Building GARGANTUAN Software

by Eva Freeman

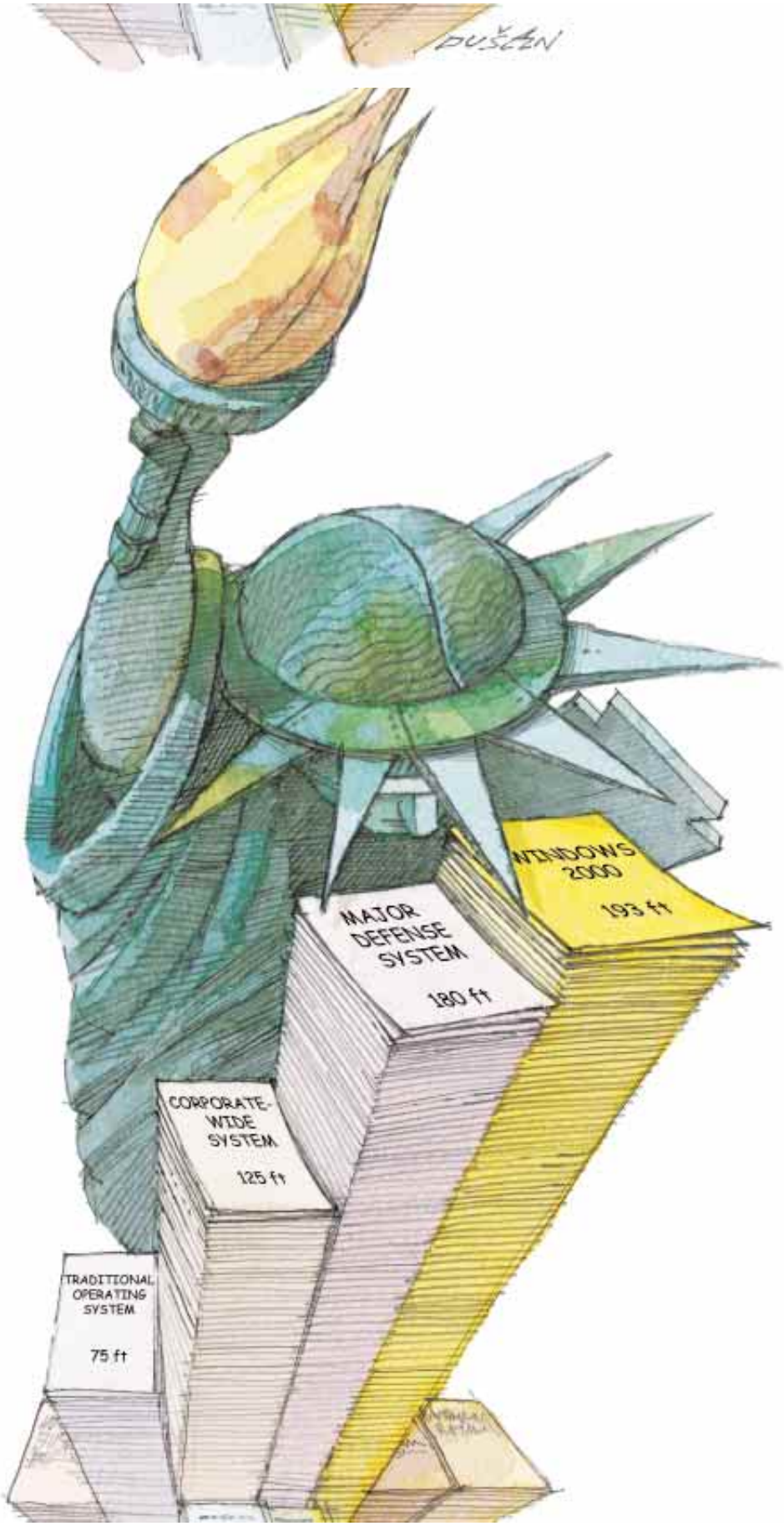
Everything about Windows 2000 is huge, starting with its 29 million lines of code. To tame this monster, Microsoft had to develop a new set of strategies, all while getting more than 4,000 computer geeks to work as a team

Imagine a stack of paper the height of a 19-story building. That's what a printout of Microsoft's Windows 2000 would look like, if anyone cared to print it. With 29 million lines of code written mainly in the C++ computer language, the new operating system (OS) is by far the largest commercial software product ever built. In fact, the development of Windows 2000, and its implementation in a wide range of computer systems and locations, is arguably the most extreme feat of software engineering ever undertaken.

To understand how software could grow to such immensity, think of it not as a monolithic object but as an assemblage of snap-together blocks. There's the core OS, large enough by itself but just one part of the whole that is Windows 2000. Also bundled in are such components as an Internet browser, transaction processing (tools for updating information almost instantaneously as new data are received) and a multitude of drivers, which link peripheral devices such as printers to the OS. The drivers alone account for more than eight million lines of code, with just one of them comprising in excess of a million lines by itself.

So it is conceptually not difficult to comprehend how an operating system with a plethora of features could grow to become a digital behemoth. Less obvious, though, is why Microsoft chose to take on this daunting venture of extreme software engineering and, after deciding to do so, how the company was able to build the product.

Microsoft officials assert that their reason for taking an all-encompassing approach to the design of Windows 2000 is simple: customers asked for it. Company management was well aware that software complexity and bugs grow roughly geometrically with size, but major customers, especially at Fortune 500 corporations, had stated that they needed certain capabilities included in the operating system. The underlying concept is controversial—that it is more efficient for Microsoft to integrate a comprehensive set of subsystems all at once, rather than for each organization on



its own to integrate the particular functions it requires.

It's a trade-off: the benefit is that the OS will perform a breathtaking number of functions; the cost is that the OS becomes very large and potentially slow, unstable and buggy (what critics refer to as "bloatware"). "We knew from the start how hard it would be to build such a functionally rich OS," remembers Brian Valentine, vice president of the Windows OS division at Microsoft. "But our customers were demanding this level of complexity. What we created with Windows 2000 was not so much a new OS as a new view of the role of the OS."

Traditionally, operating systems have handled only a limited set of tasks, for instance, the allocation of resources such as computer memory, depending on whether the OS was designed for personal computers, network management or another specialized application. Windows 2000 takes an alternative approach; it is a single OS that spans most uses, thereby providing uniform security and system services to myriad computers, from individual laptops to clustered servers in corporate data centers. The theoretical advantage is that users will need to learn just one program—albeit a mammoth one—for a wide variety of systems and applications.

Along with a novel way of thinking about operating systems, Microsoft had to invent a different methodology for developing software. Specifically, simulation tools for modeling how the software would work were of limited usefulness. (Unlike other massive engineering projects, the Microsoft venture found scale models essentially worthless.) More important, at the level of size and complexity of Windows 2000, writing code was no longer the central activity. Indeed, testing and debugging have accounted for between 90 and 95 percent of the work.

DUSAN PETRICIC; DATA FROM ARTEMIS MANAGEMENT SYSTEMS

INSOMNIACS' BEDTIME READING: If the code for Windows 2000, the largest commercial program ever written, were printed, the resulting stack of paper would reach past the Statue of Liberty's chin. In comparison, the software for a typical major defense system would be 13 feet shorter.

The greatest challenge in building Windows 2000, however, was not technical. Because every team member possessed so much specialized knowledge, a high level of staff turnover would have devastated the effort, which started three years ago. "My main responsibility is to make sure that the people who joined the project at the start stay with it to the conclusion," Valentine says.

As the individual responsible for managing the entire Windows 2000 team, Valentine has grown to appreciate how crucial the human side is for developing megasoftware: "The difference between extreme engineering in software and other types of extreme engineering is that [with software] the architects are also the builders. Virtually everyone working on this project is highly trained, and no one is expendable or easily replaced. There are no unskilled laborers here, and the most important thing I do is to try to keep everyone on board."

One vital means of keeping the Windows 2000 staff together was to create a sense of family—not an easy job on a project of this size. Consider these numbers: Valentine is ultimately responsible for 4,200 people, including 2,000 Microsoft staff, 800 employees of Microsoft's partners (Intel, for instance) working full-time on the company's Redmond, Wash., campus and 1,400 contract personnel. Another 1,500 Microsoft and contract staff are working on Windows 2000 in other parts of the U.S. and around the world, notably in Israel and India, using the design and test tools on Microsoft's global network to coordinate their efforts with the main campus.

So every Friday afternoon, the entire Windows 2000 team comes together in the company cafeteria, the only room on the Redmond campus that can hold several thousand people. Part weekly report, part pep rally, these meetings are used by Valentine as much to maintain camaraderie as to keep the staff well informed.

Sensing that the anonymity involved in such a massive endeavor was becoming an issue, Valentine brought thousands of markers to one Friday meeting. "I wish each of you could put your signature on the OS, but as the next best thing, let's put our names on the cafeteria," he told them, laughing. By the end of the meeting, the walls were covered with thousands of signatures.

For holidays, Valentine dresses appropriately, as on St. Patrick's Day, when he gave the weekly report while wearing a leprechaun costume. On April Fools' Day, the floors were covered with thousands of Superballs, those toy rubber balls with superhigh bounces. "Brian will do whatever it takes to keep the team together," says Iain McDonald, the Windows 2000 project manager. "I don't think anything embarrasses him, so long as it works." And, of course, each major release of the fledgling software is always an excuse for a huge party.

The week may end on a playful note, but the rest of the time is pure business. Because of the critical importance of testing and debugging, a group of 50 to 60 managers meets at nine in the morning every weekday (as well as on Saturdays and Sundays when a release date approaches) to go over the daily reports of errors found in the Windows 2000 code. These bugs arrive from a variety of sources: independent software vendors from the outside who are developing application software that will run on Windows; select customers at so-called beta sites, who test the software under the actual conditions of usage; Microsoft's internal tests, which involve a large portion of the computer systems at the company; and overseas test sites.

During this "war room" conference, which McDonald usually chairs, each bug's impact is carefully assessed. How much damage will it cause? Will the fix introduce a new problem? Who should take care of it?

The bug is then handed over to the test department, headed by Sanjay Jejurikar, who assigns it to one of 25 triage teams. They log the severity of the bug into a database, then make the necessary fix. After that is done, the revised code is sent to the Build Lab, the center of Windows 2000 testing.

Working in the Build Lab has got to be a hardware geek's idea of heaven. To ensure that Windows 2000 will run successfully on every possible hardware configuration, the multiple rooms of the Build Lab contain at least one of every type of system, storage device, modem card, Internet card and other electronic accoutrement. For video cards alone, as just one example, the computers in the Build Lab host almost 1,200 designs and configurations.

To enable the test group to release an updated version of Windows 2000 every day, Microsoft enforces a strict schedule for submitting revisions to the software. The day's changes—about 250 is a typical number—are checked in between 1 and 4 P.M. After that deadline, the Build Lab begins to enter the changes, and the new release, referred to as the "build," is typically ready between 6 and 8 P.M. This latest version of Windows 2000 is then available for download over the company's internal network. Additionally, by 9 P.M. the Build Lab has pressed and distributed about 2,000 CDs of the software. Before 7:00 the next morning, the build verification test, which evaluates the stability of the previous day's build, is under way.

IN A TYPICAL DAY, WORKERS EXCHANGE ABOUT 90,000 E-MAIL MESSAGES ON THE PROJECT.

ROBBIE McCLARAN *SABA*

BALANCING WORK WITH PLAY: Keeping morale high is a goal of the weekly staff meetings, attended by thousands. Realizing that staff turnover could derail Microsoft's efforts to bring Windows 2000 to market, one company vice president says, "The most important thing I do is to try to keep everyone on board."

About 3,000 individuals at Microsoft use the daily build, locally known as "dog food," as the operating system of their personal computers. Why dog food? Edmund H. Muth, group product manager for the Windows OS division, explains, "Before dog food manufacturers try their latest product in a test market, what do they do? They bring in their own dogs. Their own dogs have usually developed pretty picky habits, and if they don't like the dog food, the manufacturer doesn't test it on someone else's dog. It's the same thing here. We don't send the OS to beta sites until our internal users have said they like it."

Getting to that point has not been easy. The daily test cycle ends around 3:30 P.M., at which time all comments and criticisms are collected for the next day's war room. One benchmark of what extreme testing entails: in a typical day, workers exchange about 90,000 e-mail messages on the project.

Additional tests to stress the software in lifelike conditions are conducted in one- and two-week cycles. Every six weeks those chunks of code that have been thoroughly tested are evaluated one last time and then locked. Valentine explains the underlying theory: "We found that we can only screw up so much in six weeks. Longer than that, and it gets too hard to figure out what's going on." The code, however, is never cast in stone. If a subsequent bug is discovered, Microsoft will fix it, even if that means running additional extensive tests to ensure that the correction will not trigger problems in other parts of the program that have already been frozen.

But not every bug is fixed. "In a software system of this size, you always have to consider the risk that fixing a bug could impact the system somewhere else," Jejurikar, the head of testing, says. According to him, Microsoft always fixes four broad types of bugs: those that cause system crashes, introduce security holes, create Y2K problems or lead to users being denied some type of service. Other kinds of glitches that the company may decide are not worth eradicating include ones that will surface only under unusual conditions, affecting just a small number of customers. Microsoft documents these types of errors and saves possible fixes so that they can be provided to users as needed.

In a perfect world—and with projects to develop simpler software—the idea of intentionally leaving in bugs might seem unthinkable, but Windows 2000 brings home the reality of extreme software engineering. A system of this magnitude cannot be flawless; it can only be tested and documented as thoroughly as time constraints allow.

That said, Microsoft is in the final stage of preparing Windows 2000 for prime time. This last and most massive part of testing is occurring not within Microsoft but at beta sites of the company's key customers and partners, including thousands of firms that manufacture the accompanying computer hardware and complementary software applications. All told, the final test version of Windows 2000 is being poked and prodded in 23 languages and 130 distinct dialects at 300,000 corporate sites located in more than 50 countries.

At press time, Windows 2000 was scheduled for official release in the fourth quarter of 1999, nearly a year late (not uncommon in large software projects). Many financial analysts who follow Microsoft believe the company's future will depend on the success of the product. If that turns out to be true, every bug fixed will have been well worth the effort. SA

About the Author

EVA FREEMAN is a freelance high-technology writer based in Bellevue, Wash. She prefers to use the Macintosh operating system.