manager of JPL's Office of Computing and Information Systems. In future years he spent several weeks there almost every summer as a consultant. He was honored for his contributions to JPL by being appointed distinguished visiting scientist in 1990.

Aaron was recruited to go to the University of Michigan in 1978 as the director of the Computing Center and remained in that position until 1986. He also served as professor of electrical engineering and computer science from his arrival until his retirement in 1990, when he became professor emeritus and director emeritus (Computing Center). He thereafter continued to teach a beginning course each year. He spent winters in Florida, and was appointed a distinguished visiting professor of computer science and engineering at Florida Atlantic University.

Aaron served in a large number of capacities as a volunteer in the professional computing arena. He was on the ACM council for one of the longest periods of service (1967-1983), in several positions; these included chairman of the editorial board (1967-1970) and ACM treasurer (1973-1983). It was in that latter capacity and period that I really got to know Aaron personally as well as professionally, because of my own ACM activities. Aaron was also an ACM representative to AFIPS (1978-1982) and served on the AFIPS Executive Committee for part of that time.

Aaron had a number of other positions in ACM. He started as chair of the Long Island chapter in 1960. He served as chair of ACM SIGCSE (ACM Special Interest Group on Computer Science Education) from 1970 to 1973. He was the second editor-in-chief of *Computing Reviews* (1963-1967), and in that capacity he founded the publication that eventually grew into the annual *ACM Guide to Computing Literature*. He then came back for a second term of service as editor-in-chief on *Computing Reviews* and *Guide* in 1987 and remained as editor-in-chief until his death. He received the ACM Distinguished Service Award in 1981 "in recognition of extensive and productive participation in the management of professional society policies and operations." He was in the initial group of ACM fellows appointed in 1994. He was also named a fellow of the AAAS (American Association for the Advancement of Science) in 1983 "for outstanding service to professional societies, to computer science education, and to computer center management."

Aaron's volunteer service was not limited to ACM and AFIPS. Among other activities, he served on the board of directors of the Charles Babbage Foundation for the History of Information Processing (1981-1984) and was the program committee chair for the Jerusalem Conference on Information Technology (JCIT) in 1978.

Aaron published several papers in the areas of computing education and management of computing organizations. He was the editor of the book *University Education in Computing Science*, published by Academic Press in 1968. He was on many advisory panels and served as a consultant to numerous organizations, including IBM, the Federal Reserve System, and NASA's Goddard Space Flight Center. He traveled abroad extensively for IBM, giving lectures to numerous groups in many foreign countries.

It should be clear from the above that Aaron was a pioneer in computing center management in several organizations, and made significant contributions to the early days of computer science education and professional society publications. Aaron's broad professional interests enabled him to see many issues from several different perspectives. He had a quiet demeanor that sometimes made him seem shy. However, he was a man of strong principles and determination, and a gentle manner of persuading people to his viewpoint. He provided assistance and guidance to numerous people and organizations, and continued actively working for things in which he believed. Those of us who were his friends already miss him. Those who did not know him are poorer for the lack of his continuing contributions.

Jean E. Sammet

How Ethernet Was Invented

Editor's note: In this article, Bob Metcalfe tells how he and David R. Boggs invented Ethernet at Xerox PARC.

People often ask me whether I invented Ethernet, really? I would like to think that they ask because they admire Ethernet so much, but then maybe there is something about me that makes them doubt I could invent anything. In either case, yes, I really did invent Ethernet, but not just once, certainly not alone, and all this depends on what you mean by invent.

Roughly speaking, I invented Ethernet three times: the raw technology with David R. Boggs at the Xerox Palo Alto Research Center (PARC) in 1973, the industry standard with Gordon Bell at Digital Equipment Corporation (DEC) in 1979, and the breakthrough product with Ronald C. Crane at 3Com Corporation in 1982.

The 20th anniversary of Ethernet's first invention — of the raw technology — has passed. And now with Ethernet connecting 10 million computers worldwide, it has become the plumbing in what some are starting to call the infrastructure of the information age. So perhaps it is time to tell this story, of how Ethernet was invented the first time, with Dave Boggs at PARC.

Packet switching in ARPAnet. My enthusiasm for computer networking began in 1969. Graduating (with bachelor's degrees in electrical engineering and management) after five years at the Massachusetts Institute of Technology, I began work in computer networking as a PhD student in applied mathematics at Harvard University. I was actually still more of an MIT engineer than a Harvard mathematician, so, while continuing at Harvard, I found a full-time job working on the Advanced Research Projects Agency (ARPA) Computer Network (ARPAnet) at MIT's Multiple Access Computer Project (Project MAC).

ARPAnet was a large-scale experiment in computer packet-switching technology. At various universities and research establishments around the United States, ARPA installed switching computers called Interface Message Processors (IMPs). I built ARPAnet interface hardware and operating system software for a DEC PDP-10, connecting it to the ARPAnet's sixth IMP, installed on the ninth floor of 545 Technology Square at MIT. I designed networking protocols and wrote experimental networking protocol software. I also became an ARPAnet "facilitator," a roving technical expert for new ARPA sites who wanted to join the network.

In 1972 I completed my PhD research on ARPAnet and was looking forward to joining the newly formed Computer Science Laboratory (CSL) in the Xerox Palo Alto Research Center. While packing to move to Palo Alto, I went to defend my thesis before my PhD committee at Harvard. To my horror, Harvard rejected the thesis on the grounds that it was not theoretical enough. I left for Xerox PARC anyway. Bob Taylor and Jerry Elkind, managers of PARC/CSL, were nice enough to take me without my PhD.

Randomness in ALOHAnet. As of June 1972, I was the networking guy at PARC. I built my second ARPAnet interface, for PARC's Multiple Access Xerox Computer (MAXC), a PDP-10 clone, and I continued to work with ARPA on promoting ARPAnet. On a visit to ARPA in Washington, I slept on the queen-size sofa bed in the guest room of my friend Steven P. Crocker, a program manager at ARPA. Next to Steve's sofa bed was a bookshelf where I found *American Federation of Information Processing Societies Conference Proceedings*, Volume 37, Fall 1970, a sure cure for jet-lag sleeplessness. I began reading "The Aloha System," a paper by Professor Norman Abramson of the University of Hawaii.

Abramson's paper described a packet radio network that connected terminals scattered throughout the Hawaiian islands to a central IBM System 360 on Oahu. The 360 was front-ended by a central ALOHAnet communications controller called a Menehune, named after the Hawaiian imp, because it was a packet-switching computer similar in function to the ARPAnet IMP. I did not find this naming hysterically funny.

But the ALOHAnet was interesting because packets inbound to the Menehune were not centrally polled. They were broadcast when ready at the terminals and were identified by the terminal identification numbers they carried. When more than one inbound transmission resulted in interference, the lack of an acknowledgment from the Menehune on the outbound radio channel would cause the interfering terminals to retransmit after a randomly chosen interval, seeking to avoid a repeated interference.

Abramson's paper developed a beautiful model of the Aloha randomized-retransmission radio channel with varying packet traffic. In a simple application of Poisson queuing theory, Abramson showed that an Aloha channel can be loaded just so far — 17 percent — and then it loses throughput to self-generating packet retransmissions. The Abramson paper was about packet networking, my specialty. It used mathematics that was familiar to me from my courses at MIT. But, to make the Poisson analysis tractable, it made two assumptions about computer terminal user behavior that, on Steve Crocker's sofa bed late at night, I found totally unacceptable. Abramson's model assumed there were an infinite number of terminal users and that each of them would go on typing, whether or not they received answers to earlier inputs.

That night, and in the weeks to follow, I worked hard on the less-tractable mathematics of Aloha channels with a few users, each of whom would insist on receiving a response to his input before typing a new one. I worked so hard, in fact, that Xerox sent me to work with Abramson for a month — in Hawaii! By October 1972 t' is grueling effort resulted in a paper, "Steady-State Analy , of a Slotted and Controlled Aloha System with Blocking." It appeared as ARPAnet Satellite System Note No. 16, and was later published in the *Proceedings of the Sixth Hawaii International Conference on Systems Sciences*, January 1973. My model of the Aloha channel confirmed Abramson's discovery of the channel's instability, refined his calculations of throughput versus offered load, and showed how a traffic-based control on retransmission rate could keep the channel stable under heavy loads.

My model of the Aloha system was not immediately embraced by the queuing theory establishment. I recall an unpleasant October afternoon in Washington's National Airport with queuing theory maven and ARPAnet hero, UCLA professor Leonard Kleinrock. He and I had just both spoken at ARPAnet's October 1972 coming-out party, a large demonstration of ARPA's new packet-switching technology. Kleinrock did not immediately, and probably does not today, admire the rigor of my queuing analysis. I think in retrospect the problem was my mathematics did not use the Greek letters of traditional queuing theory. I insisted in those days — and these days — on writing with a word processor. In 1972 word processors were lucky to have upper and lower case type, let alone Greek fonts. Refusing to ink in my formulas by hand, I "word processed" them using plain old alphanumerics. My apparently unencrypted math was hard for queuing theorists to swallow. A year later, in January 1974, one of Kleinrock's students published a paper restating what seem to be my mathematics. He manually inserted Greek letters with India ink. Another of Kleinrock's graduate students would later give Ethernet its tedious technical name: CSMA/CD — carrier sense multiple access with collision detection.

Still, my questionable non-Greek ALOHAnet paper became a chapter in my enlarged and more mathematical (but still non-Greek) Harvard PhD thesis, which then was accepted without enthusiasm by Harvard in June 1973. MIT, not Harvard, published my thesis in December 1973 as "Packet Communication" (Project MAC Technical Report 114).

Personal computing at PARC. In the early 1970s, with Steve Jobs just entering high school, Xerox PARC was engaged in pioneering work on what we called distributed

computing and personal computers. We used the term personal computer (PC) in a general way to mean a single-user computer, not yet distinguishing as we do today PCs from workstations, nor PCs from non-DOS computers like Macintoshes. By the time of my arrival in July 1972, PARC was well along in demonstrating its on-line office system (POLOS), a network of mouse-controlled, bitmapped, high-resolution terminals. POLOS terminals were star-wired to a shared pool of microwave-oven-sized Data General Nova 800 minicomputers, interconnected by the DG Multiprocessor Communications Adapter (MCA).

In the fall of 1972, with MAXC and POLOS up and almost running, Alan Kay, Butler Lampson, Chuck Thacker, and Ed McCreight began work on PARC's pioneering personal computer, called the Alto (from Palo Alto). The Alto was to embody some of Alan Kay's early ideas for a so-called DynaBook. It was discussed as an early — although huge and expensive — DynaBook prototype. Chuck Thacker designed the Alto between September and November 1972, and worked on building it well into April 1973. With MAXC communicating in the ARPAnet and while polishing my PhD thesis, I took on the job of building a local network for the Alto. Notice that the term LAN, short for local-area network, had not yet been coined — that would take seven more years.

The principal use of the MCA network among our hot 16-bit 800-nanosecond Novas was to connect them to a Xerox Graphics Printer (XGP), a high-end Xerox fax machine salvaged for use like one of today's laser printers. The MCA also served to transfer files among the Novas and to connect the Novas as interactive terminals to MAXC. The new network I was to design for the Alto had to be fast enough to do all of these things, and more — it had to be able to handle the transfer of bitmapped printer files to the higher speed, higher resolution laser printers then under development at PARC.

If the uses were basically the same, why not just build an MCA for the Alto? We were planning to have at least one Alto per desk throughout Xerox PARC, which amounted to hundreds of personal computers separated by hundreds of meters. The MCA could transfer files as fast as eight million bits per second, but it did so 16 bits at a time over a thick 40-foot cable among at most 15 computers. By slowing the MCA down to four megabits per second (Mbps), we could extend the MCA to 400 feet, but then it ceased to work reliably. By using two MCAs with one Nova on both, we were able to network up to 29 Novas, but that was the limit. The ultimate 29-Nova MCA daisy chain had 28 40-conductor cables, 60 40-conductor connectors, and the nasty habit of crashing if any one of these fragile devices was disturbed. The worst part of overextending our MCA was that the networking protocol software written to move data through it assumed that the MCA was error-free — a bad assumption. The network started to hang several times a day. We all saw that we needed a highly reliable bit-serial in-building computer networking technology that would transfer at least hundreds of thousands of bits per second among hundreds of computers separated by hundreds of meters. The MCA was not even close.

Neither was another new network project at PARC called LocalNet. Charles Simonyi, then a Berkeley undergraduate, had been working with Lampson and Thacker on a very-high-speed local version of the ARPAnet — nicknamed Simonyi's Infinitely Glorious Network (SIGnet). It was to be a store-and-forward packet network carrying data locally among computers at 50 Mbps — a thousand times faster than ARPAnet. Charles handed SIGnet over to me so he could work on Bravo, a new bitmapped text editor for the Alto that would lead to the Xerox Star, the Macintosh, and Word, which Charles later developed at Microsoft. It took me a week before I also dropped SIGnet as, well, having too many moving parts for a local network.

While packing to move to Palo Alto, I went to defend my thesis before my PhD committee at Harvard. To my horror, Harvard rejected the thesis on the grounds that it was not theoretical enough. I left for Xerox PARC anyway.

A LAN for personal computers. The new local network had to be fast and simple. It had to be everywhere, yet not require a lot of thick cables. It had to be reliable. It had to cost no more than 5 percent of the cost of the PCs it was to attach. It had to fit on one of the plug-in adapter cards planned for the Alto. There were already schemes afoot outside PARC (at Dave Farber's University of California, Irvine, lab, for example) that polled for packets by circulating a token among computers wired in a ring. So, what about applying ALOHAnet?

For one thing, ALOHAnet used radio. We couldn't use (and didn't need) radio because our PCs were to cost, in 1973 prototype shop dollars, about $45 thousand, and were to remain immovable under desks. Instead of using thin air, or the "ether" as physicists used to call it, what if we used a long cable strung all over the building in an Alto ALOHAnet? Any PC wishing to be on the network would simply tap into the cable and transmit its data packets into the cable's ether.

The ether in Ethernet comes from the "luminiferous (a)ether" once thought to pervade the universe as the medium for the propagation of light. American scientists Michelson and Morley disproved the existence of the old ether in 1887 and thereby set Einstein off on his theory of relativity. The new ether, in Ethernet, is also a ubiquitous passive medium for the propagation of electromagnetic waves, in our case, data packets.

In Ethernet there wouldn't be a central host computer, or network controller, but hundreds of PCs and a variety of servers and hosts — nodes in the network. We wouldn't need two cables, one inbound to the ALOHAnet host and one outbound to the terminals. There would be one reliable unpowered ubiquitous medium, the ether. Packets would have

two addresses, one identifying the destination of a packet, and the other its source.

Using low-noise low-attenuation cable instead of radio, network adapters could easily listen for and avoid transmissions already in progress. In the unlikely case that two or more interfering transmissions began at the same time, network nodes could detect the conflict at each transmitter and, unlike ALOHAnet, abort the damaged transmissions before wasting channel time. In contrast to ALOHAnet, where fixed-length packets had to be short, our packets would vary in length and sometimes would be quite a bit longer, thus further improving channel efficiency. And to top it all off, when retransmitting after conflicts, high, stable throughput could be maintained by using a traffic back-off control technique, as first suggested in my PhD thesis.

Oh, by the way, I forgot to mention that, by using cable instead of radio, we easily could afford to send our packets at data rates over a thousand times faster than on ALOHAnet, in a target range of one to 10 Mbps.

On May 22, 1973, using my Selectric typewriter with its Orator ball (which I then used instead of word processing to dash off short items), I wrote up all this in a 13-page (starting at 0) Xerox Sensitive memo. The memo, to "Alto Aloha Distribution" from me on the subject "Ether Acquisition," was heavy with handwritten annotations — one of which was "ETHER!" — and with hand-drawn diagrams — one of which showed "boosters" interconnecting branched cable, telephone, and radio ethers in what we now call an internet. In the memo I described how transmission "conflicts" could be detected. Later you'll see why I now wish we had stuck with calling them conflicts rather than collisions. I called the modified Alto Aloha network by its new name, the ETHER Network. If Ethernet was invented in any one memo, by any one person, or on any one day, this was it.

On June 28, 1973, I submitted an "ETHERNET Invention Record" to Xerox's formidable legal department. The record included Alto Aloha memos, dated April 19 and 24, and the ETHER Network memo, dated May 22. I acknowledged Butler Lampson and, despite our differences, Chuck Thacker for their contributions to the earliest thinking on our new local network. It was unthinkable, because they were so influential in everything anyone did at PARC, that Butler and Chuck would be left off my list of Ethernet inventors.

Because of a distaste for uppercase letters at PARC (often then spelled Parc), our new network's name later became EtherNet and then Ethernet. Inside Xerox we called it "the Xerox Wire" between 1976 and 1979, and various standards bodies continue to this day to call it CSMA/CD, but the original name Ethernet finally stuck.

I closed Ethernet's invention record by saying, "the time is ripe for the development of...ETHERNET in that the interconnection of devices is now the hardest problem to be faced in harnessing small chunks of computing."

Reduction to practice. Shortly after deciding to pursue Ethernet, I bought a 1,000-foot reel of coaxial cable and started launching bits down it to see for myself what would happen. My test jig was in the basement of PARC building 34, in the same area where a band of computer engineers were cloning a batch of Bose speakers. One day while soldering, I saw Dave Boggs padding into the basement in his moccasins, his blond ponytail swinging to and fro across the shoulders of his yellow buttoned-down shirt. Dave and I shared the basement for a while as he uncrated and brought up PARC's new Nova 800s as they arrived, one per week. Dave could see that I hadn't soldered much. I could see that he was being underutilized.

Dave, a PhD student in electrical engineering at Stanford, was working on POLOS for my good friend, David Liddle. Despite his summa cum laude electrical engineering degree from Princeton in June 1972, he was known to be a hardworking and practical engineer. He started early, earning his FCC first-class commercial radio operator's license while a high school junior. Since high school, Dave had been building electronics for NASA, the US Navy, his own amateur radio station, Princeton's commercial FM radio station, Princeton's computer center, and NBC TV and radio in Washington, where he grew up. He knew a lot about electrical engineering and computer science, including how to solder.

In June 1973, just as the first Alto CPU, memory, display, and disks were being debugged, I recruited Dave Boggs to work with me. He had just completed his first year at Stanford, earned his master's degree, and qualified for the PhD program. Being a radio wizard, he recognized Ethernet's conflict detection scheme as something familiar to him, a scheme he called full break-in keying. By the fall, he was in so deep that he took a leave from Stanford to continue working with me on Ethernet. Dave was not to get back to his PhD work for a long time. It was April 1982 before he became Dr. D.R. Boggs.

Dave and I were called the Bobsey Twins in 1973 and 1974 as we designed, built, and operated a 100-node experimental Ethernet. We used to work until we were exhausted and would sleep until we woke, without regard for alarm clocks or the sun. It was a major inconvenience for us to attend the only mandatory meeting PARC had in those days, a meeting called Dealer every Tuesday at 11 a.m. Dave is certainly Ethernet's coinventor.

The first Ethernet cable did not need real transceivers as it was just a clip lead from the transmit side to the receive side of our Alto Ethernet card. The second Ethernet was a pair of twisted three-foot wires between our first Alto adapter card and our first Nova adapter card. When we had our third Ethernet adapter, however, real multipoint communication required real transceivers.

My MIT training ran from mathematics on the left, through computer programming in the center, and as far right as digital computer hardware engineering, but not into the fascism of analog electronics. Dave Boggs was therefore a better choice to design and build Ethernet's first transceivers. And he did — three of them, by hand. These devices drove data bits onto the ether cable by turning a voltage on and off. They received bits from the cable by detecting voltages above a certain threshold. Think of them as modems, only a thousand times faster than the modems commonly used for data communication through the telephone network.

Each bit was put on the cable, according to the rules of Manchester encoding, preceded by its complement so that the ether would remain busy for the duration of a packet regardless of its data patterns, and so that the transceiver could check during half of each bit time whether some other transceiver was interfering with its transmission. Of course, all of this sounds digital, but it became more analog as the cable became longer and as bits turned into sine waves. Transceivers have to maintain the properties of the cable as a transmission line and be able to recover data, despite the distortions of frequency-dependent cable attenuation and impedance distortions at connectors and taps. Transceivers also have to electrically isolate their stations from the ether cable to avoid the buildup of dangerous voltages, especially during lightning storms.

Getting our bits past hundreds of taps over hundreds of meters of cable at 3 Mbps was beyond even Dave's knowledge. We struggled for a while to get real transceivers built. Finally, we thought of Tat Lam, a quiet little Chinese man who was working on contract designing CRT display electronics in the basement at PARC. Tat agreed — and his word is as good as his bond — to design the first two breadboard Ethernet transceivers for $2,000, and then to deliver the first 10 printed-circuit prototype Ethernet transceivers for another $2,000. That has to be the best bargain in electronics history. Those transceivers were capable of carrying 3 Mbps among 256 stations over one kilometer of half-inch coaxial cable. Tat Lam went on to start his own company, TCL, Inc., to produce Ethernet transceivers for Xerox and others.

The experimental Ethernet was to be quite fast for its day. In those days the 50-kilobit-per-second (Kbps) telephone circuits of the ARPAnet were considered fast. Ethernet's actual speed was 2.94 Mbps because each Alto clock interval was 170 nanoseconds and we used two clock cycles per bit to drive our Manchester encoder. I was fond of saying that the round off from 2.94 to 3 Mbps was larger than the ARPAnet's 50 Kbps. This was my way of declaring that most intuitions from experiences with remote data communication were obsolete in local networking.

This experimental Ethernet sent and received data packets interspersed with quiet on the cable. To transmit a packet, an Ethernet adapter would wait for quiet, send a 1-bit preamble, an 8-bit destination address, an 8-bit source address, a variable number of 16-bit data words, and then finally a 16-bit cyclic redundancy checksum (CRC).

To receive a packet, an Ethernet adapter would wait until the rising edge of a bit broke the quiet on the ether, and then collect the subsequent 8-bit address to check whether it matched its own or indicated a broadcast packet intended for all stations. If so, the adapter would copy the rest of the arriving bits into main memory while computing a CRC to match the last 16 bits received. If interference was detected during transmission, the adapter would immediately stop and signal its station that a randomized retransmission had to be scheduled. If any repeated retransmissions occurred, they would be retried after a doubled random delay — binary exponential back-off. If any interference was detected during reception, or if the computed CRC of a packet did not match the CRC it carried, then the reception would be terminated and any received data discarded.

Dave and I were looking for a way to connect our networked PCs into the ether without bringing it down. We were using familiar BNC T-connectors to attach our transceivers to the ether, when David Liddle, who had worked installing CATV systems while in college, told us about the Jerrold tap. This was a device that required no cutting of cable, no bringing down of the net. You just drilled a hole for the tap, while packets sailed by unharmed. We bought some Jerrold taps and gave them to Tat Lam for inclusion in his prototype transceivers. Tat designed his transceivers so they could take T-connectors or Jerrold taps. By selecting these taps, we chose to use coaxial cable for the experimental ether.

My MIT training ran from mathematics on the left, through computer programming in the center, and as far right as digital computer hardware engineering, but not into the fascism of analog electronics.

Early opponents. Our Ethernet work attracted some attention from a Xerox physicist elsewhere in PARC, Robert Z. Bachrach. Bob wrote a stinging attack. His main complaint was that Ethernet made poor use of bandwidth. He attacked what seemed to him to be nothing more than lazy science. His memo concluded that Ethernet was not quantum noise limited. Of course, nothing in computers had ever been close to quantum noise limited, and Ethernet was not the place to start. Bob had missed the point and got himself into a bit of trouble for writing a memo to my boss, Dave's boss, and their boss, instead of talking things over with us first. I show the old memo to Bob every now and then just for the fun of it. Bob's recollection of his memo and of his discussions with Dave and me are quite different from mine — if you ever get to see his memo and his explanation for it in 1991, you will be amazed at how differently two people can view the same events. I can only say time heals all wounds, and, while success has many fathers, failure is an orphan.

Later, while we were debugging the Alto adapter and struggling to get the right transceiver design, the Ethernet project fell on hard times. Chuck Thacker, the principal developer of the Alto and a first-rate engineer, lost confidence in the Ethernet project. I think this was partly because he and I never quite got along. I was always giving him gas for the unreliability of his prototypes, first on MAXC and later on the Alto, which Dave Boggs and I found ourselves debugging long after Chuck declared it done. Or chalk it up to bad chemistry. Anyway, when Ethernet developments faltered,

Chuck began work on another local network idea for the Alto, which he called Xnet. Xnet looked to me like the MCA — a bad idea. I complained about Xnet to the head of PARC/CSL, our boss, Bob Taylor. Xnet was abandoned shortly thereafter. Had Chuck continued with Xnet, Ethernet never would have seen the light of day. Chuck's status in the lab, his role in allocating resources, and his engineering skills would have proven insurmountable. With a smile, Bob Taylor denies having anything to do with shutting down Xnet, thereby saving Ethernet. Admiring Bob's light touch, I have since been accused of being very naive. I doubt anyone knows what really went on here.

As we designed the Alto Ethernet adapter, Boggs and I wrote many programs simulating network operation. These simulations confirmed over and over again that few of our design decisions mattered. The binary exponential back-off algorithm that we settled on for controlling retransmissions with traffic load was not much better than any of the alternatives, but it was easy to implement with then-current medium-scale-integrated (MSI) transistor-transistor logic (TTL) hardware and the microcode of the Alto. Unlike the Aloha channel from which it had been derived, the Ethernet channel was stable and operated efficiently, no matter what we threw at it. In retrospect, we should have published the results of these simulations to preempt a lot of long-lingering questions about Ethernet. Instead, we wrote the simulations as design tools and threw them away.

Dave and I designed the first Alto Ethernet adapter using 67 MSI TTL chips and then, while it was being wire-wrapped, we designed one for the Nova. We debugged both adapters and had them built in lots of 10. Dave later went back and redesigned both adapters — the Alto card went up to 84 chips — because they did not meet Dave's standards, and because Chuck Thacker was still (unreasonably, I thought again) demanding that we do better.

Dave and I were awarded two of the first 30 "production" Altos. We called them Michelson and Morley after the two scientists who freed up the word ether for our use. Ethernet was, at first, an option you ordered with an Alto if you felt you needed networking. If you had an extra $500 in your budget, you ordered Ethernet. Money was, as usual, tight, and so some people were ordering Altos without Ethernet and with 48 Kwords instead of the full 64 Kwords of main memory. These false economies were soon dropped as it was discovered that standard configurations were a big help in developing software and maintaining a stable operating environment.

Early uses and protocols. Ethernet was used initially for file transfer. This application was marginal, as the early Altos came equipped with a pair of two-megabyte removable cartridge disks — a cross between a floppy and a hard disk. People could easily copy their disks and use them for file transfer. Ethernet was up against "sneakernet" from the very start.

Ethernet was (and is) a "best-efforts" transmission system, as I defined the term in my PhD thesis. This means that Ethernet delivers data packets with high probability but does not itself alone guarantee reliable delivery. Rather, Ethernet relies on higher level packet protocol software to keep track of transmitted packets and to retransmit them when acknowledgment packets fail to return. In short, Ethernet does not distinguish between the data packets and the acknowledgment packets of higher level protocols. I argued in my thesis that this was a good approach because higher level protocols, especially internet protocols, have to check for correct operation anyway, and so low-level best-efforts transport mechanisms should not duplicate this checking activity unless underlying transmission error rates are very high, which they not often are. People still argue about this, but let's just simplify and assume that I'm right. Read my thesis.

So, for the experimental Ethernet, I designed a very simple protocol for transferring files reliably between Altos — the Experimental Ethernet File Transfer Protocol (EEFTP). I implemented it for the Alto running the standard Alto operating system, and John Shoch, then a graduate student at Stanford, implemented it for the Alto running SmallTalk, an experimental object-oriented Alto programming environment. John would later do extensive work, and earn his Stanford PhD, confirming the stable performance of Ethernet in widely varying traffic situations.

In 1976 EEFTP was written up in the first Ethernet paper as an example error-correcting packet protocol. It was updated with internetworking to become EFTP (dropping "experimental"), and later became what is today the Defense Department's Transmission Control and Internet Protocol's (TCP/IP's) Trivial File Transfer Protocol (TFTP).

You would start EEFTP on one Alto listening to receive a file. Every second it would display a new letter Z on the screen to indicate how long it had been listening (sleeping). You would then start EEFTP on another Alto to transmit a file, specifying the name of the file on the local disk and the address of the listening Alto. Each 512-byte packet transmitted would be marked with a dot on both the sending and receiving displays so that you could watch the progress of your transmission. EEFTP proved very useful when a bug was discovered in the Alto disk controller that required the disk format to be changed. Old disks were shoved into Altos with old disk controllers and then copied over the Ethernet using EEFTP to new disks on Altos with new disk controllers. The old disk controllers were then updated. Ethernet was beginning to pay its way.

Birth of the laser printer. In 1975 I worked on one of Ethernet's most important applications — laser printing. Xerox, you may recall, makes copiers. A Xerox researcher, Gary Starkweather, had been working on using lasers to write on the drums of Xerox copiers so that they might be used as printers. Gary had developed a scanning laser output technique (SLOT) and lashed it to one of Xerox's marking engines, as the copiers are called. Another Xerox researcher, Ron Rider, had been working on a research character generator (RCG) to drive the SLOT with text, fonts, and graphics, updating the old Xerox Graphics Printer substantially on speed and resolution — a page per second at 500 dots per

inch. Ron Rider decided to switch from a Nova to an Alto, which he named Palo, to control his RCG. He approached me about using Ethernet to bring documents into the Alto for printing. I signed on to provide Ron with a multitasking network operating system for the Alto and the networking protocols needed to bring documents onto the Alto's disk. Ron took documents from the disk and printed them through RCG onto SLOT. Ron and I worked together until he finished his laser printer, EARS, an acronym of acronyms short for Ethernet Alto RCG SLOT. Ron's Palo Alto was a rack three feet high. The RCG was a rack six and a half feet high. The SLOT was in a copier enclosure, about four feet high and three by six feet. A later version of EARS was called Dover because of the white cliffs of paper needed to feed it (get it?). The arrival of EARS destroyed any hope of achieving any time soon the long-sought paperless office.

Networking is not an option. It was not long after EARS began printing that Ethernet became nonoptional. I recall the day I accidentally removed a cable terminator from the ether and stood up among the cubicles only to find one after another of my colleagues popping up, wondering why the network was down. From that day on, no Altos were ordered without Ethernet, and many were ordered with only one disk. Sneakernet was dead, at PARC anyway. And so the PARC Ethernet started growing rapidly — so rapidly that it soon approached its limits. The first to be hit was the limit of one kilometer of cable. Dave Boggs and I designed, and Dave built, the first Ethernet repeaters, upon which Xerox holds the patent in our names. Ethernet also connected the Altos as terminals to MAXC. For this purpose, on my new operating system, I began writing a program called ETelnet. ETelnet would take typed characters and send them in packets over the Ethernet to MAXC, and receive packets from MAXC with characters for display. ETelnet was taken over and made to work by Bob Sproull.

Packet protocols. While finishing the experimental Ethernet, I proposed and developed a set of internetworking protocols based on what I called the PARC Universal Packet (PUP). These protocols grew out of discussions with Peter Deutsch at PARC, and out of a series of seminars I attended at Stanford during the summer of 1973, from which Professor Vinton G. Cerf collected the early ideas for TCP/IP. Among the attendees of Cerf's protocol seminars was Gerard Lelann, who contributed many packet-switching ideas from the Cyclades and Cigale packet-switching projects in France.

PUP was aimed at internetworking local networks, while TCP/IP, a next generation for the ARPAnet, was aimed at remote networking. PUP later interacted with an early version of TCP to get internetting — IP — added, making it TCP/IP. PUP was picked up by Dave Boggs, Ed Taft, and others at Xerox PARC. In 1975 the PUP File Transfer Protocol (FTP) was based on the ARPAnet FTP. Thereafter, PUP became the Xerox Network System (XNS) protocols, which in turn after simplification became the widely installed Novell IPX. Without PUP and later TCP/IP, Ethernet would have gone nowhere.

After picking up PUP, Dave Boggs found an old Nova with two very large disks left over from a voice server project completed by David Liddle. Boggs wrote a PUP-based file server system he called WFS, for Woodstock File System. WFS much later became NFS, Sun Microsystems' Network File System. Boggs and Taft later, in 1976, took PUP and another old Nova and developed, as described in the Ethernet paper, a LAN-WAN internetwork router, perhaps, maybe, possibly, the granddaddy of all internetwork routers.

Ethernet was, at first, an option you ordered with an Alto if you felt you needed networking. If you had an extra $500 in your budget, you ordered Ethernet.

The paper and the patent. In 1974 I began work on a paper describing Ethernet. The legal department at Xerox was determined that this paper would not be published, at least not until proper patent applications were filed. The Ethernet patent was filed on March 31, 1975. Considerable work went into filing the patent because we had to describe parts of Ethernet in hardware terms when they were best implemented in software. At one point, much to the consternation of Xerox's patent counsel, I tried to draft a patent on Ethernet that would use patent legalese but remain understandable (this turns out to be a contradiction in terms).

In May 1975 I submitted the Ethernet paper, already a year in revision, to the *Communications of the Association for Computing Machinery* (*CACM*). The paper was reviewed by editors including DEC's Gordon Bell and Sam Fuller, who will reappear in some future story about Ethernet's second invention. *CACM* finally published "Ethernet: Distributed Packet Switching for Local Computer Networks," by Boggs and Metcalfe, in its July 1976 issue.

In the Ethernet paper, Boggs and I described Ethernet nodes as 16-bit (and up) network stations with memory and computing cycles to burn. Each station had its own inboard Ethernet controller microcode and interface hardware driving a five-pair interface cable leading, typically through a wall up into the ceiling of the passing hall, to an Ethernet transceiver with a tap into the passing ether cable. The transceiver cable carried transmitted bits up from the station to the transceiver, carried received bits down, carried an interference detection signal down from the transceiver, and carried two flavors of power (+5V and −15V) up to operate the transceiver. The ether was not a star, not a mesh, not a ring, but an ether, a branching bus, an unrooted tree, passive, without central control — in the simplest case, a coaxial cable terminated at both ends.

I presented the Ethernet paper at the National Computer Conference in June 1976. By then my wife of seven years had divorced me — I guess maybe I had been spending too

much time at the lab. I left Xerox, moved to Los Angeles, but seven months later I returned to Palo Alto to join David Liddle in Xerox's Systems Development Division to begin work on, among other things, the productization of Ethernet.

The Ethernet patent, Multipoint Data Communication System with Collision Detection, United States Patent 4,063,220, was issued on December 13, 1977, with Metcalfe, Boggs, Thacker, and Lampson as inventors, and with 19 fine-print pages, 22 broad claims, and 7 drawings. Ethernet was invented, officially, for the first time.

After the first invention. Any thoughts that Dave and I had in 1976 about the world beating a path to Ethernet's door were abruptly smashed as I delivered my NCC talk that year. Ethernet was much appreciated inside PARC, where PCs and laser printers were the norm by 1976. However, outside PARC, in the real world, we discovered that few people understood why anyone would want a 3-megabit-per-second local computer network. There was barely one computer per building, not hundreds. I found myself on conference panels with people who said that 1,200-bit-per-second modems, at less than one thousandth of Ethernet's speed, were already overkill, since people were having trouble reading computer outputs streaming past on terminals at 600 bits per second. And then I told them about "collisions," which conjured up misleading images of automobile accidents — this is why I wish I'd stuck with "conflicts" rather than "collisions."

Dave and I did not know in 1976 that Ethernet would have to be invented at least twice more. The idea of a computer industry standard Ethernet, later the IEEE's Project 802.3, would have to be invented in 1979. And the right Ethernet product, 3Com's EtherLink, would have to be invented in 1982. By now Ethernet is so widespread that I have had to get used to the idea that it is being reinvented every day, and by total strangers.

Dave Boggs stayed on at Xerox long after I left in 1979 to found 3Com. Then he moved to DEC's Western Research Laboratory in Palo Alto, where he is today. Does the co-inventor of Ethernet regret not founding 3Com with me? No. Dave stayed in research to get his PhD and to avoid the intellectual distortions of tightening schedules and reducing costs. He values his enthusiasms highly. He has developed his wine appreciation at tastings twice a week, grown his antique US military gun collection, built his own car from a Lotus 7 kit, continued his amateur radio, and assembled a first-rate machine shop in his home, with a Hardinge lathe and a Bridgeport milling machine — the world's finest. And, oh yes, he has become a millionaire anyway; it seems he plays the stock market as well. His only regret so far seems to be that he is not yet married and raising kids. I tell him that his baby, Ethernet, is doing just fine.

Acknowledgments. Comments on this account of Ethernet's invention at PARC were received from Robert Z. Bachrach, Gordon Bell, Vinton G. Cerf, Ronald C. Crane, Steven P. Crocker, Peter Deutsch, Alan Kay, Ed McCreight, Ron Rider, and Charles Simonyi. Any errors in the article, however, are my own.

Robert M. Metcalfe

Reviews

PEGGY A. KIDWELL, EDITOR

The Reviews department includes reviews of publications, films, audio- and videotapes, and exhibits relating to the history of computing. Full-length studies of technical, economic, business, social, and institutional aspects or other works of interest to Annals *readers are briefly noted, with appropriate bibliographic information.*

Colleagues are encouraged to recommend works they wish to review and to suggest titles to the Reviews editor.

❑ John Barrow, *Pi in the Sky: Counting, Thinking, and Being*, Clarendon Press, Oxford, UK, 1992, 317 pp., £19.95.

This review was to have begun with a short homily on puns as a comment on the title of the volume being reviewed. However, criticism, explicit or otherwise, of this subject in the *Annals* may not be wise since an article on the history of the lambda calculus that appeared a few years ago began with the statement that "Kleene-ness is next to Gödel-ness."[1] So let us just say that however we may feel about the use of puns in formal writing, the use of one as the title of this book should not deter the prospective reader.

Pi in the Sky gets its title from the Platonic view that mathematics is a form of objective numerical truth. In the words of the author in the last chapter, which is a long discussion of the realist or Platonic view of mathematics:

> There is an ocean of mathematical truth lying undiscovered around us: we explore it, discovering new parts of its limitless territory. This expanse of mathematical truth exists independently of mathematicians. It would exist even if there were no mathematicians at all — and, indeed, once it did, and one day perhaps it will do so again. Mathematics consists of a body of discoveries about an independent reality made up of things like numbers, sets, shapes, and so forth. "Pi" really is in the sky.

This book is a development of an argument begun in the last chapter, "Is 'pi' really in the sky," of the author's recent *Theories of Everything*.[2] A summary of much of the material in the book is given in his more recent article in the *New Scientist*.[3]

Most of the book is devoted to a very readable account of the several views of mathematics that have been held by mathematicians and users of mathematics: the logistic view that mathematics is an extension of logic, the intuitionist view that mathematics includes only those results that may be derived from basic axioms in a finite number of steps, the