The Java Saga

By David Bank
(Wired Magazine)

Sun's Java is the hottest thing on the Web since Netscape. Maybe hotter. But for all the buzz, Java nearly became a business-school case study in how a good product fails. The inside story of bringing Java to the market.

With three minutes to go before the midnight deadline in August 1995, Sun Microsystems engineer Arthur van Hoff took one last look at Java and HotJava, the company's new software for the World Wide Web, and pondered what his colleagues call Arthur's Law: Do it right, or don't do it. Satisfied, the Dutch programming wizard encrypted the files containing the software's source code, moved them to an Internet site, and e-mailed the key to Netscape Communications Corporation, Java's first commercial customer. Five years after the project was launched, Java was done - with a minute to spare.

As he sat at his workstation ready to push the button, van Hoff had good reason to hesitate. Since early versions of the software were released in December 1994, Java has unleashed stratospheric expectations. While today's Web is mostly a static brew - a grand collection of electronically linked brochures - Java holds the promise of caffeinating the Web, supercharging it with interactive games and animation and thousands of application programs nobody's even thought of. At the same time, Java offers Sun and other Microsoft foes renewed hope that Bill Gates's iron grip on the software business can be pried loose. Microsoft rules the desktop, but as networking expands its role, says van Hoff, Java could turn out to be "the DOS of the Internet." Indeed, Sun is rushing to make Java a de facto standard on the burgeoning Web. If Sun succeeds, even Microsoft will have a hard time muscling in.

Software developers are busy shaping Java into applications that will add new life to Web browsers like Netscape and Mosaic, producing programs that combine real-time interactivity with multimedia features that have been available only on CD-ROM. (Java is a programming language, HotJava an "interpreter" installed onto a browser, enabling Java programs delivered over the Web to run on the desktop.) What's a Java application? Point to the Ford Motor website, for instance, and all you'll get are words and pictures of the latest cars and trucks. Using Java, however, Ford's server could relay a small application (called an applet) to a customer's computer.

From there, the client could customize options on an F-series pickup while calculating the monthly tab on various loan rates offered by a finance company or local bank.

Add animation to these applications and the possibilities are endless. Hollywood and Madison Avenue are salivating. "Java allows us to do the things that advertisers and studios are asking us to do," says Karl Jacob, CEO and chief technologist at Dimension X Inc., a San Francisco company creating 3-D websites using Java. "Until now, everything on the Web was fizzling, not sizzling."

Even if Java turns piping hot, how might it lift profits at Sun, which turns out Unix-based workstations and servers for its bread and butter? It's rumored that Netscape paid a paltry US$750,000 to license HotJava (escaping any per-copy charges), a figure that Sun, whose annual revenues will top $6 billion this year, does not dispute. Sun is giving away Java and HotJava free for noncommercial use, in a

fast-track attempt to make them the standard before Microsoft begins shipping a similar product, codenamed Blackbird, in early 1996.

Java is unlikely ever to become a major profit center at Sun, though any increase in Web traffic is bound to increase sales of Sun's workstations and servers. But in this case, emotion may be at least as important as profit. Sun chief Scott McNealy is a fierce competitor, and his blood lust for Bill Gates has fueled the Java project from the beginning. McNealy is especially excited about Java's ability to run on any computer, using Windows, Mac OS, Unix, or any other operating system - posing a threat to Microsoft hegemony. Spinning into the future, McNealy even sees the day when disposable word processors and spreadsheets will be delivered over the Web via Java, priced per use. "This blows up Gates's lock and destroys his model of a shrink-wrapped software that runs only on his platform," effuses McNealy.

Maybe he's dreaming. But Java's progression thus far is a lesson in what can happen when a major company loosens the reins on some of its most precocious talent. The story of Java also highlights the sometimes serendipitous nature of technological development in the face of vague and fast-changing markets.

The origins of Java go back to 1990, when the World Wide Web was barely a glimmer in a British programmer's eye. The personal computer was in its ascendancy, and many inside and outside Sun thought the company had missed major opportunities in the desktop market. Its high-end workstation and server markets were rolling along fine, but as PC use spread across the landscape, the company faced being stranded in a narrowing slice of the computer market. Sun machines had a reputation for being too complicated, too ugly, and too nerdy for mass consumption.

Thus, McNealy was more than ready to listen when a well-regarded 25-year-old programmer with only three years at the company told him he was quitting. Patrick Naughton played on McNealy's ice hockey team. Over beers, Naughton told McNealy that he was quitting to join NeXT Computer Inc., where, he said, "they're doing it right." McNealy paused for a second then shrewdly asked Naughton a favor. "Before you go, write up what you think Sun is doing wrong. Don't just lay out the problem. Give me a solution. Tell me what you would do if you were God."

The following morning, Naughton threw his heart and soul into the challenge.

He typed out a list of Sun's short comings along with his own glowing appraisal of NeXT's critically acclaimed NeXTstep operating system. Twelve screens later, he e-mailed his report to McNealy, who forwarded it to the entire management chain.

A firestorm was ignited. Among Naughton's suggestions: hire an artist to pretty up Sun's uninspired interfaces; pick a single programming tool kit; focus on a single windows technology, not several; and, finally, lay off just about everybody in the existing windows group. (Naughton figured they wouldn't be needed if the previous suggestions were taken.)

Naughton held off NeXT while he awaited the response. The following morning, his e-mail box was bursting. Hundreds of CC'd readers had read his recipe for what ailed Sun and had agreed in a resounding chorus. A typical reaction: "Patrick wrote down everything I say to myself in the morning

but have been afraid to admit." Another voice was that of James Gosling, a remarkable programmer whose opinions carried great weight higher up. Naughton was "brutally right," Gosling e-mailed. "Somewhere along the line, we've lost touch with what it means to produce a quality product."

Naughton joined what one participant called a "bitchfest" attended by a number of high-level engineers. It was John Gage, Sun's science office director, says Naughton, who really dug in, asking, "What is it you really want to do?" The group blue-sky'd until 4:30 the next morning. During those wee hours, they came up with some core principles for a new project: consumers are where it's at; build a small environment created by a small team - small enough to fit around a table at a Chinese restaurant; and make the environment, whatever it may become, include a new generation of machines that are personal and simple to use - computers for normal people.

If still vague, these principles were enough to get Gage's executive juices flowing. With his support, Naughton pitched the high concept to Wayne Rosing, then president of Sun Laboratories Inc. and onetime vice president of engineering at Apple. Naughton laid down key demands he'd scribbled on the back of a restaurant place mat: the project would be located offsite, away from corporate "antibodies" well known for attacking innovative ideas; the project's mission would be kept a secret from all but the top executives at Sun; the software and hardware designs would not have to be compatible with Sun's existing products; and for the first year, the team would be given a million bucks to spend.

Rosing took the idea to McNealy over dinner, and afterward called his assistant by car phone. Via e-mail, the assistant relayed Rosing's pledge to Naughton: "I expect to make 100 percent delivery on what we discussed." Within a day, two of Sun's top dogs, Bill Joy and Andy Bechtolshiem, were throwing in their approvals. That same day, Naughton got his wish. Naughton, Gosling, and Mike Sheridan, who had come to Sun after it bought out his start-up, would be given carte blanche to pursue new projects.

But what projects? At this point the team, codenamed Green, had only a vague notion of what it would do. Competing head-on with Microsoft was out: the Goliath had already won the battle for the mass-market desktop. Instead, the team resolved to bypass Microsoft and the PC market altogether by designing a software system that could run anywhere, even on devices that people did not yet think of as computers. That meant the system had to be compact and simple - the complete opposite of Sun's existing offerings. "We wanted computers to go away, to instead become an everyday thing," Naughton said. "We thought the third wave of computing would be driven by consumer electronics. The hardware would come from Circuit City, and the software would come from Tower Records."

But Green was still a solution in search of a problem. The shared epiphany that set the team's early direction came in the spring of '91 in a hot tub near Lake Tahoe, where Sun's high-level staff had gathered for an annual retreat. Gosling, Sheridan, and Naughton, now joined by Ed Frank, one of Sun's top hardware engineers, soaked and drank beer. Gosling made the observation that computer chips were appearing in toasters, VCRs, and many other household appliances, even in the doorknobs of their Squaw Valley ski-lodge rooms. "That's getting pretty ubiquitous when it's in the bloody doorknob," he said. Yet three remote-control devices were needed just to get a television, a VCR, and a living-room sound system to work. Needless to say, most people still couldn't program any of them. The wonder wasn't that chips were everywhere but that they were being used so badly.

"With a little computer science, all of these things could be made to work together," Gosling insisted. A light switch with a liquid crystal display and a touch pad could play little movies to demonstrate what it controlled and how, he brainstormed. Any control device could do multimedia, and multimedia could help people do real, useful things. There in the hot tub, Gosling recalls, the Green team decided to build a prototype of a device that could control everyday consumer appliances.

Thus began Green's glory days. In April 1991, the team moved from Sun's main campus to office space above a branch of the Bank of America on Menlo Park's Sand Hill Road, cutting itself off from Sun's internal computer network. The programmers disconnected culturally as well. "We thought if we stayed over there, we would end up with just another workstation," Sheridan says. "I was obnoxious about keeping it secret."

They cleared the center of the large room for lab benches and couches, stocked the refrigerator with Dove bars and Cokes, and spent hours playing Nintendo games - the better to understand hypnotically engaging user interfaces. Business issues were put on the back burner to give the technical ideas room to roam. Their mission statement was laid down in a business plan they called Behind the Green Door: "To develop and license an operating environment for consumer devices that enables services and information to be persuasively presented via the emerging digital infrastructure." The key insights into the software that would run such devices came to Gosling at a Doobie Brothers concert at the Shoreline Amphitheater in Mountain View, California. As he sat slouched in front-row seats letting the music wash over him, Gosling looked up at wiring and speakers and semirobotic lights that seemed to dance to the music. "I kept seeing imaginary packets flowing down the wires making everything happen," he recalls. "I'd been thinking a lot about making behavior flow through networks in a fairly narrow way. During the concert, I broke through on a pile of technical issues. I got a deep feeling about how far this could all go: weaving networks and computers into even fine details of everyday life."

Gosling quickly concluded that existing languages weren't up to the job. C++ had become a near-standard for programmers building specialized applications where speed is everything - computer-aided design for instance, where success is measured by the number of polygons generated per second. But C++ wasn't reliable enough for what Gosling had in mind. It was fast, but its interfaces were inconsistent, and programs kept on breaking. However, in consumer electronics, reliability is more important than speed. Software interfaces had to be as dependable as a two-pronged plug fitting into an electrical wall socket. "I came to the conclusion that I needed a new programming language," Gosling says.

As it happened, Gosling, who wrote his first computer language at 14, had been working on a C++ replacement at home. "From the initial 'Oh, fuck' to getting it to a reasonable state" took only a few months, he says. Naughton, meanwhile, had been working on graphic animations, which would serve as the device's interface. By August 1991, Gosling had the graphics running in his new language, which he called Oak (named for the tree outside his office window); this was the progenitor of Java.

By now, the Green team's ambition was to build a device that would work as an interface to cyberspace. Its aim was to create a visual interface to a virtual world. If you wanted to record a TV program while you were away from home, you could control your video by working a virtual video in

the virtual world Naughton was designing. The virtual world - in color and in 3-D - was written in Oak language and spruced up by graphic artists.

All the new programs needed now was a device to run them on. The team wanted a working box, small enough to hold, with batteries included. To build one, the members trotted out what they call "hammer technology"; as Naughton describes it, this involved finding "something that has a real cool 'mumble' (a neat piece of hardware). Then you hit it with a hammer, take the mumble off, and use it. We got a consumer-grade Sharp minitelevision, hit it with a hammer, and got an active-matrix color LCD. We put a resistive touch screen on the front, making sure there'd be no moving parts on the system, no buttons, no power switches, nothing," Naughton explains. The team then wanted to add stereo speakers inside, but couldn't find any to fit the case. "We went to Fry's and bought a dozen Game Boys, played like mad for about three hours, then broke them open - that's where the speakers came from."

The guts of the device were even more remarkable: one of Sun's high-end Sparc workstations stuffed into a dark green aluminum case barely bigger than a softball. Frank's hardware team built three custom chips and designed a motherboard that folded in on itself to save space. The team hacked furiously all through the summer of 1992. "It was a blood bath," Naughton says. "We bit off way more than any seven people should chew. We were arrogant sons of bitches to think we could pull it off. We had so many free variables that we had nothing to come back to. We didn't have anything we knew would work."

The demo was shown to McNealy in August 1992. McNealy saw a hand-held contraption with a small screen and no buttons. When you touched the screen, it turned on. Cool! It opened to a cartoon world - no menus! A character named Duke, a molar-shaped imp with a big red nose, guided the user through the rooms of a cartoon house. You steered with your finger - no mouse! Sliding your finger across the screen, you picked up a virtual TV guide on the sofa, selected a movie, dragged the movie to the cartoon image of a VCR, and programmed the VCR to record the show. This was even more elegant than it first sounded. Everything was done without a keyboard, simply by ripping objects with your finger and dropping them with a "ka-ching" sound.

Sun's boss was ecstatic. Nothing like this smooth, natural interface existed at the time (no Bob, no Magic Cap, no eWorld). And nothing like Oak. A natural cheerleader, McNealy dashed off a hyperactive e-mail. "This is real breakthrough stuff. Don't fail me now. We need to sell this puppy hard, and there's tons of work to make it real. You deliver, it will win.

I can sell this stuff. Charge! Kill H-P, IBM, MSFT, and Apple all at once." Rosing added his own exudations: "Bill Gates is gonna weep."

The prototype was not the only thing the Green team brought to the table. Oak was more than cartoons. Oak was to be an industrial-strength object-oriented language that would work over networks in a very distributed manner. Little packets of code (objects) would scoot around the Net, functioning independently of the devices they were in (computer, telephone, or toaster). Entire applications, like an e-mailer, say, could be built by stitching together objects in a modular way - and the objects wouldn't all have to live in the same place. Plus, Oak dealt with a chief concern of

distributed computing: it encased security, encryption, and authentication procedures into its core so security was essentially invisible to users.

At demos, Naughton would go to the white board to show the scope of Oak, filling the blankness with lines crisscrossing from home computers, to cars, to TVs, to phones, to banks, to - well, to everything. Oak was to be the mother tongue of the network of all digital things.

The Green team had been talking to Mitsubishi Electric about using Oak-based interfaces in cellular phones, televisions, and home and industrial automation systems. France Telecom, which was looking to upgrade its Minitel system, was also interested, Sheridan says. He followed up with a detailed business plan dubbed Beyond the Green Door, proposing that Sun establish a separate subsidiary to push the Oak technology into the consumer marketplace.

Two months after the demo, Sun set up the team as FirstPerson Inc., a wholly owned subsidiary. Rosing moved over from Sun Labs to head the project. Meanwhile, Sheridan, passed over and feeling disrespected, cleaned out his desk and went home. "I thought that having brought it so far, I should be running it," says Sheridan, who now heads a technology consulting group in Virginia. Gosling and Naughton stayed. "There was a changing of the guard, and it wasn't exactly smooth," Gosling remembers.

As FirstPerson grew from 14 employees to more than 60 and moved from its one-room hideout to palatial offices in downtown Palo Alto, things began to drift.

Mitsubishi and France Telecom weren't interested after all, nor was anyone else. Sun figured the cost of the chip, memory, and display device needed for Oak could be squeezed to about $50, but consumer electronics makers were used to paying nominal bucks for chips that made their products easier to use.

In the meantime, the notion of the information superhighway had taken over rational minds and declared itself to be arriving Real Soon Now. Most people understood the I-way to be interactive TV. So when Time Warner circulated proposals in March 1993 to begin interactive television trials in Orlando, FirstPerson jumped at the chance to provide the set-top boxes. As far as Sun was concerned, interactive TV was the technology of the moment, and the company was desperate to use Oak as its entry. Naughton, Gosling, and Joy, who had taken a sporadic interest in Oak, trekked many times to Time Warner's technology center in Denver. There they labored over prototype designs and hashed out cost estimates for a set-top box that would link a television to the information superhighway, using Oak to coordinate a vast complex of images, data, and money securely over a distributed network.

But the deal went to Silicon Graphics Inc., Sun's cross-town rival in the high-end workstation market. Around Sun, McNealy took heat for not pushing as hard with Time Warner as Jim Clark, SGI's founder and then-chair. "All that mattered to Time Warner was who was going to commit to delivering a $300 box on top of the TV," McNealy grouses. "Nobody knows how to do a $300 set-top box that does what they want it spec'd to do." SGI then delivered a box that cost almost 10 times that amount in the summer of '93.

In hindsight, losing the deal was a stroke of good luck, given the overruns, glitches, and weak consumer interest that made the Orlando project a disaster for both Time Warner and SGI. (Clark later left SGI, abandoning interactive TV in favor of the emerging World Wide Web, and founded Netscape - whose browser would later incorporate Java, the successor to Oak. But more of that in a moment.)

A few months later, FirstPerson got tantalizingly close to a deal with 3DO, a company having trouble selling an expensive CD-ROM game machine and thus trying to double the product as a set-top box. It took just 10 days to get Oak running on one of 3DO's game boxes but three months to negotiate a commercial agreement. Finally, the paperwork was ready, and Trip Hawkins, the founder of 3DO, weighed in demanding exclusive rights to the technology. McNealy refused. Given 3DO's precarious state - its expensive game machines had languished on store shelves for two years - that loss turned into a blessing, too.

There were few new avenues left for FirstPerson to try. "What was really dumb for us was focusing on set-top boxes and putting on blinders," Gosling said. "Interactive TV was a mistake. There was so much enthusiasm about it we didn't understand the unreality of that universe."

Those blinders kept Sun - along with many others - from grasping the significance of another emerging phenomenon. In June 1993, Marc Andreessen and Eric Bina at the National Center for Supercomputing Applications at the University of Illinois had released the first version of the Mosaic browser, and the formerly obscure World Wide Web began to take off. But it was at least three more months before Eric Schmidt, Sun's chief technology officer, saw the software for the first time. How had Sun, so long a supplier of well over 50 percent of all host computers to the Internet, missed the Web's mass appeal? "We just took our eye off the ball," Schmidt admits.

FirstPerson was in disarray. Nerves were frayed. Marching orders came down from Sun management: Find something to produce profits. Now! A new business plan was drawn up early in 1994, which unceremoniously dumped the speculative markets FirstPerson had pursued and began focusing on personal computers - the `technology the project was supposed to leapfrog in the first place. The new plan was to create a corps of CD-ROM developers who would write in Oak and, ideally, stick with it as their platform language while moving applications to the commercial online services. Eventually, by the turn of the century, the team thought, broadband networks for interactive TV would finally be ready, and Oak would be established as the programming language of choice. The plan, remarkably, contained no mention of Mosaic or the Web.

For different reasons, the plan met with little enthusiasm among Sun's top executives. They wanted a strategy that would drive demand for Sun's core hardware products, something a program for CD-ROMs would not do. With no profitable strategy in sight, FirstPerson was scrapped in the spring of 1994, the set-top box project shelved, and the interactive TV crew, now renamed Sun Interactive, collapsed into an alliance with Thomson Consumer Electronics to develop scaled-down box and video servers - without Oak for now. It took Bill Joy, who again turned his attention to the project, to rescue Oak.

Joy, a Sun co-founder, is a rare hybrid: a legendary programmer who understands every line of code and carries enormous clout with execs. He is described lovingly by colleagues as a brilliant wild man

whose ideas are at the lunatic fringe. Four years ago, annoyed by Silicon Valley area traffic after the Loma Prieta earthquake and disillusioned about the prospect of doing anything interesting until the Microsoft juggernaut had run its course, Joy established a small Sun research lab at the foot of the ski mountain in Aspen, Colorado. (The town also had a good bookstore, he says, explaining his choice of locale.) Tired of disputes within Sun, he had mostly disengaged, though he dropped in occasionally on the FirstPerson project.

The Web's sudden emergence changed all that. For Joy, the prospect of bringing Oak to the Internet recalled his days at the University of California, Berkeley, nearly two decades earlier, when he'd developed the Berkeley flavor of the Unix operating system out of the original code from Bell Labs and pushed it into widespread use through the Net. That had laid the foundation for Sun. Because of Joy, Sun finally saw that the Internet could become Oak's redemption. Joy's support was critical in what became known as the Internet Play, the "profitless" approach to building market share - a ploy Netscape had made famous by giving away its browser. "There was a point at which I said, 'Just screw it, let's give it away. Let's create a franchise,'" Joy says.

Joy and Schmidt wrote yet another plan for Oak and sent Gosling and Naughton back to work adapting Oak for the Internet. Gosling, whom Joy calls "the world's greatest programmer," worked on the Oak code, while Naughton set out to develop a true "killer app."

In January 1995, Gosling's version of Oak was renamed the more marketable Java. Naughton's killer app was an interpreter for a Web browser, later named HotJava. He wrote the bones of it in a single weekend. Following Joy's dictum, they intended to make it available free on the Web.

But Naughton and Gosling didn't entirely trust Sun to turn Java over to the Internet. Sun has always been a proponent of open standards for software interfaces that allow anyone to build his or her own compatible applications. But this strategy was going further to include the free release of a software implementation. Kim Polese, Java's senior product manager, wrote in big letters on her office white board, "Open means...." and kept adding items to the list. It was one thing for college hackers to release university-grade software like Mosaic for free, or even for a start-up like Netscape to offer its browser for free noncommercial use. It was something else for a major company such as Sun to give away its technological crown jewels, the source code for some of its most valuable technology.

Even Schmidt had doubts about whether he'd be able to live up to his promise to protect the team from the pressure at Sun. "The conversation that never took place, but that I could feel all around me, was, 'Eric, you are violating every principle in the company,'" Schmidt says. "'You are taking our technology and giving it away to Microsoft and every one of our competitors. How are you going to make money?' At the time, I didn't have an answer. I would make something up. I would lie. What I really believed was that Java could create an architectural franchise. The quickest way was through volume and the quickest way to volume was through the Internet."

In December 1994, Java and HotJava (at this stage still called Oak) were posted in a secret file deep in the Net; only a select few were given pointers and invited to check it out. Three months later, Marc Andreessen, who had gone on to start Netscape with Jim Clark, was given a copy. Andreessen gushed to the San Jose Mercury News: "What these guys are doing is undeniably, absolutely new. It's

great stuff." That was how the Java team knew it was going to finally make it. "That quote was a blessing from the god of the Internet," Polese says.

Now that HotJava is being given away, Sun has to make sure it cements Java as a standard - and then figure out how Java can make money. The Netscape deal - incorporating Java into its browser - helps establish a population of Java users. But Sun has to go much further to make it easy enough for anyone other than hard-core nerds to populate the Web with applets.

Sun has promised, but not delivered, tool kits that will allow artists, writers, and other would-be Web authors to speak Java fluently. That's what's needed to increase the supply of enticing applications and to spur users to demand that software suppliers include the technology in their offerings. And Sun has to accelerate that circle into a dizzying spin before competing technologies come along to challenge Java's position. "Sun's window is six to twelve months," says Sheridan. "They need to move quickly because Microsoft will respond in a way that freezes development."

The Java applets are the key. Here's why: for a program to run on a computer, it must first be translated from a language like Basic or C into the machine's native tongue. Because this translation process is incredibly time-consuming, most software comes already translated. But that means different versions have to be created for different computers. Java gets around this problem by using an intermediate language - a sort of Esperanto that is not machine specific but that can quickly be interpreted by any computer.

The result is that small programs - applets - can fly around the Net without regard to what kind of hardware they end up on. If you need to watch an animation that requires a particular fancy doodad to run it, but you don't have that doodad, your machine will pick up the Java-coded applet along with the animation file and run both. Who cares where the software lives? Who cares what kind of machine you have? Who cares about Microsoft?

Microsoft's response will be Blackbird, a package due for initial release in January 1996 that will contain a Web application programming language and an interpreter of its own, at first based on C++ and later on Microsoft's own Visual Basic. Unlike Java, Blackbird's language will work only on the Windows platform at first, but that may not be such a problem given Windows's 80 percent share of the PC market. Anyway, Microsoft is planning a Mac version. And it also claims that Blackbird will be easier to use than professional-programmer-oriented Java.

Sun is pursuing licensing deals as fast as it can to set Java as the standard before Blackbird flutters in. It recently penned an agreement with Toshiba to use Java on a wireless Internet device and claims to have more than 25 potential Java agreements in the pipeline. Though the company says the low-price licensing deal with Netscape is a onetime exception, Sun is explicit about making its technology cheap. The published rates for licensing Java's source code for commercial use include a $125,000 upfront fee plus $2 a copy. "It's priced below our cost," Schmidt says. "This loses money in the licensing business for the foreseeable future. It's a strategic investment in market share."

Another way to avoid that fate would be to license Java to Microsoft and thus complete the penetration of the entire market. Schmidt says he's willing, but Bill Gates hasn't called. Others doubt whether McNealy could bring himself to consort with the enemy even if Gates showed up at the door.

They believe that Sun's desire to beat Microsoft may be even stronger than its desire to see Java succeed. "There are cheaper ways to say 'Fuck you' to Bill Gates," says Naughton, who left Sun last year after he became, as he puts it, "damaged goods" during FirstPerson's nadir. After Sun showed him the indignity of a 2 percent raise, he says, Naughton joined Starwave Corp. in Seattle, where he is using Java to create online services.

Sun is racing to stay ahead of the accelerating wave. The day after that midnight deadline for sending the finished code to Netscape, Joy was already at work pushing the limits of what Java could do. His team was aiming a videocamera at a computer-controlled water fountain in Aspen and putting its image on the Web. The idea was to let anybody with a HotJava interpreter anywhere on the Internet control the spray and interact with kids playing in the water. "I've got 15 patents I could file as soon as I type them," Joy says. "I figure I've got five years. It's like we've got a blank sheet and it says 'Internet.' Normally, the best products don't win. The Internet is an opportunity for the best products to win. Java is great technically and people want it. I'm happy to get that once in my life - or maybe twice."

Often, great technologies are born into the world without one of three essential factors for success: a committed champion, a willing marketplace, and a workable business model. Clearly, Java had its champions - believers like James Gosling, Arthur van Hoff, Bill Joy, Patrick Naughton, and the many who fell by the wayside during Java's long, twisting history. Tweaked, renamed, and repositioned, this time the idea has found a willing marketplace; in time, a distributed object-oriented language like Java will probably establish itself as the foundation of the Net. But whether the standard will be Java depends on whether Sun finds a business model to keep it alive.