BUG Zapper

Computer scientist **NANCY LEVESON** aims to keep the skies friendly with glitch-free software. By P. B. Gray

"YOU SHOULD NOT READ ANYTHING INTO THE FACT that I've been taking the train a lot recently," Nancy Leveson writes on her Internet home page. It has nothing to do with the work she and her students did on TCAS II, a collision-avoidance system now required on all commercial aircraft with more than 30 seats flying in the United States. Lately, she adds, her participation in air traffic safety evaluations has caused friends to "make sure their passports are up to date," in case the system she devises sends them to Azerbaijan or Argentina by mistake.

The wisecracks belie the seriousness of what she does. Leveson is one of the nation's leading authorities on safety-critical software, particularly programs embedded in hugely complex machinery like missile-guidance systems and jumbo jets. The National Aeronautics and Space Administration and the Federal Aviation Administration have been major collaborators. For TCAS II, Leveson invented an elegant language much admired by engineers for its ease of use. And she has teamed up with a group of researchers to enhance the nation's air traffic control system to reduce midair collisions, which caused 20 deaths in the U.S. in the first nine months of 1998. One advantage of her work, Leveson notes, is that "nobody questions its goals, except for a few misanthropes who don't matter anyway."

ROBERT ANDREW PARKER

Leveson's academic career has been equally distinguished. After teaching and research stints at the University of California at Irvine and the University of Washington in Seattle, she became a full professor last month in what is arguably the most prestigious engineering enclave in the world, the department of aeronautics and astronautics at the Massachusetts Institute of Technology in Cambridge. MIT brought her in to sharpen its focus on new aviation technologies. "Software is of paramount importance in the future of aviation," says department chair Ed Crawley, "and Nancy Leveson is the authority in this field."

LET NONE OF THIS SUGGEST A lifetime spent plowing the dusty acreage of scholarly texts. Leveson's career has been full of fits and starts, dead ends and second chances, long sabbaticals and lucky breaks. Brusque and prickly, driven by a fierce independence and a yen for adventure, Leveson has cut a professional path to suit her restless intellect and combative spirit. She has danced with tribe members in the most remote corners of New Guinea, hopscotched across the Australian outback on freight trains, walked through the slums of Calcutta, and hiked the foothills of the Himalayas. "I redesign my career every two to four years," says Leveson, who, at 54, looks a little like a mature Mama Cass, with her ample figure, billowing outfits, and a long dark mane of hair flecked with gray. "I don't like to be bored."

No kidding. Ask about her work, and she waves you away. "Just read my book," she says, referring to *Safeware*, a 680-page tome in which she discusses terms like "nondeterminism" and "path robustness." She has even less patience for doomsayers who predict that the Y2K problem—when flummoxed computers crash on New Year's Day, 2000—could lead to Armageddon. Fixing the date problem, she says, is "trivial."

It couldn't seem otherwise to a woman who understands not only arcane software instructions but also the mechanics and hydraulics of the complicated machines they command. In her work with the folks at NASA, to enhance the safety of the space shuttle, Leveson has shown herself to be a code genius—a person who can envision, after looking at hundreds of lines of computer code, what they actually do.

No matter how cleverly constructed, virtually all software contains bugs, and even the most extensive testing program may not ferret out every error, Leveson explains. Glitches are commonly found in the code itself. Considering the millions of lines of instructions involved in complex programs, it's easy for a programmer to key in the wrong number or letter. Flaws can also grow out of the original specifications. During the development of one missile, Leveson claims, software released a silo latch but closed it before the missile could be fired from the plane's wing. Other problems lie, like land mines, deep inside a program, emerging only after months or even years of service. "Software has to be designed with careful attention to all the ways in which the system may be used—or even misused, meaning pushed beyond its limits," she says.

The obvious preventive to mechanical failure is a backup system, a duplicate part or engine that takes over when the first one fails. But software is like a phonograph record; if there's an error in the original, there's an error in the copy. So scientists came up with a more sophisticated approach, called N-version programming, which uses separate teams to design software that performs the same function. Presumably, backups created by different programmers would reduce the risk of error.

In the mid-'80s, Leveson and John Knight, now a professor of computer science at the University of Virginia in Charlottesville, decided to put N-version programming to the test. They gave 27 software programmers the same specifications for a relatively simple program that used radar data to detect enemy missiles. After the programmers completed their task, Leveson and Knight ran each program though a simulation. The programs detected 99 percent of incoming missiles. But in a stunning development, each program erred—under the same circumstances—by failing to pick up a missile that was programmed in a certain way. The implication: Software is often no safer when duplicated.

"Lots of academics questioned the theory of redundancy in software," says Ricky Butler, senior research engineer at NASA's Langley Research Center. "Nancy was the first to get her hands dirty with real data." Safety in software, she has since argued, should grow out of a tough analysis during the design process of every conceivable hazard.

By looking at lines of computer code, Nancy Leveson can envision what they will do.

Leveson received a second vault to national prominence in the early '90s, when she became involved in the Therac-25 cases. In one of the most horrifying series of accidents in modern medical history, a radiation therapy machine mistakenly gave massive overdoses to six cancer patients in the U.S. and Canada. Two died shortly afterward of what appeared to be complications from radiation poisoning. Subsequent investigations by hospital physicists traced the problem to a computer software malfunction. The machine was programmed to shoot high-intensity beam into an internal tumor through a tungsten shield. For external

cancers, the shield was removed, and the machine emitted a much lower dose. But when the Therac-25 operator accidentally selected the wrong mode and then corrected the error with quick keystrokes, the machine delivered a massive dose of radiation without the shield. Leveson was hired as an expert witness in legal cases against the machine's manufacturer. Her detailed accounting of the machine's malfunction and its poor design—including chronic incorrect "error" messages and the lack of mechanical safeguards—helped victims or their heirs win settlements.

NANCY LEVESON'S UPBRINGING didn't cast her in the role of unconventional thinker. The daughter of an accountant and a sportswear designer, she grew up in a comfortable suburb west of Los Angeles and chose the University of California at Los Angeles because it was convenient and affordable. Selecting a major was more problematic. Math came easily to her, but she worried that it was "unfeminine" and that she would be regarded as a "brain." She chose it anyway, only to find out later that it was not the most marketable major. "You can teach, or you can be an actuary. Both of them sounded deadly to me," she says.

After graduation, Leveson joined the flood of classmates pouring into graduate school. At UCLA's School of Management, a new specialty, computer science, intrigued her. Armed with a master's degree, in 1967 she joined IBM's battalion of "systems engineers," the glamourless grunts who installed the massive machines. "It was to be my first and last experience punching the clock in the corporate world," she says. "I hated all the rules. I would often work late into the evening on a client's computer, fixing a problem. But if I walked in at 10 minutes past 9 the next morning, I'd be reprimanded by my supervisor for tardiness."

Two years later, Leveson quit to fulfill a long-standing ambition: travel. With a girlfriend, she flew to Germany, bought a new Volkswagen bus, and joined the flocks of students, hippies, and truth seekers who were roaming Europe, India, and the Far East in the early 1970s. By the time she reached New Zealand, her cash had dwindled, so she took a factory job canning fruit cocktail. "It was supposed to be simple. Two cherries per can, two cans at a time, coming down the assembly line," says Leveson. But, like Lucy Ricardo, she fell behind picking rotten cherries out of the cascading fruit and was soon demoted to the peach line by an irate supervisor. She next hitchhiked across the desolate, snake-ridden Australian outback and trekked through the Asian subcontinent, ending up in Nepal. Travel-weary and weakened by intestinal parasites, she decided to come home.

Following a stint as a high-school math teacher in Glendora, California, Leveson returned to UCLA for a Ph.D. in

She took a job canning **fruit cocktail** but was demoted to peaches.

computer science, which she received in 1980. Shortly after joining the faculty of the University of California at Irvine, she was invited by Hughes Aircraft to study one of its torpedo designs. The giant defense contractor was concerned that the program governing the torpedo's microprocessors could cause it to boomerang and blow up the submarine or ship that had fired it. This problem sparked Leveson's keen interest in software and safety, which would become the focus of her research. In 1993, she became Boeing Professor of Computer Science and Engineering at the University of Washington and moved to MIT five years later.

Absorbed by her work, seminars, consulting activities, and projects—including Safeware Engineering Corp. in Seattle, which she formed with her former graduate students—Leveson spends little time at home in her Cambridge apartment. She sails competitively and relaxes by playing mandolin and piano with friends. She has not opted for marriage or children.

Leveson finds that life in academia suits her. "You have enormous intellectual freedom," she says. "You have your own research projects. You can go as fast or as far as you want." As 1 of 8 female professors in a department of 40, she complains from time to time of the "loneliness" and "isolation" she sometimes feels as a woman in computer science.

The problem will only get worse, she believes, because of the drastic decline in female undergraduates majoring in the field. In the '80s, 35.7 percent of students earning degrees in computer science were women. Today that figure has fallen to 27.5 percent. "Boys are encouraged to use computers at an earlier age, and by the time they get to computer lab in school, they think they know it all," says Leveson. "Girls are intimidated by not knowing the jargon, so they avoid challenging the boys for use of the machines."

Computer-focused though she is, her research is becoming more oriented toward people. Is it reasonable, Leveson asks, to replace humans—pilots, nurses, factory workers—with computers? Perhaps not. She speculates that computers may work better when they merely assist people in controlling dangerous systems. For that reason, she adds, "you can no longer design a system without considering the interaction between the human and the computer." This complex interplay is something she's eminently qualified to study, both because she's a scientist, and, she declares, "because I am a human." W

P. B. GRAY *wrote about former Tropicana executive Ellen Marram in the December/January issue.*