CHAPTER 3

info.cern.ch

While it seemed to be uphill work convincing anyone at CERN that global hypertext was exciting, one person was an immediate convert: Robert Cailliau.

Though now the Electronics and Computing for Physics division, by coincidence Robert had in 1980 been in the same Proton Synchotron division as I, and had in fact written the text-formatting program I had used to print the Enquire manual. A Flemish-speaking Belgian, Robert had had the lifelong frustration of people insisting on addressing him in French. After taking an engineering degree at the University of Ghent he picked up a master's at the University of Michigan, an experience that left him with an accent in English that is impossible to identify. Indeed, it became a parlor game for newcomers at CERN to try to guess exactly where he was from.

A dapper dresser who methodically schedules haircuts according to the solstice and equinox, Robert is fastidious in all

things. He is the kind of engineer who can be driven mad by the incompatibility of power plugs. No wonder, then, that he would be attracted to a solution to computer incompatibility, especially coming with a simple user interface. In the marriage of hypertext and the Internet, Robert was best man.

Robert's real gift was enthusiasm, translated into a genius for spreading the gospel. While I sat down to begin to write the Web's code, Robert, whose office was a several-minute walk away, put his energy into making the WWW project happen at CERN. He rewrote a new proposal in terms he felt would have more effect. A CERN veteran since 1973, he lobbied among his wide network of friends throughout the organization. He looked for student helpers, money, machines, and office space.

By the time Mike Sendall approved my purchase of the NeXT machine, I had already gone to the hypertext industry looking for products onto which we could piggyback the Web. At CERN there was a "Buy, don't build" credo about acquiring new technology. There were several commercial hypertext editors, and I thought we could just add some Internet code so the hypertext documents could be sent over the Internet. I thought the companies engaged in the then fringe field of hypertext products would immediately grasp the possibilities of the Web. Unfortunately, their reaction was quite the opposite. "Nope," they said. "Too complicated."

Undaunted, in September 1990 Robert and I went to the European Conference on Hypertext Technology (ECHT) at Versailles to pitch the idea. The conference exhibition was small, but there were a number of products on display, such as a multimedia training manual for repairing a car.

I approached Ian Ritchie and the folks from Owl Ltd., which had a product called Guide. In Peter Brown's original Guide work at the University of Southampton, when a user clicked on a hypertext link, the new document would be inserted right there in place. The version now commercialized by Owl looked astonishingly like what I had envisioned for a Web browser—the program

that would open and display documents, and preferably let people edit them, too. All that was missing was the Internet. *They've already done the difficult bit!* I thought, so I tried to persuade them to add an Internet connection. They were friendly enough, but they, too, were unconvinced.

I got the same response from others at the conference. It seemed that explaining the vision of the Web to people was exceedingly difficult without a Web browser in hand. People had to be able to grasp the Web in full, which meant imagining a whole world populated with Web sites and browsers. They had to sense the abstract information space that the Web could bring into being. It was a lot to ask.

The hypertext community may also have been slightly demoralized. Their small conference was not getting any bigger, and no one was sure where the field was headed. The lack of commercial successes had perhaps left a certain cynicism about bright new ideas that could change the world.

Another possibility I saw was called Dynatext, and was from Electronic Book Technology, a company in Rhode Island started by Andy Van Dam, the Brown University researcher who had coined the term *electronic book*. I thought the company's software could be turned into a Web browser/editor rather easily. However, like many hypertext products at the time, it was built around the idea that a book had to be "compiled" (like a computer program) to convert it from the form in which it was written to a form in which it could be displayed efficiently. Accustomed to this cumbersome multistep process, the EBT people could not take me seriously when I suggested that the original coded language could be sent across the Web and displayed instantly on the screen.

They also insisted on a central link database to ensure that there were no broken links. Their vision was limited to sending text that was fixed and consistent—in this case, whole books. I was looking at a living world of hypertext, in which all the pages would be constantly changing. It was a huge philosophical gap.

Letting go of that need for consistency was a crucial design step that would allow the Web to scale. But it simply wasn't the way things were done.

Despite the "Buy, don't build" credo, I came to the conclusion that I was going to have to create the Web on my own. In October 1990 I began writing code for the Web on my new computer. The NeXT interface was beautiful, smooth, and consistent. It had great flexibility, and other features that would not be seen on PCs till later, such as voice e-mail, and a built-in synthesizer. It also had software to create a hypertext program. Its failure to take over the industry, despite all these advantages, became for me a cautionary tale. NeXT required users to accept all these innovations at once—too much.

My first objective was to write the Web *client*—the program that would allow the creation, browsing, and editing of hypertext pages. It would look basically like a word processor, and the tools on the NeXT's system, called NeXTStep, were ideal for the task. I could create an application, menus, and windows easily, just dragging and dropping them into place with a mouse. The meat of it was creating the actual hypertext window. Here I had some coding to do, but I had a starting place, and soon had a fully functional word processor complete with multiple fonts, paragraph and character formatting, even a spellchecker! No delay of gratification here. Already I could see what the system would look like.

I still had to find a way to turn text into hypertext, though. This required being able to distinguish text that was a link from text that wasn't. I delved into the files that defined the internal workings of the text editor, and happily found a spare thirty-two-bit piece of memory, which the developers of NeXT had graciously left open for future use by tinkerers like me. I was able to use the spare space as a pointer from each span of text to the address for any hypertext link. With this, hypertext was easy. I was then able to rapidly write the code for the Hypertext Transfer Protocol (HTTP), the language computers would use to communicate over the Internet, and the Universal Resource Identifier (URI), the scheme for document addresses.

By mid-November I had a client program—a point-and-click browser/editor—which I just called *WorldWideWeb*. By December it was working with the Hypertext Markup Language (HTML) I had written, which describes how to format pages containing hypertext links. The browser would decode URIs, and let me read, write, or edit Web pages in HTML. It could browse the Web using HTTP, though it could save documents only into the local computer system, not over the Internet.

I also wrote the first Web *server*—the software that holds Web pages on a portion of a computer and allows others to access them. Like the first client, the server actually ran on my desktop NeXT machine. Though the server was formally known as nxoc01.cern.ch (NeXT, Online Controls, 1), I registered an alias for it—"info.cern.ch."—with the CERN computer system folks. That way, the server would not be tied by its address to my NeXT machine; if I ever moved its contents to another machine, all the hypertext links pointing to it could find it. I started the first global hypertext Web page, on the info.cern.ch server, with my own notes, specifications of HTTP, URI, and HTML, and all the project-related information.

At this point Robert bought his own NeXT machine and we reveled in being able to put our ideas into practice: communication through shared hypertext.

At long last I could demonstrate what the Web would look like. But it worked on only one platform, and an uncommon one at that—the NeXT. The HTTP server was also fairly crude. There was a long way to go, and we needed help. Ben Segal, who had a knack for adjusting staffing levels behind the scenes, spotted a young intern named Nicola Pellow. A math student from England, Nicola was working for a colleague in a neighboring building but didn't have enough to do.

A big incentive for putting a document on the Web was that anyone else in the world could find it. But who would bother to install a client if there wasn't exciting information already on the Web? Getting out of this chicken-and-egg situation was the task before us. We wanted to be able to say that if something was on the Web, then anyone could have access to it—not just anyone with a NeXT!

When I gave talks, I showed a diagram with machines of all types connected to the Internet, from mainframes with simple character-oriented terminals through PCs, Macs, and more. To make this possible, I urged Nicola to give the Web the best browser she could, but to assume as little as possible, so this interface could work on any kind of computer. The least common denominator we could assume among all different types of computers was that they all had some sort of keyboard input device, and they all could produce ASCII (plain text) characters. The browser would have to be so basic that it could even work on a paper Teletype. We therefore called it a *line-mode* browser, because Teletype machines and the earliest computer terminals operated by displaying text one line at a time.

Meanwhile, I took one quick step that would demonstrate the concept of the Web as a universal, all-encompassing space. I programmed the browser so it could follow links not only to files on HTTP servers, but also to Internet news articles and newsgroups. These were not transmitted in the Web's HTTP protocol, but in an Internet protocol called FTP (file transfer protocol). With this move, Internet newsgroups and articles were suddenly available as hypertext pages. In one fell swoop, a huge amount of the information that was already on the Internet was available on the Web.

The *WorldWideWeb* browser/editor was working on my machine and Robert's, communicating over the Internet with the info.cern.ch server by Christmas Day 1990.

As significant an event as this was, I wasn't that keyed up about it, only because my wife and I were expecting our first

child, due Christmas Eve. As fate would have it, she waited a few extra days. We drove to the hospital during a New Year's Eve storm and our daughter was born the next day. As amazing as it would be to see the Web develop, it would never compare to seeing the development of our child.

As the new year unfolded, Robert and I encouraged people in the Computing and Networking division to try the system. They didn't seem to see how it would be useful. This created a great tension among us about how to deploy our limited resources. Should we be evangelizing the Web? Should we develop it further on the NeXT? Should we reprogram it for the Mac or the PC or Unix, because even though the NeXT was an efficient machine, few other people had them? After all, what good was a "worldwide" web if there were only a few users? Should we tailor the Web to the high-energy physics community, so they'd have a tool that was theirs and would support it, since CERN was paying our salaries? Or should we generalize the Web and really address the global community, at the risk of being personally disenfranchised by CERN?

Trading in the NeXT for some ordinary computer would have been like trading in a favorite sports car for some truck. More important, the Web was already written for it. If we switched to developing the Web for the much more widely used PC, acceptance might be quicker, but the point was to get people to try what we already had. If we stopped progress and went back to redoing things for the PC, we might never get it done. I decided to stick with the NeXT.

As for the application, my gut told me I had to pursue my larger vision of creating a global system. My head reminded me, however, that to attract resources I also needed a good, visible reason to be doing this at CERN. I was not employed by CERN to create the Web. At any moment some higher-up could have questioned how I was spending my time, and while it was unusual to

stop people at CERN from following their own ideas, my informal project could have been ended. However, it was too soon to try to sell the Web as the ultimate documentation system that would allow all of CERN's documents, within and between projects, to be linked and accessible, especially given the history of so many failed documentation systems. Small but quantifiable steps seemed in order. Our first target, humble beginning that it was, would be the CERN telephone book.

The phone book existed as a database on CERN's aging mainframe. Bernd Pollermann, who maintained it and all sorts of other central information, was charged with somehow providing all this material to each and every user on his or her favorite system. I managed to persuade Bernd that the Web was just what he needed to make life a great deal simpler. If he created a server, I told him, we would get the browsers onto everyone's desktop. He went for it.

I got my simple server to run on the mainframe, then chopped it in two, so that the essential HTTP-related Internet functions were done by my code (written in C language) and Bernd was left to write the rest of the server in his favorite language, "REXX." To make all the documents available, he just had to learn to write HTML, which took him only a few afternoons. Soon the entire world of his search engines, databases, and catalogues was available as hypertext.

That brought us back to the search for a browser. We started porting Nicola's line-mode client onto all sorts of machines, from mainframes through Unix workstations to plain DOS for the PC. These were not great showcases for what the Web should look like, but we established that no matter what machine someone was on, he would have access to the Web. This was a big step, but it was achieved at some sacrifice in that we decided not to take the time to develop the line-mode browser as an editor. Simply being able to read documents was good enough to bootstrap the process. It justified Bernd's time in getting his servers up. But

it left people thinking of the Web as a medium in which a few published and most browsed. My vision was a system in which sharing what you knew or thought should be as easy as learning what someone else knew.

Mundane as it was, this first presentation of the Web was, in a curious way, a killer application. Many people had workstations, with one window permanently logged on to the mainframe just to be able look up phone numbers. We showed our new system around CERN and people accepted it, though most of them didn't understand why a simple ad hoc program for getting phone numbers wouldn't have done just as well.

Of course, we didn't want our brainchild with all its tremendous potential to be locked in at this rather pedestrian level. To broaden the Web's horizons, I set about giving talks and conducting demonstrations. So that people could see something "out there on the Web" other than the phone book, and to practice what we preached, Robert and I continued to document the project in hypertext on info.cern.ch.

What we had accomplished so far was based on a few key principles learned through hard experience. The idea of universality was key: The basic revelation was that one information space could include them all, giving huge power and consistency. Many of the technical decisions arose from that. The need to encode the name or address of every information object in one URI string was apparent. The need to make all documents in some way "equal" was also essential. The system should not constrain the user; a person should be able to link with equal ease to any document wherever it happened to be stored.

This was a greater revelation than it seemed, because hypertext systems had been limited works. They existed as databases on a floppy disk or a CD-ROM, with internal links between their files. For the Web, the external link is what would allow it to actually become "worldwide." The important design element would be to ensure that when two groups had started to use the

Web completely independently at different institutions, a person in one group could create a link to a document from the other with only a small incremental effort, and without having to merge the two document databases or even have access to the other system. If everyone on the Web could do this, then a single hypertext link could lead to an enormous, unbounded world.

CHAPTER 4

Protocols

Simple Rules for Global Systems

Incompatibility between computers had always been a huge pain in everyone's side, at CERN and anywhere else where they were used. CERN had all these big computers from different manufacturers, and various personal computers, too. The real world of high-energy physics was one of incompatible networks, disk formats, data formats, and character-encoding schemes, which made any attempt to transfer information between computers generally impossible. The computers simply could not communicate with each other. The Web's existence would mark the end of an era of frustration.

As if that weren't advantage enough, the Web would also provide a powerful management tool. If people's ideas, interactions, and work patterns could be tracked by using the Web, then computer analysis could help us see patterns in our work, and facilitate our working together through the typical problems that beset any large organization.

One of the beautiful things about physics is its ongoing quest to find simple rules that describe the behavior of very small, simple objects. Once found, these rules can often be scaled up to

describe the behavior of monumental systems in the real world. For example, by understanding how two molecules of a gas interact when they collide, scientists using suitable mathematics can deduce how billions of billions of gas molecules—say, the earth's atmosphere—will change. This allows them to analyze global weather patterns, and thus predict the weather. If the rules governing hypertext links between servers and browsers stayed simple, then our web of a few documents could grow to a global web.

The art was to define the few basic, common rules of "protocol" that would allow one computer to talk to another, in such a way that when all computers everywhere did it, the system would thrive, not break down. For the Web, those elements were, in decreasing order of importance, universal resource identifiers (URIs), the Hypertext Transfer Protocol (HTTP), and the Hypertext Markup Language (HTML).

What was often difficult for people to understand about the design was that there was nothing else beyond URIs, HTTP, and HTML. There was no central computer "controlling" the Web, no single network on which these protocols worked, not even an organization anywhere that "ran" the Web. The Web was not a physical "thing" that existed in a certain "place." It was a "space" in which information could exist.

I told people that the Web was like a market economy. In a market economy, anybody can trade with anybody, and they don't have to go to a market square to do it. What they do need, however, are a few practices everyone has to agree to, such as the currency used for trade, and the rules of fair trading. The equivalent of rules for fair trading, on the Web, are the rules about what a URI means as an address, and the language the computers use—HTTP—whose rules define things like which one speaks first, and how they speak in turn. When two computers agree they can talk, they then have to find a common way to represent their data so they can share it. If they use the same software for

documents or graphics, they can share directly. If not, they can both translate to HTML.

The fundamental principle behind the Web was that once someone somewhere made available a document, database, graphic, sound, video, or screen at some stage in an interactive dialogue, it should be accessible (subject to authorization, of course) by anyone, with any type of computer, in any country. And it should be possible to make a reference—a link—to that thing, so that others could find it. This was a philosophical change from the approach of previous computer systems. People were used to going to find information, but they rarely made references to other computers, and when they did they typically had to quote a long and complex series of instructions to get it. Furthermore, for global hypertext, people had to move from thinking about instructions to thinking in terms of a simple identifier string—a URI—that contained all the essential details in a compact way.

Getting people to put data on the Web often was a question of getting them to change perspective, from thinking of the user's access to it not as interaction with, say, an online library system, but as navigation though a set of virtual pages in some abstract space. In this concept, users could bookmark any place and return to it, and could make links into any place from another document. This would give a feeling of persistence, of an ongoing existence, to each page. It would also allow people to use the mental machinery they naturally have for remembering places and routes. By being able to reference anything with equal ease, the Web could also represent associations between things that might seem unrelated but for some reason did actually share a relationship. This is something the brain can do easily, spontaneously. If a visitor came to my office at CERN, and I had a fresh cutting of lilac in the corner exuding its wonderful, pungent scent, his brain would register a strong association between the office and lilac. He might walk by a lilac bush a day later in a

park and suddenly be reminded of my office. A single click: lilac . . . office.

The research community had used links between paper documents for ages: Tables of contents, indexes, bibliographies, and reference sections are hypertext links. On the Web, however, research ideas in hypertext links can be followed up in seconds, rather than weeks of making phone calls and waiting for deliveries in the mail. And suddenly, scientists could escape from the sequential organization of each paper and bibliography, to pick and choose a path of references that served their own interest.

But the Web was to be much more than a tool for scientists. For an international hypertext system to be worthwhile, of course, many people would have to post information. The physicist would not find much on quarks, nor the art student on Van Gogh, if many people and organizations did not make their information available in the first place. Not only that, but much information—from phone numbers to current ideas and today's menu—is constantly changing, and is only as good as it is up-to-date. That meant that anyone (authorized) should be able to publish and correct information, and anyone (authorized) should be able to read it. There could be no central control. To publish information, it would be put on any server, a computer that shared its resources with other computers, and the person operating it defined who could contribute, modify, and access material on it. Information was read, written, or edited by a client, a computer program, such as a browser/editor, that asked for access to a server.

Several protocols already existed for transferring data over the Internet, notably NNTP for Network News and FTP for files. But these did not do the negotiating I needed, among other things. I therefore defined HTTP, a protocol simple enough to be able to get a Web page fast enough for hypertext browsing. The target was a fetch of about one-tenth of a second, so there was no time for a conversation. It had to be "Get this document," and "Here it is!"

Of course if I had insisted everyone use HTTP, this would also have been against the principle of minimal constraint. If the Web were to be universal, it should be as unconstraining as possible. Unlike the NeXT computer, the Web would come as a set of ideas that could be adopted individually in combination with existing or future parts. Though HTTP was going to be faster, who was I to say that people should give up the huge archives of data accessible from FTP servers?

The key to resolving this was the design of the URI. It is the most fundamental innovation of the Web, because it is the one specification that every Web program, client or server, anywhere uses when any link is followed. Once a document had a URI, it could be posted on a server and found by a browser.

Hidden behind a highlighted word that denotes a hypertext link is the destination document's URI, which tells the browser where to go to find the document. A URI address has distinct parts, a bit like the five-digit zip code used by the U.S. postal system. The first three numbers in a zip code designate a certain geographic region—a town, or part of a city or county. The next two numbers define a very specific part of that region—say, a few square blocks in a city. This gets the mail to a local post office. Carriers from there use the street name or box number to finish the routing.

Slashes are used in a URI address to delineate its parts. The first few letters in the URI tells the browser which protocol to use to look up the document, whether HTTP or FTP or one of a small set of others. In the address http://www.foobar.com/doc1, the *www.foobar.com* identifies the actual computer server where these documents exist. The *doc1* is a specific document on the www.foobar.com server (there might be hundreds, each with a different name after the single slash). The letters before the double slash signify the communications protocol this server uses.

The big difference between the URI and postal schemes, however, is that while there is some big table somewhere of all

zip codes, the last part of the URI means whatever the given server wants it to mean. It doesn't have to be a file name. It can be a table name or an account name or the coordinates of a map or whatever. The client never tries to figure out what it means: It just asks for it. This important fact enabled a huge diversity of types of information systems to exist on the Web. And it allowed the Web to immediately pick up all the NNTP and FTP content from the Internet.

At the same time that I was developing the Web, several other Internet-based information systems were surfacing. Brewster Kahle at Thinking Machines had architected their latest powerful parallel processor. Now he saw a market for the big machines as search engines and designed the Wide Area Information Servers (WAIS) protocol to access them to form a system like the Web but without links—only search engines.

Clifford Newman at the Information Sciences Institute proposed his *Prospero* distributed file system as an Internet-based information system, and Mark McCahill and colleagues at the University of Minnesota were developing a campus-wide information system called *gopher*, named for the university's mascot. To emphasize that all information systems could be incorporated into the Web, I defined two new URI prefixes that could appear before the double slash—"gopher:" and "wais:"—that would give access to those spaces. Both systems took off much more quickly than the Web and I was quite concerned at the time that they would suffocate it.

HTTP had a feature called *format negotiation* that allowed a client to say what sorts of data format it could handle, and allow the server to return a document in any one of them. I expected all kinds of data formats to exist on the Web. I also felt there had to be one common, basic lingua franca that any computer would be required to understand. This was to be a simple hypertext language that would be able to provide basic hypertext navigation, menus, and simple documentation such as help files, the minutes

of meetings, and mail messages—in short, 95 percent of daily life for most people. Hence HTML, the Hypertext Markup Language.

I expected HTML to be the basic warp and weft of the Web, but documents of all types—video, computer-aided design, sound, animation, and executable programs—to be the colored threads that would contain much of the content. It would turn out that HTML would become amazingly popular for the content as well.

HTML is a simple way to represent hypertext. Once the URI of a document tells a browser to talk HTTP to the server, then client and server have to agree on the format of the data they will share, so that it can be broken into packets both will understand. If they both knew WordPerfect files, for example, they could swap WordPerfect documents directly. If not, they could both try to translate to HTML as a default and send documents that way. There were some basic design rules that guided HTML, and some pragmatic, even political, choices. A philosophical rule was that HTML should convey the structure of a hypertext document, but not details of its presentation. This was the only way to get it to display reasonably on any of a very wide variety of different screens and sizes of paper. Since I knew it would be difficult to encourage the whole world to use a new global information system, I wanted to bring on board every group I could. There was a family of markup languages, the standard generalized markup language (SGML), already preferred by some of the world's documentation community and at the time considered the only potential document standard among the hypertext community. I developed HTML to look like a member of that family.

Designing HTML to be based on SGML highlighted one of the themes of the development of the Web: the constant interplay between the diplomatically astute decision and the technically clean thing to do. SGML used a simple system for denoting instructions, or "tags," which was to put a word between angle brackets (such as *<h1>* to denote the main heading of a page), yet it also had many obscure and strange features that were not well

understood. Nonetheless, at the time, the Web needed support and understanding from every community that could become involved, and in many ways the SGML community provided valuable input.

SGML was a diplomatic choice at CERN as well. SGML was being used on CERN's IBM machines with a particular set of tags that were enclosed in angle brackets, so HTML used the same tags wherever possible. I did clean up the language a certain amount, but it was still recognizable. I chose this direction so that when a CERN employee saw the angle brackets of HTML, he or she would feel, *Yes, I can do that.* In fact, HTML was even easier to use than CERN's version of SGML. The people promoting the SGML system at CERN could possibly be powerful figures in the choice of CERN's future directions and I wanted them to feel happy about the Web.

I never intended HTML source code (the stuff with the angle brackets) to be seen by users. A browser/editor would let a user simply view or edit the language of a page of hypertext, as if he were using a word processor. The idea of asking people to write the angle brackets by hand was to me, and I assumed to many, as unacceptable as asking one to prepare a Microsoft Word document by writing out its binary coded format. But the human readability of HTML was an unexpected boon. To my surprise, people quickly became familiar with the tags and started writing their own HTML documents directly.

As the technical pieces slowly fell into place, Robert and I were still faced with a number of political issues that gave us more than a twinge of anxiety. First of all, the Web was still not a formal project. At any moment some manager of the Computing and Networking division could have asked me to stop the work, as it wasn't part of any project, and it could have been considered inappropriate for CERN.

For eight months Robert, Nicola, and I refined the basic pieces of the Web and tried to promote what we were creating.

We drafted a work plan for the Electronics and Computing for Physics division, where Robert was, to try to get funding from them, but no one responded. Accordingly, while developing the technology and trying to promote it to our colleagues, we still had to maintain a somewhat low profile.

The other problem we faced was simply the climate at CERN. There was a constant background of people promoting ideas for new software systems. There was competition among systems created within the experiment groups themselves—software for running a physics experiment, but also for everything from handling electronic mail and organizing documents to running the Coke machine. There was competition over which network to use, among them DECnet, the Internet, and whatever home-brewed thing could be justified. With so many creative engineers and physicists in one place, innovations were constant. At the same time, CERN obviously couldn't tolerate everybody creating unique software for every function.

Robert and I had to distinguish our idea as novel, and one that would allow CERN to leap forward. Rather than parade in with our new system for cosmic sharing of information, we decided to try to persuade people that we were offering them a way to extend their existing documentation system. This was a concrete and potentially promising notion. We could later get them to sign on to the dream of global hypertext. Our argument was that everyone could continue to store data in any form they like, and manage it any way they like. The Web would simply help people send and access information between each other, regardless of the operating system or formats their computers use. The only thing they'd have to do was follow the same simple URI addressing scheme. They didn't "have to" use HTTP or HTML, but those tools were there if they ran into an incompatibility problem.

As we made these points, we also noted that using HTML was easy, since it was so much like SGML. I may have promoted

this angle too much, however. Although SGML had been adopted as a standard by the ISO, it was not well defined as a computer language. I also got a strong push back from many people who insisted that it would be too slow. I had to explain that the only reason SGML was slow was the way it had historically been implemented. Still, I often had to demonstrate the World Wide Web program reading an HTML file and putting it on the screen in a fraction of a second before people were convinced.

Some people were intrigued, but many never accepted my argument. Rather than enter into useless debate, I simply forged ahead with HTML and showed the Web as much as possible. Robert and I held a few colloquia open to anyone in our divisions. We also told people about it at coffee. Occasionally, a group of people getting ready to do an experiment would call to say they were discussing their documentation system, and ask if I could come over and give them my thoughts about it. I'd meet a group of maybe twenty and show them the Web, and perhaps they wouldn't use it then, but the next time through they'd know about it and a new server would quietly come into being.

Meanwhile, Robert and I kept putting information on the info.cern.ch server, constantly upgrading the basic guide to newcomers on how to get onto the Web, with specifications and pointers to available software.

I continued to try to get other organizations to turn their hypertext systems into Web clients. I found out about a powerful SGML tool called Grif, developed by a research group at the French lab INRIA, which ran on Unix machines and PCs. A company by the same name, Grif, had since been spun off in nearby Grenoble, and I was hopeful its leaders would entertain the idea of developing a Web browser that could also edit. They had a beautiful and sophisticated hypertext editor; it would do graphics, it would do text in multiple fonts, it would display the SGML structure and the formatted document in two separate windows, and allow changes to be made in either. It was a perfect match.

The only thing missing was that it didn't run on the Internet. Same story.

I tried to persuade the people at Grif to add the software needed for sending and receiving files over the Internet, so their editor could become a Web browser, too. I told them I would give them the software outright; they would just have to hook it in. But they said the only way they would do that was if we could get the European Commission to fund the development. They didn't want to risk taking the time. I was extremely frustrated. There was a growing group of people who were excited about the possibilities of the World Wide Web, and here we had the technology for a true hypertext browser/editor mostly developed, and we couldn't bridge the gap. Getting Commission funding would have put eighteen months into the loop immediately. This mindset, I thought, was disappointingly different from the more American entrepreneurial attitude of developing something in the garage for fun and worrying about funding it when it worked!

In March 1991, I released the *WorldWideWeb* program to a limited number of CERN people who had NeXT computers. This would at least allow them to write their own hypertext and make the Web information that Robert and I were putting on info.cern.ch available to them.

Word spread within the high-energy physics community, furthered by the cross-pollinating influence of travel. In May 1991 Paul Kunz arrived for a visit from the Stanford Linear Accelerator (SLAC) in Palo Alto. Like me, he was an early NeXT enthusiast, and he had come to CERN to work on some common NeXT programs. Since he had the right computer, he was in a position to use the Web directly, and he loved it.

When Paul returned to SLAC he shared the Web with Louise Addis, the librarian who oversaw all the material produced by SLAC. She saw it as a godsend for their rather sophisticated but mainframe-bound library system, and a way to make SLAC's substantial internal catalogue of online documents available to

physicists worldwide. Louise persuaded a colleague who developed tools for her to write the appropriate program, and under Louise's encouragement SLAC started the first Web server outside of CERN.

Seeing that the high-energy physics people at SLAC were so enthusiastic about the Web, we got more aggressive about promoting it within CERN. In May, Mike Sendall got us an appearance before the C5 committee, which was continually looking at computing and communications, to explain how useful the Web could be, so management would continue to justify the work. Robert and I wrote a paper, too, "Hypertext at CERN," which tried to demonstrate the importance of what we were doing.

What we hoped for was that someone would say, "Wow! This is going to be the cornerstone of high-energy physics communications! It will bind the entire community together in the next ten years. Here are four programmers to work on the project and here's your liaison with Management Information Systems. Anything else you need, you just tell us." But it didn't happen.

In June we held talks and demonstrations within CERN, and wrote about the Web in the CERN newsletter. Because I still had no more staff, it was taking longer than I had hoped to get the functionality of the NeXT version onto PCs and Macs and Unix machines.

I was still hoping that by spreading the word we could attract the attention of more programmers. Since those programmers were unlikely to be high-energy physicists, in August I released three things—the *WorldWideWeb* for NeXT, the line-mode browser, and the basic server for any machine—outside CERN by making them all available on the Internet. I posted a notice on several Internet newsgroups, chief among them alt.hypertext, which was for hypertext enthusiasts. Unfortunately, there was still not much a user could see unless he had a NeXT.

Putting the Web out on alt.hypertext was a watershed event. It exposed the Web to a very critical academic community. I

began to get e-mail from people who tried to install the software. They would give me bug reports, and "wouldn't it be nice if . . ." reports. And there would be the occasional "Hey, I've just set up a server, and it's dead cool. Here's the address."

With each new message I would enter in info.cern.ch a hypertext link to the new web site, so others visiting the CERN site could link to that address as well. From then on, interested people on the Internet provided the feedback, stimulation, ideas, source-code contributions, and moral support that would have been hard to find locally. The people of the Internet built the Web, in true grassroots fashion.

For several months it was mainly the hypertext community that was picking up the Web, and the NeXT community because they were interested in software that worked on the platform. As time went on, enough online people agreed there should be a newsgroup to share information about the Web, so we started one named comp.infosystems.www. Unlike alt.hypertext, this was a mainstream newsgroup, created after a global vote of approval.

Another small but effective step to increase the Web's exposure was taken when I opened a public telnet server on info.cern.ch. Telnet was an existing protocol, also running over the Internet, that allowed someone using one computer to open up an interactive command-line session on another computer. Anyone who used a telnet program to log into info.cern.ch would be connected directly to the line-mode browser. This approach had the disadvantage that the user would see the Web as a text-only read-only system. But it opened the Web to millions of people who could not install a Web browser on their own machine. It meant that someone putting up a Web server could say to "telnet to info.cern.ch then type 'go www.foobar.com,'" which was a whole lot easier than requiring them to install a Web browser. The initial home page seen by users of this public service would include links to instructions for downloading their own browser. Years later we would have to close down the service, since the

machine couldn't support the load, but by then it would have done its job.

The most valuable thing happening was that people who saw the Web, and realized the sense of unbound opportunity, began installing the server and posting information. Then they added links to related sites that they found were complementary, or simply interesting. The Web began to be picked up by people around the world. The messages from systems managers began to stream in: "Hey, I thought you'd be interested. I just put up a Web server."

Nicola had to leave the effort in August 1991, since her internship ended and she had to return to college. True to form, Ben Segal found yet another gem to replace her. Jean-François Groff was full of enthusiasm for the whole idea of the Web, and for NeXT. He came to CERN from France through a "cooperant" program that allowed the brightest young people, instead of spending a year in military service, to work for eighteen months at a foreign organization as a volunteer.

By this time we had reached another awkward decision point about the code. Much of the code on the NeXT was in the language objective-C. I wanted people to use it widely, but objective-C compilers were rare. The common language for portable code was still C, so if I wanted to make it possible for more people around the Internet to develop Web software, it made sense to convert to C. Should I now, in the interest of practical expediency, convert all my objective-C code back into the less powerful C, or should I keep to the most powerful development platform I had?

The deciding factor was that Nicola's line-mode browser was written in C. I decided to make the sacrifice and, while keeping the object-oriented style of my design, downgraded all the common code that I could export from *WorldWideWeb* on the NeXT into the more common C language.

This was a pile of work, but it opened up new possibilities and also allowed a certain cleaning up as I went along. Jean-François arrived at just the right time. For weeks we sat back-to-back in my office spewing out code, negotiating the interfaces between each other's modules in remarks over our shoulders.

"Can you give me a method to find the last element?"

"Okay. Call it 'lastElement'?"

"Fine. Parameters?"

"List, element type. You got it."

"Thanks!"

We rolled out the Web-specific code and also had to duplicate some of the tools from the NeXTStep tool kit. The result, since a collection of bits of code for general use is called a library, we called "libwww."

Unfortunately, CERN's policy with cooperants like Jean-François was that they had to leave when their time was up. They saw a danger in the staff abusing the program as a recruitment stream, and forbade the employment of any of these people in any way in the future. When Jean-François came to the end of his term, we tried everything we could to allow him to continue to work on the Web, but it was quite impossible. He left and started a company in Geneva, infodesign.ch, probably the very first Web design consultancy.

Meanwhile, I had begun to keep logs of the number of times pages on the first Web server, info.cern.ch at CERN, were accessed. In July and August 1991 there were from ten to one hundred "hits" (pages viewed) a day.

This was slow progress, but encouraging. I've compared the effort to launch the Web with that required to launch a bobsled: Everyone has to push hard for a seemingly long time, but sooner or later the sled is off on its own momentum and everyone jumps in.

In October we installed "gateways" to two popular Internet services. A gateway was a little program, like that opening up Bernd's mainframe server, that made another world available as

part of the Web. One gateway went to the online help system for Digital's VAX/VMS operating system. Another was to Brewster Khale's WAIS for databases. This was all done to add incentive for any particular individual to install a browser. VMS targeted the physics community, and WAIS the Internet community. I also started an online mailing list, www-talk@info.cern.ch, for technical discussions as a forum for the growing community.

Always trying to balance the effort we put into getting involvement from different groups, Robert and I decided we now had to promote the Web hard within the hypertext community. A big conference, Hypertext '91, was coming up in December in San Antonio. Most of the important people in the field would be there, including Doug Engelbart, who had created the mouse and a collaborative hypertext system way back in the 1960s. Though it was difficult to find the time, we cobbled together a paper for it, but didn't do a very good job. It was rejected—in part because it wasn't finished, and didn't make enough references to work in the field. At least one of the reviewers, too, felt that the proposed system violated the architectural principles that hypertext systems had worked on up till then.

We were able to convince the conference planners to let us set up a demonstration, however. Robert and I flew to San Antonio with my NeXT computer and a modem. We couldn't get direct Internet access in the hotel. In fact, the hypertext community was so separated from the Internet community that we couldn't get any kind of connectivity at all. How could we demonstrate the Web if we couldn't dial up info.cern.ch? Robert found a way. He persuaded the hotel manager to string a phone line into the hall alongside the main meeting room. That would allow us to hook up the modem. Now we needed Internet access. During our cab ride from the airport, Robert had asked the driver what the nearest university was and found out that it was the University of Texas in San Antonio. So Robert called the school and found some people who understood about the Internet and maybe the

Web, and they agreed to let us use their dial-in service so we could call the computer back at CERN.

The next challenge was to get the Swiss modem we had brought to work with the American electrical system. We bought a power adapter that would take 110 volts (rather than the Swiss 220 volts). Of course it didn't have the right little plug to connect to the modem. We had to take the modem apart, borrow a soldering gun from the hotel (Robert was rightly proud of this feat!), and wire it up directly. Robert got everything connected, and it worked.

We didn't have real Internet connectivity, just a dial-in Unix login, so we could show only the graphic World Wide Web program working on local data. Nonetheless, we could demonstrate the line-mode browser working live. We were the only people at the entire conference doing any kind of connectivity. The wall of the demo room held project titles above each booth, and only one of them had any reference to the World Wide Web—ours.

At the same conference two years later, on the equivalent wall, every project on display would have something to do with the Web.