends to get much more through testing then the code for error recovery; not only is it more difficult to test, but the general feeling of 'this code will never execute' demotivates programmers. In this case, it sounds like the people at BONY never held a 'fire drill' to figure out how to handle a corrupt primary database. Does anyone else have examples where attempts at

error recovery magnified problems? (Peter Trei)

A few weeks later, more details had emerged [Washington Post, 13 December 1985, p. D7] (submitted by Al Friend), in an article titled "Computer Snarled N.Y. Bank -- $32 Billion Overdraft Resulted From Snafu" (by John M. Berry, Washington Post Staff Writer). [Please excuse some duplication with the above report, athough the historical evolution of the story is interesting as well.]

> The Bank of New York, the nation's 18th largest, had a brief $32 billion overdraft on its cash account at the New York Federal Reserve Bank when a computer failure last month snarled thousands of government securities transactions, a congressional committee was told yesterday. By the end of the day, the overdraft had been reduced to $24 billion, and the bank actually had to borrow that amount from the New York Fed -- pledging all of its assets -- in order to balance its accounts overnight.
>
> Aside from the unprecedented scale of the borrowing, and the spillover effects on the government securities market, the incident intensified concern at the Federal Reserve over the vulnerability of the nation's financial payments system to a technological glitch that could have disastrous consequences. Federal Reserve Chairman Paul A. Volcker and New York Fed President E. Gerald Corrigan went before a House Banking subcommittee yesterday to describe how the computer failure occurred and how the Fed and the bank dealt with the crisis it caused.
>
> On Wednesday, Nov. 20, transactions involving more than 32,000 different government securities issues poured into the Bank of New York, one of the largest processors of such deals on behalf of others. The bank's computer system was supposed to be able to cope with up to 36,000 issues, but a programming glitch developed and, unknown to anyone, the computer began to "corrupt" the transactions and make it impossible for the bank to keep them straight. Because of the computer system breakdown, the bank could not instruct New York Fed where to send the securities arriving at the Fed on behalf of the bank's clients, and therefore could not get paid for them. The New York Fed was automatically taking money out of the Bank of New York's cash account to pay the sellers for the incoming securities, all of which are represented simply by computer records, rather than the familiar paper bonds still used by most corporations. By Thursday evening, as hundreds of employees at a host of banks and government securities dealers tried to sort out the problems caused by the failure of the intricate and largely automatic network handling these transactions, the bank had a $32 billion overdraft on its cash account at the New York Federal Reserve Bank. The bank's computer specialists finally came up with a "patch" for its computer program -- a process described yesterday by its chairman, J. Carter Bacot, as the electronic equivalent of patching a tire -- that allowed it to begin to clear some of the backlog. But just after midnight, the patch failed too, after the overdraft had been whittled down to about $24 billion.
>
> The Fed kept both its nationwide wires for securities and cash transactions open in the early hours of Friday morning. When the patch failed, the Bank of New York was still able to borrow $700 million from other banks. The rest was covered by a $23.6 billion loan from the New York Fed. As collateral, the bank pledged all its domestic assets and all its customers' securities it was allowed to use for such purposes. Altogether, the collateral was worth $36 billion, according to the Fed.
>
> The drama was not over. Around 5 a.m. Friday, the bank finally completed reconstruction of its customers' transactions from Wednesday. By 10 a.m., it had done the same for the Thursday deals. But, meanwhile, the rest of the government securities industry had begun its Friday activities, and securities and an overdraft were piling up again in the Bank of New York's account at the New York Fed. "Faced with this situation," New York Fed President Corrigan told the banking subcommittee, "at about 11:30 a.m., we temporarily stopped accepting securities transfers for the account of Bank of New York in an attempt

to stabilize the situation somewhat and to see whether it was practical to prevent further increases in the overdraft without causing excessive disruption in the market more generally. . . . Operationally, this meant that holders of government securities who had contracts to deliver those securities . . . to the Bank of New York for one of its customers [in return for payment] were temporarily unable to make delivery under those contracts," Corrigan said.

The stoppage lasted only for about 90 minutes that afternoon, and news of it did not spread widely for nearly an hour. Yet that disruption at the clearing bank was enough, Corrigan said, to make some market participants unwilling to trade securities among themselves. "Perhaps most importantly, there was also some evidence that investors were beginning to seek to break trades and financing transactions with dealers serviced by the Bank of New York." Shortly after noon, the Bank of New York was able to begin handling the Friday transactions that had been piling up, and the Fed was again able to accept securities destined for the bank. By that point the bank was operating with a computer system that had undergone a major overhaul in less than 24 hours.

The crisis was over, but its final bill is still mounting. The Bank of New York was out of pocket about $5 million, an amount equal to about 7 percent of its earnings in the first nine months of this year, to pay interest on the money it had to borrow that Thursday. It is still negotiating with many of the parties who may have sustained losses in transactions that were not completed on time. Such negotiations are common, said an official of one major securities dealer, because a few transactions are always going awry. This time it was thousands.

Some customers walked away in better shape. "Indeed, those individuals and institutions who bought securities in question received a windfall in that they received interest for a day [on the securities], but did not incur any cost of financing," Corrigan noted. But any loss or gain in dollars, even with millions of dollars at stake, is not the real issue. What worries both Federal Reserve officials and participants in the government securities market is the potential for a failure of the system.

On the average day, about $200 billion worth of government securities transactions take place involving about 27,000 separate transactions, Corrigan said. Some days the totals are far larger. "Like it or not," Volcker told the subcommittee, "computers and their software systems -- with the possibility of mechanical or human failure -- are an integral part of the payments mechanism. The scale and speed of transactions permit no other approach. . "In the last analysis, no mechanical system can be entirely 'fail-safe' and also be commercially viable," he said. "The costs would simply be too high, and the money and Treasury securities markets could not operate at the present level of efficiency." The Fed chairman pointed out that, in this case, the Fed was available to lend the $23.6 billion, on good collateral. "The effects in this instance were of unprecedented magnitude, measured by the amount of the overnight loan," he said. "But the effects in terms of market performance and risk were well contained. . . . I believe it would be wrong to overdramatize this incident."

Corrigan in his more detailed testimony sounded more notes of concern. "I believe our actions were prudent, disciplined and appropriate. In saying this, I should also confess that in some respects we were a bit lucky," he said. Part of the luck was that the bank was able to get its computer going again as soon as it did. Another part, Corrigan said, was that Thursday was not an especially heavy day for securities transactions.

One government securities trader summed up the situation this way. "We're all afraid something will go bump and send the market into a tailspin. . . . The Fed is working night and day to figure out what it can do. The banks are working night and day. But the amount of [trading] in financial markets is so large that we feel this is the No. 1 financial

problem of the next few months. Banks have to be able to make settlements with each other."

C&P Computer Problems Foul 44,000 D.C. Phones

(Mike McLaughlin excerpted the following article by Elizabeth Tucker, Staff Writer, from Washington Post, Friday, 3 Jan 86, pp D1 & D4. The italics are his, for emphasis.)

Up to 44,000 business and residential phone lines in the District (of Columbia) did not work or worked intermittently (Thursday, 2 Jan 86) because of computer problems at Chesapeake & Potomac Telephone Co. (C&P). C&P... said the ... company had equipment trouble in its central office... between 2:20 and 4 pm. The problem was fixed when (the company) shut off connections to the 44,000 lines for a split second, and then turned the connections back on. C&P has more than 780,000 phone lines in (DC).

(For) nearly two hours... customers often were unable to make or receive calls... The telephone company had not diagnosed the precise cause of the problem late yesterday.... Neither the White House nor the General Services Administration... reported problems... (GWU) Hospital experienced a delay in getting dial tones, but only for about 10 minutes...

...the Associated Press... could receive calls but not make them between 2 and 4 pm.... "You don't know what's going on in terms of news... I thought someone cut the cables. I was worried." (AP spokesman) The Washington Post Co. also experienced problems... One State department official ... "... heard corridor gossip [that people] weren't getting calls in or out." The DC police... reported no problems in receiving 911 emergency calls, and sid there was no appreciable drop off in calls... C&P... said some people may have experienced problems reaching 911... "It could be that no one had problems with 911."...

The problem is not considered usual... "They don't know what caused the problem, but it's up and working fine . . . For all intents and purposes they reset the system, turned off all the connections and then turned them back on again -- *like resetting a computer.* {Emphasis supplied} "They are researching and analyzing the tapes to see what caused the problem."... such problems can occur when heavy calling is taking place ... but that such was not the case (2 Jan 86). "We ruled it out . . . A lot of people aren't working downtown... calling volumes are down dramatically." The telephone system "sometimes can get confused," and think there is heavy calling when there isn't...

Risks using robots in industry

Bill Keefe offered the following article, noting that it brings up many questions as to who bears the responsibility (liability?) to protect people from such occurrences.

In The Lion's Cage" [Forbes Oct. 7, 1985]

On July 21, 1984, at about 1 p.m., a worker at Diecast Corp. in Jackson, Mich. found Harry Allen, 34, a diecast operator pinned between a factory pole and the back of an industrial robot. But Allen's co-worker couldn't come to his aid. Using the robot's controller, the company's director of manufacturing finally unpinned Allen, who was alive but in cardiac arrest. He died in a hospital five days later.

Allen had entered a restricted area, presumably to clean up scrap metal from the floor. While there, he got in the way of the robot's work, and thus became the first - and so far only - U.S. victim of an industrial robot-related accident.

That's not a bad safety record, considering that 17,000 robots are now installed in the U.S. But the bet is he won't be the last. The Japanese, who lead the world in robot installations, also lead in robot-related fatalities: There have been reports of at least 5, and possibly as many as 20, such deaths in Japan.

That's only fatalities. In this country, companies are not required to report injuries related to specific equipment, so no reliable data are available. But in Sweden, a pioneer in the use of industrial robots, one study estimates that there is 1 accident a year for every 45 robots. By 1990, when the number of robots installed in American Industry could climb as high as 90,000, the number of injuries could climb accordingly. That's because robots move quickly and are programmed to go through a series of motions without stopping. A worker who gets in the way can be struck, pushed aside, crushed or pinned to a pole as Allen was.

How will industry minimize the risk to its workers? Probably with difficulty. Robots don't easily accommodate safeguards. Whereas most machinery operates within a fixed set of boundaries, robots have a large "striking distance" - the reach of their mobile arms within three dimensions. In automotive assembly plants, maintenance workers often collide with robots adjacent to the ones they're servicing because they don't realize they are in another robot's work area. A robot may perform one task five times and then start on a completely different activity, and with it a different set of motions. Also, a robot can sit idly for a time and then come to alive again, threatening injury to a worker who mistakenly thought it was shut down.

What's being done to make robots safer? Right now, not much. "The extent of most safety precautions are signs saying, 'Restricted Area: Keep Out,' or maybe a guardrail," says Howard Gadberry of the Midwest Research Institute in Kansas City, Mo. Indeed, the most common safeguards - perimeter barriers such as guardrails and electric interlocked gates, which automatically shut down the robot when opened - don't protect those maintenance workers and programmers who must enter the lion's cage. Presence-sensing devices, such as pressure-sensitive mats and light curtains, both of which automatically cut off a robot's power, also don't seem to offer as much protection as is needed, if only because workers are even more unpredictable in their movements than robots. They may not step on the mat when feeding parts to a robot, or they may not break a light curtain's beam.

That's not to say that robots can't be made safer. Researchers at the Renssalaer Polytechnic Institute, for example, recently completed a research prototype for several large U.S. companies of a four-sensor safety system that continuously monitors the area around a robot. Using ultrasonic, infrared, capacitance and microwave sensors, the RPI system is designed to stop a robot in its tracks if a worker gets too close. Cost? Five thousand dollars in production, according to Jack Meagher, a senior project manager at RPI.

The National Bureau of Standards has also been working with ultrasonic sensors on robot arms similar to the system at RPI. They both have developed a secondary, or watchdog, computer to monitor the actions of the robot and its microprocessor. After all, if the robot's computer goes berserk, how can it monitor itself? That's more important than you might think, 30% of robot accidents seem to be caused by runaways, according to John Moran, director of research at the National Institute for Occupational Safety & Health.

While such systems slowly make the transition form research to the factory floor, industry is trying to put basic safety standards into practice. Recently, the Robotic Industries Association proposed a set of national safety standards for robots that could go into effect as early as next summer.

Would such standards have prevented Harry Allen's death? Maybe not. The robot at the Diecast plant was surrounded by a safety rail with an electric interlocked gate that automatically shut down the robot when the gate was opened. However, there were two gaps in the rail that allowed workers to easily bypass the safeguard; that has since been corrected by the company.

Says Allan Harvie, deputy director of the Michigan Department of Labor's bureau of safety and regulation, "I could only presume Harry Allen thought he could go in and do what he

intended to do without having to shut the robot down."

Jerry Saltzer responded with the following note ("Robots are different").

> When someone gets pinned to the wall by a robot, something different is going on as compared to when someone gets gunned down by an FBI agent operating under incorrect information retrieved from the NCIC. Both cases may lead to specific tragedies, yet the example of risks from robots seems to me to be qualitatively different from many other computer-use risks.
>
> The difference is that robots are used primarily in environments where mechanically-oriented people are accustomed to balancing the risks of new machinery against the benefits. These people have, over the years, learned to deal with risks from gear trains and drive belts, from swinging tailends on steamshovels, from runaway elevators, from inadequately supported cranes. They watch out, they learn from accidents, their insurers offer advice, they make mistakes and take risks, and they learn. To a first approximation, an industrial robot presents a risk similar in kind to other new machinery, and there is a moderately well-working system in place that is accustomed to watching for the risks. If anything, the average mechanic is suspicious of a new piece of machinery in direct proportion to its complexity, newfangledness, and gadgetry level, so is probably expecting the robot to screw up in marvelous ways. One might wish to argue with the particular balance that an industry has struck between risks and benefits, but it is unusual to find one in which mechanical risks are not understood at least moderately well.
>
> The mechanic's suspicion of the new gadget is the essence of what seems to be missing in many other applications of computers, and why it is so important to raise awareness of the need to assess risks. I'm not convinced we need to harass our colleagues in the robot business with risk-consciousness-raising. We should be instead talking to their installers to find out what we can learn.

Some of the Seven Deadly MediCines

Nancy Leveson (University of Calif. Irvine) was just on a panel concerned with Software Safety at an IEEE conference on Computers in Medicine and heard about some more incidents involving software faults. The first was cited already cited in ***SEN*** (10 2, April 1985) (about the programmable implanted pacemaker which was inadvertently reprogrammed by emitted magnetic fields from an anti-theft device in a retail store), which indicated that the patient had survived. Unfortunately, his weakened heart subsequently was unable to stand the increased pace, and he died.

Other recalls by the FDA reported by Nancy involve:

> 1) An infusion-pump (used for insulin) had a software problem which caused the infusion of insulin or dextrose to be delivered at the maximum rather than the lower intended rate. This occurred when certain valid data was entered according to user instructions.
>
> 2) A programmable pacemaker "locked-up" when being reset by an external programming device. Luckily this occurred in a doctor's office, and the doctor was able to revive the patient.
>
> 3) A multiple-patient monitoring system was recalled because the software got patients' names mixed up with the wrong data.
>
> 4) An algorithm was incorrectly programmed in a diagnostic lab instrument which caused certain patient data to be reported erroneously as all zeros.
>
> The reference for these incidents is: H. Bassen, J. Silberberg, F. Houston, W. Knight, C. Christman, and M. Greberman. "Computerized Medical Devices: Usage Trends,

Problems, and Safety Technology," in Proc. 7th Annual Conference of IEEE Engineering in Medicine and Biology Society, Sept. 27-30, 1985, Chicago, Illinois, pp. 180-185.

Police Computer Information -- from David Dyer

The human element really is where the action is, and it is a completely two-edged sword; Human actions which have the power to "fix" something almost inherently also give the power to "break" things equally severely. Conversely, weighty check and balance systems intended to prevent abuse end up preserving the status quo, however good or bad that may be.

The "police computer horror story" I'm most familiar with is illustrative. This is a well documented case I've been reading about in ACLU publications.

It seems some poor soul had his wallet stolen, and some criminal adopted his identity and later was involved in a robbery/murder. Through some circumstances peculiar to the case, the stolen identity, but not the culprit, were known to the LAPD. The detectives working on the case put the stolen identity into a national police computer. Our hero was stopped for a routine traffic citation, the computer coughed his name up, and he ended up on ice for a few days as a murder suspect.

So far, this is pretty harmless and understandable. Eventually the guy's identity and non-involvement were established and he was turned loose. Then it happened again. And Again. The guy began carrying a letter from the local chief of police, saying he wasn't the guy the computer said was wanted, but that didn't cut it when he traveled.

The problem was that the LAPD detectives who put in the original "want" refused to remove it. Eventually the guy (and the ACLU) got the courts to mandate expunging the computer. I think the detectives involved and the LAPD are being sued. Quite rightly.

The point is, it is **hard** to design a system that can do its intended job, permit discovery and correction of errors, and resist unauthorized or inappropriate use. I can't imagine a system that can do all three. [David Dyer]

Activities in Europe -- reported by Udo Voges

I would like to bring some activities to your attention which are going on in Europe, especially within and triggered by EWICS TC 7.

The European Workshop on Industrial Computer Systems (EWICS), TC on Systems Reliability, Safety and Security (TC 7) is working since about 10 years in this area, having some 100 members from industry, research and university. Previous work resulted in Position Papers on

Development of safety related software
Hardware of safe computer systems
Guidelines for verification and validation of safety related software
Guidelines for documentation of safety related computer systems
Techniques for verification and validation of safety related software
System requirements specification for safety related systems

Current working areas include:

System integrity
Software quality assurance and metrics
Design for system safety
Reliability and safety assessment

Besides conducting about four working meetings a year the TC is organizing the IFAC/IFIP Workshop on Achieving Safe Real-Time Computer Systems (Safecomp'79, '82, '83, '85, '86).

The results of the work of TC 7 are introduced into the standardisation process (IEC, ISO, and national bodies) as well as used by companies and licensing authorities.

Those interested in more information can either contact me or the current Chairman of TC 7: Mr. J.M.A. Rata, Electricite de France, 1 Avenue du General de Gaulle, F-92141 CLAMART FRANCE.

There exists an American counterpart to EWICS TC 7, but it was not possible to attract enough interested persons to keep it alive. The Japanese counterpart is also active, but due to the language barrier communication is minimal. [Udo Voges]

Safety Group Activities in the U.S. -- reported by Nancy Leveson

Udo Voges writes about the EWICS TC7, and said that such a group died through lack of interest in the U.S. Actually, there is a similar group which has been active in the U.S. for about three years. It is called the Software System Safety Working Group and was started by the Air Force although it is now a tri-service group. Although sponsored by the DoD, it is not limited to military applications and has participants from other branches of the government and industry. The latest meeting was held in conjunction with a conference on computers in medicine.

Meetings are held approximately twice a year and usually have from 50-200 participants. One of the products of the group is a Software Safety Handbook which is meant to accompany the recent MIL-STD-882b (System Safety) update. The main purpose of the group has been to meet and discuss new techniques, share experiences, exchange ideas, etc. There is tentatively a meeting planned for January in Washington D.C. Anybody interested in the group should contact me (nancy@uci.edu) or Al Friend (friend@nrl-css) who is with the Navy and is currently chairing the group. A future plan is to have an on-line safety database which will reside on-line.

Other activities in which I have been asked to participate and which might be of interest to readers of this forum are a conference on safety which will be held in Washington D.C. next July and a workshop on safety and security sponsored by the Center for Software Reliability in England next September. I am also considering organizing a workshop in California on safety which would be held right before the next International Conference on Software Engineering in Spring 1987. Anyone interested in more information on any of these activities can again contact me and I will direct you to the right people. [Nancy Leveson, University of California, Irvine]

Automobile Computer Control Systems Susceptible to Interference -- Bennett Smith

> By chance I saw an article in an issue of the "Journal of Environmental Engineers" (published in England, date of issue about 10 months ago, I believe) about the sensitivity of a microprocessor-controlled automobile control system to external electromagnetic radiation. As I recall, a CB transmitter near the car could, at the right frequency, make the engine slow down or speed up. Perhaps this article would interest some of your contributors. [Bennett Smith, IRCAM; 31, Rue Saint Merri, 75004, Paris, France]

That led to a message from John Brewer:

> Re: Bennett Smith's comments of emi-rfi susceptibility in automobile control applications... cb's are low power, limited frequency devices. As an Amateur radio operator, one has to be aware of much higher output power, as well as a much wider bandwidths. Amateur Radio frequency allocations include segments from 1.8Mhz to Ghz ranges.
>
> As I remember, some of the control modules are also pretty good emitters of Emi/Rfi hash as well. Typical (legal) output power of a CB is 5 watts or less. A typical ham radio mobile transmitter output power is 100-200 watts.

More on the Failure of Shuttle Launch STS-6

A seemingly reliable source has contributed this post-mortem.

> The JSC Mission Control Center has a pool of 4 IBM 370/168 systems for STS mission operations. During a mission, one system is on-line. One is on hot backup, and can come on line in about 15 seconds. During critical periods such as launch, reentry, orbit changes, or payload deployment, a third is available on about 15 minutes notice. The third 370 can be supporting simulations or software development during a mission as this can be interrupted with no real problem. Prior to STS-6, the 370 software supported only one activity (mission, simulation, or software development) at a time. Later, the Mature Operations Configuration would support 3 ativities at a time. These would be called the Dual and Triple Mission Software deliveries. STS-6 was the first mission after the Dual Mission Software had been installed. At lift-off, the memory allocated to that mission was saturated with the primary processing, and the module that allocated memory would not release the memory allocated to a second mission for Abort Trajectory calculation. Therefore, if the mission [had been] aborted, trajectories would not have been available. After the mission, an error in the software was corrected.

SDI Debates

EXCERPTED FROM Arms-Discussion Digest Tuesday, October 29, 1985 9:33AM Volume 5, Issue 8, author John L. Mills:

> This is a summary of my impressions of the panel discussion/debate entitled "Star Wars: Can the Computing Requirements be Met?", Monday 21 October at MIT. The panelists where Danny Cohen, David L. Parnas, Charles L. Seitz, and Joseph Weizenbaum. The moderator was Michael L. Dertouzos.
>
> I was basically disappointed in this panel discussion. I was hoping to hear a good counter to the the arguments Dr Parnas had put forth in his papers. Dr Cohen started what looked like an organized attack on Dr Parnas' "Octet", refering to the series of eight papers Dr Parnas presented his arguments in. Dr Cohen correctly dismissed the eighth paper, "Is SDIO An Efficient Way To Fund Worthwhile Research", as being outside the bounds of the current discussion. Unfortunately Dr Cohen only further discussed one of the other papers. The other six where dealt with some minor hand waving. I have to admit I don't remember which paper Dr Cohen "went into detail" on. This is because the detail amounted to a one slide outline of the major six points of this paper. This slide was up for no more than one minute with some more hand waving that none of these points were true.
>
> Back to the side claiming the software is not feasible, Dr Weizenbaum didn't really add much of anything to Dr Parnas' arguments. He thought that Dr Parnas had done a wonderful job and there wasn't much he could add. He gracefully didn't take up much time saying this either. Dr Parnas basically presented the material in his papers. He added the new point that even if we build this thing and it "tested OK", we could never really trust it and would be forced not to rely on it.
>
> Charles Seitz made no attempt to directly attack Dr Parnas' argument. He focused his presentation on a simplistic hierarchical structure the software for SDI could take. Unfortunately this looked like a highly centralized form of controlling all the weapons and sensors resulting in a high degree of complexity and size.
>
> Both Dr Cohen and Seitz hit upon the point that the software for SDI is not necessarily as large and complex as some people might think. They claimed that it could be built of smaller fairly independent parts. To me this appeared contradictory to Dr Seitz's hierarchical control structure. It did come through that if you had enough totally

independent platforms shooting things down, you stood a good change of hitting most the targets. It is also clear that you would need a very high level of overkill to make this work since the other platforms don't know who else is shooting at what.

Back to Dr Parnas' points, I did get the feeling that there is some general agreement that there is a limit to the scale and complexity our software engineering can handle. Dr Parnas furthered this point by saying large advances in the mathematics of discrete functions are going to be a major stumbling block in the furthering software engineering. He doesn't expect these large advances on the grounds that if you simplify the equations too much you are losing information. A discrete function can represent only so many bits. I may not have this argument exactly right. He also went thru his standing arguments against AI or automatic programing helping very much.

> [I think the argument is that we need concise, manipulable discrete functions modelling software in order to achieve what other fields of engineering can do with concise, manipulable continuous functions. However, such concise representations may not be possible due to information-theoretic constraints on the number of bits that can be represented by a certain number of symbols. -- MDAY@MIT-XX]

> [I didn't get quite this impression, though I agree with it. Rather, I thought Parnas was saying that the problem was in the fact that with software that is fundamentally digital, there is no such thing as a continuous function, and that therefore the usual engineering assumption valid in most of the world that small changes in input or in correctness necessarily mean small changes in output or result simply isn't valid in the software engineering world. Until it is possible to analyze software in terms of approximately correct functions, graceful software degradation (in terms of an approximately correct program always doing approximately correct things) is not really possible. -- LIN@MIT-MC]

Both sides came up with a number of interesting and colorful analogies. The most relevant is the Space Shuttle. Dr Cohen claims that the Space Shuttle works. This is obviously true in some sense. However, it was also pointed out that there have been times when the software on the shuttle has not worked within seconds of launch. It seems that it would be impractical to ask the Soviets to wait 15 minutes while we reboot the system.

> [Indeed, Seitz conceded that under certain circumstances plausible in the context of a nuclear missile attack, it might be necessary to re-boot the system. He then proceeded to ignore the consequences of that; he did not even say that there are ways to eliminate the need for re-booting. -- LIN@MIT-MC]

In summary it seems that there are very real limits on what our software engineering can handle reliably. We are actually not that far from those limits in some of our current efforts. If SDI is to work its architecture must be dictated by what is doable by the software. It is unclear that SDI is feasably from a material cost point of view if the platforms are small and independent enough to be reliable from the software standpoint.

In closing I would like to say that I don't think either side did a particularly good job sticking to just the software feasibility issue. One other interesting thing happened. Dr Parnas claimed to have asked some person with authority over SDIO whether "No, we can't do this" was an acceptable answer. He did this for the first time at this debate because he did not want to say this behind this person's back. Unfortunately, I don't remember this other person's name, but he was in the audience. Dr Parnas claims that the answer was, "No is not an acceptable answer" and challenged the other person to deny this. The other person promptly stood up and did exactly that.

> [If you mean that it was political, that's certainly true. But politics is really the determinant of the software specifications at the top level. That is how it should be, and people who want to ignore that are ignoring the most important part of the problem. However, in other instances, the Administration has noted that the SDI is central to future US defense policy. In addition, it has never specified what evidence it would consider adequate or sufficient to turn off the SDI. Herb Lin
>
> [John Mills]

Expecting the Unexpected -- Herb Lin

Regarding PGN's comments about spontaneous failure: The Russians have a saying regarding rifles used on stage in plays: once every decade an unloaded gun will fire; once every century a rake will fire. [Herb Lin]

[Perhaps that is what prompted Stravinsky to stage "The Rake's Progress". PGN]

The Titanic Effect

(Source unknown, but contributed by JAN Lee)

> The Titanic Effect: The severity with which a system fails is directly proportional to the intensity of the designer's belief that it cannot.
>
> Corollary: The quantity and quality of built-in redundancy is directly proportional to the degree of concern about failure.

InSIGnifica noted SOFTly

Bill Riddle reminded me of the 18-page *recipe* for fruitcake that the government (yep, read "DoD") put out. Among other things, it specified that the product should be *organoleptically pleasing*, and -- when *bisected vertically* -- should not crumble, smear, etc., etc., etc. They evidently got some multiple tons at the bargain-basement price of only $1.51 per pound. Now ... the **real** question is what was the price per pound for the specification?!!!

As a final note, Ed Joyce sent me a copy of a computer-generated letter addressed to

> Mr. Rudolph E. Hirsch Ass, Dir.
> University of M.D.
> Computer Science Cen
> College Park, MD 20742

that began "Dear Mr. Ass:" [Enough to make a man Comma-Tose.]

NEXT ISSUE. Please SENd me your contributions for the next issue before 21 March 1986. Peter G. Neumann