Oral History of Raymond (Ray) Tomlinson

Interviewed by:
Marc Weber and Gardner Hendrie

Recorded June 5, 2009
Cambridge, Massachusetts

CHM Reference number: X5409.2009

Marc Weber: Today is June 5, 2009 and I'm here with Gardner Hendrie from the Computer History Museum and we're interviewing Ray Tomlinson, who's a major computing and networking pioneer and well known for email. Thank you very much for joining us.

Raymond (Ray) Tomlinson: You're welcome, glad to be here.

Weber: I wanted to start with a little bit about where were you born, where did you grow up and what led you toward computers?

Tomlinson: Well, I was born in upstate New York in the city of Amsterdam, New York, which is where my mother and father lived at the time. We very shortly moved a little further north, about nine miles to a little, tiny, unincorporated place called Veil Mills but whenever I've been asked I say I come from Broadalbin, which is almost as obscure but it's about nine miles north of the Mohawk River, about 30 miles west of Saratoga Springs. I went there to grade school and high school and when it came time to go to college I was advised that there were many good technical schools around and I had decided pretty much to go into electrical engineering. I had always taken things apart and liked playing with vacuum tubes and things of that sort. So I got off to college at RPI, that's in Troy, New York, and I went there for four years and got involved in the co-op program there, which involved going off to industry for, I think it was three semesters, four semesters if you include the summer after graduating, and I worked for IBM. And IBM, of course, is well known for their expertise in computers but I was not interested in computers. I was a double E, I was [in] electrical engineering. So I worked first on building test equipment to test transistors and after about two or three years of this I finally saw a computer down the hall that was available for the engineers to use and decided I'd learn to program it. It was an IBM 1620, it had a variable field length so you could have huge numbers, numbers up to as many digits as the computer could hold. And so I decided to write a program to compute pi and that was my first computer program.

Weber: And you did it on your own.

Tomlinson: I did it on my own. It had nothing to do with anything I was working on, it was just something to do and was kind of interesting. And then when I got to MIT for my graduate work…

Weber: As an electrical engineer.

Tomlinson: Oh, I was still a double E, yeah. I was working in the speech department there on speech synthesis, which at the time couldn't be done by computers. It was far too compute-intensive to be done on digital computers. But one afternoon I was walking down the hall in building 26 there and there was this darkened room off the corridor and lots of noise coming out of there, yelling and screaming, and I looked in and here's about a dozen people all gathered around a CRT [cathode ray tube] and they were playing Spacewar!. And I said "This is really neat."

So I went in and watched for a while and I said, "I got to learn how to program one of these things." So when it came time to decide what I was going to do for my master's degree I said, "I'm going to work in speech synthesis but I'm going to use this computer somehow in doing it." So I came up with the idea of having analog circuitry that would be controlled by the digital computer. So it was a hybrid, it was half analog circuitry and half digital circuitry. The analog circuitry did all the signal processing but the

computer told it how to set the gain of the amplifier and the resonances of the resonant circuits and that sort of thing and then by drawing with a light pen a contour, for let's say the fundamental of the voice, you could make it go up or down and you could do a similar thing with the performance of the vocal tracts so you could make <long consonant sounds> and that kind of variation. And that was fun and then it sort of took over my life. I stopped going to classes and spent more time on that computer and I did get a master's degree but I never really got into the point where I could get a thesis proposal together to work on my doctorate. So after a while, my thesis advisor advised me that I should come to work for BBN. He consulted here one day a week. We have a fairly strong speech program here. And I did and I've been here ever since. So that's how I got started in computers.

Weber: And as a child you said you were interested in electronics, were your parents in a technical field?

Tomlinson: Not really, no. My father worked for General Electric. He was a machine operator. He, you know, made things, a machinist. My mother was not working at the time, although she had worked in an accounting capacity for a department store before she got married. Later, they went into business in a grocery store and so they sold groceries for a while and I worked stocking shelves, which is not something I would want to do again but it was interesting at the time. So, no, they were not [technical] but they were very supportive and I forget just when it was, I was probably about 12 at the time, they got me a little kit from the equivalent of Radio Shack and it had things you could make, electronic circuits, and that was my first exposure to electrical things. Before that, I'd taken apart other things that were not necessarily electrical, like clocks, and I was very good at taking these things apart. Unfortunately, I never quite ever got them back together again so I figured I would find out how to make some of these things as well as take them apart.

Weber: So you made mechanical things, too.

Tomlinson: Yeah, yeah.

Weber: Models or your own things?

Tomlinson: Yeah, things like, you know, model airplanes. I never liked the part where you had to make it look good but I got it to the point where it worked, you know, it would fly but it looked like crap <laughs>, just the bare wood. You know, why bother painting this thing? It flies just as well without the paint.

Weber: Were you interested in music going on in the <inaudible>?

Tomlinson: Sure, yeah. My father played the trumpet and he actually played in a band in nightclubs during high school but then between high school and again, when he got married, he had played in this orchestra. They had four or five players and so he played the trumpet. And he introduced me to the trumpet. He had a trumpet and a coronet and when I was probably as young as four, started me playing it. And you know, I'd play *Twinkle, Twinkle Little Star* and some simple things. And I think it was his idea, it was not necessarily my idea, that I should play in the grade school band, which was really intended for fourth graders and fifth graders and sixth graders. So I did and I didn't really know what was going on. It was an interesting experience. At one point we had to march in the Halloween parade and I couldn't play and march at the same time but I walked along and got very tired.

So then I stopped playing music for a while and it wasn't until I got into fourth grade and there was a required class that everybody had to take to play what was called a Tonet. It was a recorder-like wind instrument. And the music director for the school recognized me from my experience in first grade and said "Don't you want to play in the junior high band?" <laughs> So I did and that got me really interested. He was a strange character but he was very supportive of making sure that everybody had a chance to play music. And so I played the trumpet, learned to play reasonably well, not wonderfully but not enough to make any money at it certainly. And so yeah, that was my introduction to music and it went away for a while after college. And during college there were three of us in this co-op program that I spoke of before, who all played the trumpet. And so we had our own trumpets. Mine was at home so I brought it down and we'd go into the bathrooms and play because you get all those reverberant hard surfaces from the ceramic and all and we sounded wonderful no matter how badly we played. We had a lot of fun and I've been interested in music ever since. I do a little composing now and I've learned to play a keyboard sort of a little bit but don't play the trumpet anymore. That's something you got to work at more or less continually.

Weber: And then you like math?

Tomlinson: Yes, I do. I like math. I'm an engineering mathematician. That is I learn enough math to do the engineering but I don't feel a need to understand all the intricacies behind the math and, you know, differentiation, yeah. I could probably go through all the rigor that a mathematician does given enough time, or at least recall it, but I don't feel the need to do that as part of my daily working with math.

Weber: And so speech you got interested in fairly early and then when you came here you did go into the speech group?

Tomlinson: No, no, I came here and worked in the computer center. In fact, at that time the speech group was--let's see, how to say this--it hadn't really come together with much yet. There were several people interested in it and it was working out of the psychology group that we had but speech as a part of acoustics was always in the picture here. But as a [branch of] computer science it hadn't really gotten to that point yet. Computers weren't powerful enough to really do much in that area. It was still largely purpose built, mechanical and electrical devices that we used in those studies. So, no, I used my computer experience and my experience with hybrid operation on other projects we had here instead.

Weber: Like what sorts of things?

Tomlinson: Well, we were building, how to say this? We had a fairly strong interest in human/machine interaction and how people did things, doing psychological experiments, how people performed under various conditions, and rather than building more or less very complex mechanical devices, we started using computers to present stimulus to experimental subjects so they could see things and do things and maybe try to drive with a joystick some sort of a display thing. And since the analog part of that was still necessary to do some of the operations, I used some of my experience there, but mostly I fairly quickly got away from that area of things and got more into, you know, ordinary computer problems and then when networks came along I got really involved with networks.

Weber: In what year did you come here?

Tomlinson: I started at BBN in 1967, June 2nd. It's been 42 years and three days or something like that.

Weber: And you were in this building then?

Tomlinson: At the time, I was in the building right across the bridge here, building two. These buildings were all in use. The computer group was over on this side of the house. They were right down the hall there.

Weber: Right, the IMP people were all down where the cafeteria is.

Tomlinson: Right, and I was in building two with all the other-- a lotof the man/machine interaction started this sort of thing since I got the psychology group and--

Weber: Okay, but was there a lot of interaction between the two groups?

Tomlinson: Some, but not a lot. The bridge was actually something of a barrier. In fact, when it came time to get our network connection, it was actually a physical barrier because the early interfaces would only work within a matter of a few feet and that was about 100 yards by the time you go up and down and around and everything. So they had a design and built a distant host interface before we could get our connection here. So we finally did get the connection and so then we had to worry about how to program the computers to use that connection.

Weber: And what were your first impressions at BBN and the atmosphere?

Tomlinson: Well, it was very, very relaxed, very nonchalant about a lot of things. They weren't really too concerned about going through a lot of procedures and rigor in terms of hiring me. I mean, I probably talked to somebody from their human resources [department], whoever he was or she was. I have no recollection of it so it had to have been a pretty informal process. At the time, the kinds of contracts that we had had were very open-ended in terms of what our statement of work was. They were very fairly general statements of what sorts of things we should be investigating. So there was a lot of room for coming up with new ideas and, you know, if you had said, "Yeah, there's a machine here and there's a human here and they're interacting," that's man-machine interaction. So it was fairly relaxed and I thought it was quite productive in that sense.

Weber: Similar to an academic environment in that way.

Tomlinson: More of an academic environment than a lot of other places.

Weber: And people were friendly?

Tomlinson: Yes, yeah, it was just friendly. Everybody was interested what other people were doing and you could always find somebody to ask a question of and say, "Is this a good idea?" And somebody would say "No." Okay, sometimes there were bad ideas as well as good ideas.

Weber: So then describe how you moved towards the computing group here.

Tomlinson: Well, let's see, there was a research computer center, which actually ran the computers and that's what I was part of. The researchers, the people using the computers, were [running] a lot of LISP at the time and the problem they were having is that most of these machines had limited random access memory so they were using a lot of what amounted to virtual memory. It was sort of a homebrew virtual memory system that they were using and we were reaching the limits of the capacity of the machine we had at the time so we needed a new computer system and we chose among several different possibilities the DEC PDP-10 as the hardware of choice.

Weber: What was the old computer?

Tomlinson: Well, when we started it was a Sigma Data Systems [computer] and they became Xerox.

Hendrie: Oh really, not the PDP-1 because there was a PDP-1.

Tomlinson: There was a PDP-1 there but I wasn't using it.

Weber: That was in the computer group, right?

Tomlinson: Well, there were two PDP-1's here, is that right? There was one in the same building I was in and it'd only support two or three users. It did do time-sharing but it was not the computer that everybody was using. That came here in roughly 1963, I think, and they developed a time-sharing system for it and it was being used. I didn't use it myself. We were using a Sigma Data Systems [SDS] 940. And we were on, you know, the Berkeley time-sharing system, TSS-1.85 I think it was, if I remember. And it was getting too small for our needs so we tried to buy a system that supported virtual memory in a way that we could use it for these LISP applications. And the main problem was that most of the virtual memory system used very large page sizes and the key thing with these LISP programs was they really liked little tiny pages and lots and lots of them. So we decided to go ahead and start with a PDP-10 computer but add our own virtual memory system on the memory bus and we designed the BBN pager to do that. It basically took the addresses as they came out of the KA10 processor, fielded those requests and then modified them and passed them onto the actual memories.

Hendrie: Was this a segmented system or just a page system?

Tomlinson: It was paged [system] so we had…

Hendrie: How big were the pages?

Tomlinson: The pages were 512 words. It had an associative memory to keep track of some of the ones that were actively being used and then a very quick way of reloading those registers to actually map the entire physical memory.

Hendrie: Okay, and you designed the hardware yourself?

Tomlinson: Yes. Jerry Burchfiel and I did most of the hardware work on that.

Tomlinson: It was actually a fairly quick effort. We decided to write an operating system from scratch that was called TENEX and it was going to be virtual memory so we started with this paging box. It took us about--Jerry's memory is probably better than mine on this--but probably a few months, not a really long time. And we put that in there and then we wrote an operating system for it. So I took off my hardware hat, set that aside and put on my software hat and started working on the monitor.

Dan Murphy and I did most of the monitor part of that and there was about a half a dozen other people that were working on other parts of the operating system, like the exec and the subsystems and so on. Other people did some work on the monitor too but Dan Murphy worked on, let me see if I get this right, the scheduler, memory management, the pager interface hardware, software and lots of other things and I'm probably leaving things out here. I worked on file system. The JSYS [instruction] support; JSYS was a Jump Into System Instruction that we used to get in and out of the operating system and all the higher level I/O stuff, the interface to the lower level drivers. And we worked on that again fairly quickly. I think we got the pager working in December and I don't remember which December that was. I could probably figure it out if I had a calendar but it was around March or May, when we had something running. It kept crashing and so on but then we used a few months to get to an operating system going.

And that was TENEX and then, of course, TENEX was there, [so] we had to do things with it. First, we got all our, you know, LISP stuff going and things that the researchers needed and we were waiting for this distant host interface so we could get hooked up to the network. And we finally got that connection and then we had to make a network control, the NCP, Network Control Protocol program, so that we could interact with the IMP, and [inaudible] wrote Telnet. I wrote a Telnet application so we could use the Telnet protocol. So we got these things going and I said, "You know, you've got to be able to transfer files." There has to be a way to transfer files. There was no official protocol out there. It was a work in progress for the file transfer protocol but I said, "Oh, we'll just put something together."

I worked on the file system interface to the network at the user level, the user application level. So when you open a file this way you open a connection in a similar way. So it was using the same system calls to open network connections as opening files. And so I wrote a little program to open a file here, open the other in the other place and send files back and forth and that was called Copy Net. And it was pretty simple-minded, you know. One, you put out a string, this is what the file name is that you're writing to and the other one intercepts that and in turn, opens the file at the other end and then it just streams the data through, and now we're getting close to the email part of this.

Weber: To back up a little bit. So NCP, you were working on that but I mean, that was really for ARPA right?

Tomlinson: Oh yeah.

Weber: So I mean, how would you have made the transition to being…

Tomlinson: Oh, I frankly don't know. It was a fairly fuzzy line to begin with. I mean, the computer side was working for the researchers and at some point--you know, I don't--I frankly, didn't pay attention. I tended to be fairly oblivious to all the politics and the organizational stuff.

Weber: Tell me if I'm right, I mean, we've interviewed Frank Heart and Dave Walden so I have this picture of this IMP group that was a very distinct unit.

Tomlinson: It was a quite distinct unit.

Weber: And what was your relationship with them?

Tomlinson: They were somebody to call when we had trouble with our network connection, for the most part. Yeah, they were over here, we were over there and, you know, we've seen each other in the lunch room maybe or something like that but the activities were, at that time, quite separate.

Weber: But the ARPA contract was with them.

Tomlinson: Oh yeah, yeah, the ARPANET-- contract to build the ARPANET was with the IMP group, Dave Walden and Frank Heart and those guys.

Weber: You were sort of doing this on your own initiative.

Hendrie: Who was supporting TENEX?

Tomlinson: Well, I mean, the research that we were doing, that the computer center was being used for, was also ARPA-sponsored but that was more on the application end of things. Write a LISP application to do some AI stuff or figure out other ways of where humans and machines can interact, some of the early natural language recognition. Type in a phrase and it analyzes it, figures out what it means and gives you some kind of an appropriate answer based on some data behind it. And initially we were running the computer that they used to do that kind of research but then as that framework developed and matured and didn't need a lot of effort anymore, the people, like myself, gradually started working on the projects that were being sponsored by ARPA in that area.

Hendrie: So this was overhead. Doing TENEX was sort of a weekend project.

Tomlinson: Yes, that was overhead, yeah.

Weber: And Jerry Burchfiel at that time, who was he working for?

Tomlinson: I don't know <laughs>.

Weber: Okay, but he was around and you were in close contact.

Tomlinson: His office was next to mine whether he was working for the computer center or Danny Bobrow or somebody like that, I don't know. It may not have been the very next office but the one when we were working on TENEX we were… I'd have to… as I say, I didn't tend to pay attention to who was doing what. At least some it probably was ARPA-sponsored. How much of it, I don't remember now. And so the computer center basically got to the point where we had some operators and they operated

the computers and, you know, things would break every once in a while and somebody would have to go fix some software or add some new software feature but most of the software was being developed as in conjunction with some aspect of the research that we were doing and that applied to things like Send Message, for example. That was a program that came with the Berkeley time-sharing system. It's called Send Message, that's where the name came from.

And when we moved our stuff to the PDP-10, that came with it. We said we need a version that's just like it so we can continue doing the same sorts of things we were doing before. So I rewrote that in machine language for the PDP-10 and so I had these two programs and, looking back at the RFC's it was in April that Dick Watson at SRI wrote an RFC proposing a way of sending messages across the network to print them on a line printer. Most of the high-speed printers were line printers or chain printers. They print a line at a time. And then taking the paper that resulted and filing it in a box so somebody could pick it up. And a lot of the protocol he was proposing had to do with how to format things on the printer, how to make a form feed and indentations and tabs and all that sort of thing. And when I looked at it, my reaction was, "Yeah, he wants you to be able to send messages to other places," and then I kept on thinking about it, I said "Yeah, and why not just send them to the people, you know, just none of this numbered mailbox stuff." Everybody had a login account on their computer so if you just knew what that account was, you should be able to send it to that person. At least on our system, we had Send Message and most of these machines had something similar to Send Message, if not the actual program. There were a number of derivatives of the same time-sharing system around. So my idea was just to put those two things together, the Copy Net to transfer the files because mailbox was just a file and Send Message so you could compose the message, give it an address, send it and most people didn't have any special-purpose reading software, they just typed it or printed it, depending on if they wanted to get it on hard copy--they'd put it on a printer. If they wanted to get it slowly, they'd use their Teletype. CRT displays were coming into common use but they were not all that prevalent in many places.

Weber: And with Send Message, it would work fine as long as both people were logged on to the same time-sharing system at the same time?

Tomlinson: Right, with Send Message they didn't have to be at the same time. There was always that… whatever the right auto iso-synchrony, auto something but not being synchronous .

Weber: Async[hronous?].

Tomlinson: Yeah, not quite, Async is almost the right work but it's also not quite correct. But you could leave a message and they'd come and read it when they logged in.

Tomlinson: And so all the needed things were there. We had Send Message to do the basic message composing and addressing. Anybody could just type it out on their terminal if they wanted to. It didn't need any special purpose reading software and Copy Net was an experimental version of File Transfer and it was a starting point to say if you started writing the file here, you put it into this program and it writes the file over there and the 'over there' is, you know, the role played by the '@' sign. It says it's the preposition that links the person with where he is and allows you to determine what post to send the message to. Well, this I was going to say in April 1971, was when the first RFC came out, there was a revision a couple of months later and during the summer I became aware of the RFC. I hadn't seen it when it first came out and I worked on it for a few weeks and we thought we had other things to do, so this was not my main… I was not challenged with saying ",You know, go do something with this RFC." I

just said, "We can do more than what this is suggesting here." So it took a few weeks because it was part-time but the overall software [INAUDIBLE] was, you know, not very much at all, probably maybe three or four days total over a couple of weeks.

Weber: Do you remember a moment when the ideas gelled?

Tomlinson: I think the idea gelled fairly quickly. There were details to work out, you know: what key on the keyboard should I use to separate the user from his host? But the rest of it was already sort of there so I didn't have to think about how I type in a message. I could always type in a message. There was already a place to type in the user's name. TENEX had names rather than numbers and some of the other login systems had, you know, numbers and various ways of designating who the users were.

Weber: Within TENEX, what would the message look like? You would just give the user name and that's all the information required.

Tomlinson: Right, yeah, the user name and the subject and the text for the message and a date if…I think the dates were filled in automatically…

Hendrie: So this wasn't very much code, I mean, like a hundred lines, a thousand lines?

Tomlinson: It was probably about 400 lines of code in addition to what was already there, right. They had to be fairly tight because they didn't have a lot of memory in those computers back then.

Hendrie: So this is assembly language.

Tomlinson: Yeah, it was assembly language.

Tomlinson: And that was sometime in late '71 when the first message actually made it from one computer to the [other]. We had two computers side by side. They were roughly the same hardware. One had more memory and it was used for developing the operating system and for testing out new versions and things like that so it wouldn't interfere too much with the other. And so and they were physically side by side and so you could just sit at a table with two terminals on it and type something here and look over there and go back and forth. There wasn't a lot of running around to do. There were a lot of false starts. With any program you, you know, you write some code, you test it, it doesn't work. And if it works, you go home and celebrate because that's very unusual. And there were a lot of false starts so when anybody asks me what was the first email message, I have to tell them that there's really no way to know because I would type whatever came to mind, usually you just, you know, dragging some fingers across, [the keyboard] you know, or if I was feeling really bored, I might say, "Well, how much of the Gettysburg Address can I remember?" And I'd say "Four score and I can't remember anymore, so." So there was a lot of, you know, and which one actually was the one that made it through in its entirety, I have no idea. A lot of them made it through in various parts. You know, you get the first line and then something will happen on the carriage return and the rest wouldn't be there. So then the next step in some sense was delivery of the next release of TENEX. At the time I think there were about a half a dozen sites that were running TENEX on the ARPANET.

Weber: I'm sure you've been asked this dozens of times but the '@' sign decision…

Tomlinson: Was a no-brainer. The @ sign was…, in retrospect I might have been friendlier to the Multics folks, in which as I recall the @ sign was the line delete character. So anytime they typed an @ sign the keyboard would lock up. They had these 2741 terminals that IBM made [and] the keyboard would lock and if they really meant @ sign, then they had to type it again, it would lock and then-- the locking was built into the hardware, into the terminal. They had programmed the controller the terminal was talking to to say, "Well, we want to actually get this @ sign through," so it would send back a signal to unlock the keyboard and then you could type another character and so if you really meant @ sign, you could type another @ sign or something like that. So it was really awkward for them to use @ sign so they found other ways of not having to enter an @ sign there.

But from my point of view, it was the one that made the most sense because it was a preposition. It's the only preposition on the keyboard. The other things are conjunctions of various kinds and I'm pretty sure there's no other @ sign unless you think an underscore is an @ sign. I mean, it says under so maybe it's sort of prepositional, I don't know. So anyway, it made sense. This person was at his host computer and it just made sense to me and it was right there, not even on the number row at the time. That's another mistake that many people make when they talk about that early event they go see what that they go to the '2' key and look at the @ sign there. Well, that's not where the @ sign was on the Model 33 Teletype.

Weber: Where was it?

Tomlinson: It was <laughs> you have to ask -- it was right next to the 'P'. It was right next to the 'P' on the letter row because if you look at the ASCII Code, which all these terminals used, the digits were all together and there's a correspondence between the physical layout of the keyboard and the character codes that they produce. So there are mechanical encoders in there [so that] when you press a key, that set these three bits and the column in some sense, sets the other four bits of the character code. You know, I'm probably getting it wrong, but roughly that's the way it worked. So to put the @ sign up in the digit row would have would break that pattern because the ASCII Code puts it right next to the 'A' and the [inaudible] needs to be in the letter row.

Email got its first exposure as part of a distribution of TENEX , probably in January of '72, or thereabouts.

Hendrie: How did TENEX spread to other sites?

Tomlinson: Well, that's a good question. The researchers here at BBN, of course, were interacting a lot with researchers in other institutions, particularly universities. University of California had three or four sites on the ARPANET, and the University of Utah had a site, and MIT had a site. So there were a lot of people with whom our researchers were collaborating anyway, and now they could collaborate more efficiently over the ARPANET. That was part of the whole ARPANET plan to begin with, was to get that kind of cooperation, and to find ways of sharing resources. Because DARPA didn't want to buy big computers for every one of their researchers, they were looking for ways of sharing those resources. And I believe to the extent that DARPA supported the TENEX development, they were interested in that as a possible operating system for their other researchers to use as well.

Hendrie: Okay, so they probably supported and encouraged the distribution.

Tomlinson: Yes.

Hendrie: Like BSD's. [Berkeley Software Distribution]

Tomlinson: Right, and so we formed a users group, and the interested parties got together, they purchased the BBN pager and got their own KA10s, and we would help them set it up and supply the software. So by then all that was under the auspices of DARPA.

Weber: By the way, Jennie Connolly gave us a lot of the TENEX documents, but if there's code around we're certainly--

Tomlinson: The only code I'm aware of is the code that Sabre does, at the Sabre site. And that's old. That's actually-- well it's not very old code. Sorry.

Weber: New, it's new.

Tomlinson: It's a more recent version of TENEX than… it was the version that went to DEC when DEC purchased the rights to TENEX; or thereabouts, that sort of timeframe, which would've been probably '73, '74.

Weber: Anything like printed source code or anything?

Tomlinson: Not that I can find. I can't even find the technical reports. We had quarterly reports with this DARPA contract, within which one of them I would expect some mention to have been made of email, and I can't find a copy of it. I believe our library was cleaned out at some point. Somebody said, "We're not in the history business here -- got to get current journals and use the space for some other way."

Weber: So following email, what was the next step?

Tomlinson: Well, there's a little more to the email story. In April '72 there was a meeting of the FTP working group, and in that meeting we started designing the commands to add to file transfer protocols to support email. And we ran it until SMTP came along, and Dave Crocker worked out the next stage of support for email in terms of protocols. For myself here at BBN, we continued to work on TENEX for a while. And then it would have been in 1974, or maybe late '73… well the meeting itself was in '74, and there was an operating system symposium in Hawaii, which I got to go to. We were there to present our stuff about TENEX and how we had done there. But at that meeting, Vint Cerf and Bob Kahn were presenting their TCP/IP paper, and I became fascinated with that. And I knew I was going to have to write TCP for TENEX at some point, and so I needed to know what it was all about.

And so we got involved in working on that and helped develop the protocol. I did write one little paper on selecting sequence numbers, which pointed out a couple of the deficiencies of the protocol as it stood, but actually my intention was to point out the problem that would occur if you had these long running conversations to connections in which sequence numbers would tend to wrap around and recycle. Because the 32-bit sequence number dimension is going to repeat. And because of the fact that the network itself can delay and hold and produce duplicates of the packets, it's possible for a packet to be delivered multiple times, and if you get an old one that comes in and it's confusable with a current one, it could cause mistakes. And that was the point of the paper. But it turned out that the three-way handshake was also an aspect of it that nobody had thought of. It turns out the sequence number

problem has been dealt with in a different way--and that's okay--but the other was getting the right solution; what I had wasn't perfect, it needed some improvements. So other people worked on that. I don't want to claim full responsibility for the three-way handshake but…

Weber: And that paper you submitted, when was that?

Tomlinson: It was probably in 1975, but I don't remember the exact details of it now.

Weber: Sorry, stepping back a bit to email, it did start to get adopted a lot, but when did that happen? When did you see the community start to…

Tomlinson: Well, it was adopted fairly quickly. It seemed to fill a niche that was needed, because at the time telephone answering machines existed but they were very uncommon. If you were lucky, the person you wanted to contact had a secretary who would take a message and then somebody would call you back. If you were calling somebody who got a lot of this, then they would have an answering service or something like that. But this, being able to communicate with somebody who was not ready to answer the telephone right then and there… there was nothing between that point in the spectrum and sending a letter by postal mail.

So it became an instant hit for those people who had to collaborate with others who were either, say, in a different time zone or just were not on the same schedule. The problem, of course, was the community of people with access to the ARPANET and computers was very limited. At the time that I'm speaking of (early '72), I think there were 28 nodes. And at each site there was maybe 50 or 100, at the most, usually often less than that, individuals who had access, and so you were talking about a community of 1,000, maybe 2,000 individuals that could communicate this way. So while it was very popular within that small community, it had no real global impact at all.

Weber: And within BBN, did you use it internally?

Tomlinson: Yes, it caught on fairly quickly. Not with everybody. There's always the Luddite in the group who feels the need to communicate face-to-face, as often as possible. And anybody who's done both will know that face to face is very, very effective communication. You get a lot more bandwidth in a face to face conversation. You waste a lot less time going down false trails or trying to clean up your English. But when the timescale is wrong, it doesn't work at all. And so it became an instant hit I think. And I think there's some early studies that show this, but although they tend to be a little later. Email itself as a term did not get coined probably for three or four years after this point.

Weber: So what did you call it?

Tomlinson: Messaging, message, mail; not email. But mostly you just sent a message to somebody.

Weber: But because time-sharing systems did have this internally, it felt very familiar.

Tomlinson: Oh yes.

Weber: It was just an extension of…

Tomlinson: Right. It was just a slightly larger community than what you had before then.

Weber: So it was a matter of degree more than [of] kind in the way it felt.

Tomlinson: Yes, it was more of a matter of degree than of kind. Why can't I do the same, what I can do with my co-worker in the next building, that I can do with somebody across town or across the country?

Weber: And the early messages, [was the] content similar to now?

Tomlinson: I think it's very similar to now. It didn't take very long before people started using it for non-business use: arrange meetings, arrange lunchtime get-togethers or announce… or business uses, like announcing conferences or seminars, or somebody's going to come in for a lunchtime talk, that sort of thing.

Weber: And was it easy to 'cc:' [copy'] a group of people at the beginning?

Tomlinson: Yes. Even the original program had a cc: list. The main thing that happened over time was that more efficient ways of distributing the multiple copies emerged. At the time you really had to send every message individually to all the recipients. We continue to send individual messages even when there's a group. It may go for a short ways to a mail list server, but from there they tend to get dispersed as individual copies to the recipients. And that's mainly because the bandwidth requirements--at least for text messages—[are] hardly noticeable on the network as a whole.

Even at the time it was a very low impact application. Which is another thing that contributed to its growth. Nobody thought twice about installing an email program or sending emails in terms of whatever kind of resources they were consuming. It was just miniscule.

Weber: Email later on could take quite a while to go across multiple networks. But in the beginning it was just all within the ARPANET, and that's it.

Tomlinson: Yes, the delays mostly had to do with just limiting the amount of polling and other activities that you did looking for things to do.

Weber: And within the ARPANET community, this got you a lot of recognition at the time; or not?

Tomlinson: Not really.

Weber: No?

Tomlinson: No, because… it was an interesting phenomenon because everybody decided that they could write an email reader. It was quite interesting to observe. But it didn't take much: the sender was pretty trivial anyway. Although three or four versions of different kinds of composition programs were

written, everybody thought they had a better way of presenting the information to the user for reading. And when you think about it, that makes sense. The composition phase is just another word processor kind of an application; a very simple one at that. But on the reading size, organizing your mail, once you've got a lot of it, becomes more of an issue than simply printing it as it comes on your terminal. If you get one message a day, it's no big deal; you print it, you're done. You got 200 messages a day, then it's more of an issue. So everybody thought they could write a better reader, and many of them were quite interesting and quite novel, and probably none of them survived.

Weber: In the '70s there were also these-- I know the group, Doug Engelbart's lab at SRI, didn't they do some readers and they composed lots of fields? There were some protocol discussions, right?

Tomlinson: There were discussions on standardizing the fields of the header. They occurred fairly early, and I think it was fairly quickly decided that there was half a dozen or so that pretty much had to be standard, because they were universal, and all the others tended to serve very special-purpose needs. And so the experimental ones were marked with Xs. And it got to the point where mostly you didn't have to understand the header. All you needed to know was that it was a header and you could show it. And you just didn't want to have 20 different ways of doing the same thing. So the standardization happened that way.

Weber: And did you get involved much with that?

Tomlinson: Not a lot, no. I said there's the subject, which I called ‘Re:’, which was adopting a memo format that I saw. ‘Re:’ was Latin for the word meaning ‘concerning’ or ‘about’, and the recipient, sender and the other copies that you were sending to the cc: list, those seemed to be fundamental to the whole thing, and everything else is… dates, well yes at some point you got to get a date because you want to know when things were sent, and that's about it. Everything else has to do with tracking threads so you know that this is in response to that one, and ‘Re:’ becomes ‘Subject:’ not a big deal. The weird one is the subject that says Re in it, which is one that always bugged me. Somebody said, "I'm going to call this thing Subject, but we're going to put the word Re in there also." And I never understood that one.

Weber: And the emoticons came in, they came in on Barrett's [ph?] systems, right?

Tomlinson: The emoticons, yes they-- I believe there's somebody who's believed to have sent the first MODE CON, but I don't know who it was or when it. I want to say it was more towards the '80s that that came about, but I'm not sure…

Weber: And the idea of attachments, when did that…?

Tomlinson: That happened when the content type, the MIME encoding stuff, was specified, and basically gave a way of dividing an email message into multiple parts and saying what the various parts were. Once that mechanism was in place, it became fairly trivial to do attachments. Simple file insertion happened quite early, but you had to sort of manually separate out the file from the stuff surrounding the file that told you what it was. And that was probably late-'70s sometime. I don't know exactly when. Maybe even mid-'70s. But it didn't take too long to get to that point where people were trying to use the mechanism.

Weber: Do you remember any particularly interesting early messages?

Tomlinson: No I don't.

Weber: And did you use it yourself personally very much?

Tomlinson: Oh yes. Yes. I think I've always been a more fundamental user of email than some others. Some people have used email fairly creatively in various ways. Myself, I like straight text. I don't like HTTP encoded or HTML encoded messages. But yes, I've always kept simple myself.

Weber: So then what was the next… after email…

Tomlinson: After email. Well let's see, we could be here all day; I don't really have all day. There was packet radio work. We had an interesting project for the Post Office. It was a project called INTELPOST. They had the idea of-- this was not email--but it had to do with sending messages. Their idea was to send a document by using a facsimile machine on the input side, route it with TCP messages to other post offices, presumably, where there would be another facsimile machine that would turn it back into paper, and then they would carry it to their postal carriers to the recipients, as a way of sending large documents, presumably less expensively and certainly more quickly than they could otherwise do.

Weber: And longer than telegrams.

Tomlinson: Longer than telegrams; sort of the 10, 12, 50 page document that one might want to send as maybe a contract or something of that sort. And they did have the notion of sending it to multiple recipients. So if you wanted to send 10 copies of the same thing, they could route it out, distribute it to 10 different places or five different places with two copies each, and then deliver it within the local area as ordinary mail. I don't know where it went from there. The whole idea was that facsimile machines were very expensive, at the time, and that you could get some synergy here by combining it with network capabilities. I think what happened is that facsimile machines became much less expensive and everybody did it for themself.

There was writing TCP for TENEX that slipped in there at some point.

Weber: Oh yes.

Tomlinson: We wrote an implementation of TCP. Yes, we incorporated the TCP into the TENEX monitor. It worked much the same way as the NCP did. You open connections and send the traffic. Bill Plummer and I worked on this. I think he did most of the work. But yes, we were using it for connecting to a line printer. We had a line printer on one machine and we sent… it was actually a test vehicle more than anything. We needed to print things on this printer and it became a way of testing the TCP protocol to see how it'd break when you actually used it for something. Two different machines that tended to have their own idiosyncratic failures and things like when they come back up and it looks like you're continuing the same conversation, because the addresses involved are the same, and we found that there was a real reason for having the three-way handshake. We would start it up and it would think it was continuing with the old message, and didn't realize the other host had gone down.

Things like that. So that was interesting. And we did some work on convincing the Department of Defense that the IP protocol was the thing that they should adopt for internetworking. That would've been in the late-'70s.

Weber: Following the three-network experiment.

Tomlinson: Right, yes. And so packet radio was in there. We worked on so-called station, which had a lot to do with setting up routing within the individual packet radios. We also worked on some security aspects of… I think, was that packet radio? Boy, it's fading into oblivion here. We need a history museum or something to keep track of these things. But we did some work on doing secure communication, and that was probably packet radio, because I remember Rockwell, or Collins Radio…

Weber: Rockwell Collins yes.

Tomlinson: …and then Rockwell was working on that unit. So I suspect it must've been associated with packet radio. Then about this time Xerox, the Palo Alto Research Center of Xerox, who had been been stealing our employees for a while, were doing a very nice job of developing a business and a way of doing research. And one of the things that came out of their activity was the Alto workstation; a desktop unit that had bitmapped graphics and a nice computation engine. And we were jealous, or envious, of what they had, and [since] they weren't in the position to be able to sell these, and nobody else was making them yet, we decided to go ahead and do our own.

So we designed Jericho, which is a workstation that we developed here. It's a bit-slice architecture with a built-in disk and bitmapped graphics. And it turned out [to be] a little larger than we would have liked; just sort of a file cabinet-size box. And I worked on that for a while and got the hardware working, and then worked on some Pascal microcode for it, and then started doing some multimedia development. And we developed… it's called JADE, but it was all written in Pascal. And this got BBN started in this multimedia computing business, that we continued under various names over the years.

Weber: And you did your own operating system for that?

Tomlinson: We did our own operating system for that, yes. Let's see, how did that work? It directly interpreted Pascal or LISP; there are two versions of the microcode, one for Pascal and one for LISP. And so you're pretty much running on your own computer, with more like subroutines than a protected mode kernel to keep things going.

The Pascal was compiled into p-code; which I forget exactly who developed p-code. I think it was one of the West Coast… University of California, Berkeley, UCSB; I don't know, someplace like that. And we modified it just a little bit, but it was just a way of representing the Pascal code in bytes, one byte for this, a five-byte address or something. And then the microcode interpreted that and ran the machine, right. There were some fairly complex instructions.

Hendrie: How many people were working on the hardware?

Tomlinson: About four.

Hendrie: Okay.

Weber: Who were the main ones?

Tomlinson: Myself, Jim Calvin. Oh, I'm blanking out his name; Jim MillerI think it was Jim Miller. Bob Clements. Alan Bell actually started the whole thing. He started it on this project and then I think he went elsewhere; I think he went to PARC. He was one of the persons most frustrated by not being able to get… he started the project and then he decided to change jobs. I think those are the four that come to mind. There were others who contributed from time to time, and I'm probably leaving somebody out. I'm sorry, whoever that somebody is.

Weber: And do you remember the year roughly?

Tomlinson: It was around 1980. It was just early enough that Sun hadn't come out with their workstation yet.

Hendrie: And had Apollo come out with theirs yet?

Tomlinson: I think they had their workstation. I don't remember. It was that same era. And FiberNet had one of the first fiber optic networks, that we had running in Building Six.

Hendrie: Did you build a lot of these machines?

Tomlinson: I think there were about 30; maybe 25, 30, something in that vicinity. Well, I think there were slightly more of them that were running LISP than Pascal. They made a very nice LISP pack. That was l-code. Again, it was all the main operation; CONS and CDR and all that were implemented in the microcode, so you weren't using full machine instructions for doing those things. It was sort of built in to the way it worked. Had a good-sized disk, a very weighty disk that held the whole machine down.

Weber: And what sort of processor?

Tomlinson: I think it was the AMD 2910. I think 2610 was the microcode controller and the 2901 was the ALU, the Arithmetic Logic Unit.

Weber: And do any of these still exist?

Tomlinson: I believe there is one someplace.

I had a CPU board up until recently. I don't know if I still have it or not. But I believe there was [one] around, and Alex McKenzie might know about it, but I wouldn't know how to put my hands on it.

Weber: And who worked on the software? You, but then who else?

Tomlinson: Well once the software was there, then a lot of people used it for their ordinary research; they wrote their programs and instead of using TENEX, they would have their own computer.

Hendrie: Well it's a lot more fun than a time-sharing system.

Tomlinson: Yes, exactly. A couple of dozen people might have had these machines for their use. Because they were personal, it would've been very awkward to share them. I don't recall if there was any access control on them or not. There may have been, but I don't remember.

Weber: It was a big monitor like the Alto or the--

Tomlinson: Yes. Well, the video hardware was fairly adaptable. It could run an ordinary 640 x 480 color display. But it also operated larger pixel count, black and white displays, and you could basically program the refresh and sweep rates and all that. So it could in principle accommodate almost any monitor. We actually used it probably for about three different types of monitors.

Weber: And mouse?

Tomlinson: Had a mouse, and a keyboard.

Weber: No chorded keyboard though?

Tomlinson: No, no. That somehow never caught on, on the East Coast.

Weber: It didn't catch on so well in the West. But it is on the Altos.

Tomlinson: Yes. Had a good-sized disk. Had this Fibernet interface. It had a serial interface, so you could hook up an ordinary terminal to it if the keyboard wasn't working or something like that. Actually no, the keyboard had an RS232 interface, we just didn't use the output side. You could. It was sufficiently general that you could hook up a terminal to it so before you got the video going you could debug it. It had test microcode. You had to pop out the PROMS to change the microcode. It couldn't be coded in place, you had to burn the PROMS offline and then plug them in; which is probably how most of them broke. You unplug those PROMS enough times…

Weber: Are there any pictures of these do you think?

Tomlinson: Oh there probably are, yes. I have in my mind a picture that I saw recently, but I can't remember where I saw it. So we did that, and we got to use them for about two years, and then Sun came out with the 100… what do they call it? The model 100 and the 150 workstations. And we got a couple of those, and we just kept getting more and more. I think the LISP folks continued to use Jericho longer than the Pascal guys did. But we started working, taking these jobs we were doing and coding using the Sun workstation for them instead.

And let's see. The next thing we worked on, which is something that never really saw the light of day but it was a lot of fun doing. We got the idea that we could design our own VLSI circuits. This was when DARPA was running a foundry, and you basically could put together the designs and ship them off to the foundry and then just get your chips back. And we decided to carry that to an extreme where we started working on a design for a 65,000 processor supercomputer. And this was going to be a serial computer rather than a parallel computer. So all the computation was going to be serial; at least all the intercommunication was going to be serial between the memories and the processors. And the reason we felt this was a good idea [was that] one of the issues you have with a large array of computers like that is keeping their clocks synchronized. You can either make them be asynchronous, intrinsically asynchronous, and then continually resynchronize them when they have to talk with each other; but we thought we could come up with a design in which we could tune the synchrony of all the clocks if you only had one bit per CPU to worry about, one wire. Then we started working on the LSI design for this, and we had a circuit… I don't know if we got a patent on it or not. We were thinking of patenting it, [in] which you basically could tune the delay electronically and it could sense the delay. It was sort of like a phase-locked loop, it was similar in concept to that.

And get them all running in sync, and then they would do all the arbitration from memory access and the streams would flow together and it would just work. And we came up with a design. The only problem was it was physically going to be a monstrosity. We worked out the area that it was going to take, and it was going to take a room where it was 40 x 40 feet, had lots of cooling machines around the periphery, and could not have any columns because [of] all the interconnects between them; we had to go sort of in the middle. And it was about an eight-foot thick mat of wires, and I don't know, about 2 megawatts of power. And it became clear that this was probably not going to go anywhere. So it was very interesting and we worked up a lot of very cool parallel processing algorithms and techniques in the course of doing this. Because obviously if you're going to do this parallel processing thing, you better come up with algorithms that let you actually take advantage of all those 65,000 processors. And I think at the time there was this benchmark of about 24 problems that you'd like to be scalable. Will Crowther was largely responsible for this. He came up with ways of dealing with each one in which you could see the linear speedup with a number of processors over all these problems. And you were never at a loss of what to do with some of your processors. Very cool stuff.

Tomlinson: That was mid-'80s, probably '86, '87, something like that.

Weber: The Connection Machine was roughly in that period.

Tomlinson: Yes.

Weber: Were you aware of that? Thinking Machines [Corporation] weren't too far away, were they?

Tomlinson: Let's see, I was aware of it because one of the people who worked on our project went and worked for them.

Weber: Okay, there's somebody else. And going, just stepping back a bit to the packet radio era, I've interviewed people at SRI and Collins Radio. But the role of BBN in that, what chunk…?

Tomlinson: Yes, so our role in the packet radio project was to provide some of the higher-level control functions of the network. It was called the station. I'm not sure where the name came from, but the radios, the packet radios themselves, were relatively autonomous in the sense that they needed to be, because they weren't physically connected to anything else. But they had to have a bootstrapping process where they could get themselves on the air and receive their initial information, whatever they needed to organize into a network. At the time, a completely autonomous solution didn't seem to be the right thing. We're doing some work today in which these things are almost entirely autonomous, in the sense that there is no centralized control. Because centralized control, of course, is a potential Achilles heel in such a system. But we knew how to do this with a central control. So we decided to do it that way. I think that was the overall thinking. They'd be autonomous to the extent they could get themselves working, but to work well they needed some guidance from the station. In fact, what happened is we tended to get into almost everything, because you couldn't really work on this stuff in isolation, and so we got involved in other aspects of the packet radio development too. But that was our nominal role.

Weber: And so you were going to all those meetings.

Tomlinson: Yes.

Weber: And there was the packet radio newsletter type thing.

Tomlinson: Right.

Weber: And then later sort of military trials with that. Were you involved with those?

Tomlinson: No, I think we got out of that business. There was another phase of packet radio development here, which I was not involved in myself. So I can't really say much about that. When our project terminated, we had it to the point where we had radio. We could do networks. We could gateway into the Internet and drive trucks around San Francisco or wherever they were, or Palo Alto, wherever they were doing the testing on the West Coast, and then it all pretty much worked. Then it presumably got deployed, or at least developed further, to the point where it might be deployed. And that's a long time ago, and I don't remember very much about that.

Weber: But you remember the '70s, the two- and three-network experiments though?

Tomlinson: Yes.

Weber: And you were playing a role in those?

Tomlinson: Let's see. Well, I did to the extent that I actually wrote a gateway or router at some point. Yes, this had to be in the early days of the packet radio effort, because we were using the [DEC] LSI-11s, and I spent some time trying to get a very, very efficient router code written. And of course it worked really well, and others came along and said, "Well, but it has to do all these other things." And I said, "Oh okay, well…" And then they took it over and made it be slow again. That's my point of view. I think it's one of those problems that if you do the minimal thing it works really, really well, in terms of efficiency and what it does. It just doesn't support the things you need for administrative purposes or access control or privacy or any of the things that actually complicate many aspects of modern life.

Weber: And so after the parallel processing work, what was…?

Tomlinson: I started working on some of the multimedia work that we were doing. I had been working on it before and I went off to do the LSI stuff, and came back. Nothing really stands out in my mind as to what we did. We did some video conferencing. We did some multimedia conferencing. These were event-driven conferencing techniques rather than…there are many ways. The idea was to get the same application in front of multiple users and distributed across the net. And there are many ways to do that. Today most of what you do is you have it operating one place and you send screen images around. But that was not feasible with the networks at the time. So we were working on a scheme whereby you would do the same computation everywhere and send the event stream that was driving the computation. So you were basically replicating the computation in multiple places by taking the event stream and getting the canonical event stream, and making sure everybody played that stream. And it works as long as everything stays up, but if something dies, you can't get it back into sync. So it had a place but it also had its limitations. But there were other things too, having to do with ways of processing video. We had one of the early equivalents of a webcam. I don't know if anybody's mentioned it. For a while there on the fourth floor of Building Six over there, there was a video camera sitting in the window, broadcasting on the net.

Weber: When?

Tomlinson: That would've been probably roughly 1990, '91, something like that.

Weber: What tool would you use to access it?

Tomlinson: It was something we called PicWin, for ‘picture’ and ‘window.’ And we had a modest effort to commercialize that, but it didn't go anywhere, and I didn't have much to do with it myself at all, but I was there and I saw it. It was looking out the window at the intersection there where the traffic light is, and every once in a while we'd catch an accident. When you got really bored, you could watch it to see if anybody was going to have a near collision at the light.

Let's see, that was probably slightly before 1990, because about the time of Desert Storm, somebody, part of BBN, was off trying to sell a program to DARPA, telling them about some capabilities we had and some things we wanted to do. And it was working on a kind of experimental test bed for doing a certain kind of research, and we thought we could build this test bed and then various universities and so on could contribute their parts to it. But somebody, not myself, was down in DARPA talking with them. And this was about the time that, I guess, Iraq moved some troops into or near to Kuwait, and so it became sort of a crisis, and the person we were talking to said, "We're going to need something like this."

And it had to do with logistics and the ability to quickly make logistical decisions and rearrange your transportation schedules and do things. S o we embarked on this very rapid effort to try to bring this very experimental, demo software really, with lots of holes. You say, "Well imagine that this would happen now, and then we'll do this" kinds of explanations, and bring that to the point where it was actually could be used. And so I was only peripherally involved. I was not involved at all, except that we needed some people to work on it right away, and they thought I could help. And I probably helped in the sense of not getting in the way too much, because others did most of the work. But we put together this tool very quickly and took it over to Germany where they were doing a lot of the logistical staging for that activity.

The only problem was this was very experimental software and so it had to be handheld. It had to be watched 24 hours a day practically. And it kept breaking and somebody would have to restart it and redo it. And we were mostly running this in parallel with the standard logistical tools that they were using. So it was kind of a backup. We had some things we could do, like some 'what-if' kinds of experiments so that if you had this plan, so what-if we do this? We could do this relatively quickly with our software, which is something they couldn't do with the standard software they were using. But it also tended to verify that their plan was good. And we did save there them about three or four days worth of work at one point when some of their stuff crashed and they lost a day's worth of planning. And we happened to have a backup of it. So we restored from our stuff. But I did some babysitting over there myself, and that was my introduction to databases.

Weber: Was that a good thing?

Tomlinson: They were using a database, and I'd never seen a database in my life. For one reason or another, I never had the opportunity to think about databases. One Sunday morning, I was in there with nobody around and it wasn't working. And all I could see on the screen was there was some corrupted rollback segment. And they were corrupting that. It didn't sound good. It was midday here in the 'States, so I could call somebody, and got some advice. I just ended up doing a restore from a full dump. It was an easy fix. I didn't really have to understand databases to fix it. But it was an interesting experience nonetheless.

Weber: Did it get you interested in databases or no?

Tomlinson: Yes, yes. I actually taught something, somebody about a database today; yesterday actually. They didn't know what DDL and DML were. And they knew what SQL was, and I said, "Well there you go. DSQL. You go DDL plus DML." Almost exactly.

Weber: Okay.

Tomlinson: SQL has two parts. It has the data manipulation language the data definition language. Anyway.

Weber: What was the name of the military logistics project?

Tomlinson: We called it DART.It stood for something, and I probably can't remember it. [Dynamic Analysis and Replanning Tool].

Weber: And I've seen in some of the boxes we were going through references to a video teleconferencing system. That's what you're talking about, right? Earlier. Right.

Tomlinson: I think they were trying to sell it. I'm not sure how hard they were trying to sell it but…

Weber: Okay. Do you remember the name of that?

Tomlinson: Slate?

Weber: Yes. But it was not just video, it was also groupware.

Tomlinson: No, it was not just video, it was groupware. It was spreadsheets and word processing, things like that.

Weber: And it was networked over the internet then?

Tomlinson: Yes.

Weber: But that got out of here, or not really?

Tomlinson: Its hard to say what happened to it. It just sort of withered away. BBN has historically not had a lot of success in marketing our products. We're having more success these days, a lot more success. But back then the mindset was wrong. It takes a different kind of people, a different class of individuals, to develop some snazzy idea and to actually turn that snazzy idea into a marketable product, and that can be supported and good profit margins and things of that sort. It's almost like you have to have two different companies to do that, or at least two different groups within the company, and it's very seldom that you find somebody who's good at both.

Weber: When did Slate come out?

Tomlinson: It would have been roughly that same period, 1990; maybe a little later, maybe a little earlier. Probably a little later.

Weber: '91, '92.

Tomlinson: Yes.

Weber: And after Desert Storm?

Tomlinson: Oh let's see. We came back, and I think I continued working on some of this multimedia stuff for a while, and then… what would've been next? Probably Pathfinder. Pathfinder was a wearable personal support system. So it allowed you to communicate, to monitor your condition, at least in principle. It had GPS, so you could tell where you were. It had radio, so you could communicate with others and they could know where you were. It had a map base so that you could see where you were on a map. And although we never actually developed it, you hook up to any kind of sensors you might be wearing. So in principle if you were injured, or if you just fell down, you could tell it to tell somebody about you. The idea was this was something that a small, platoon sized group might use for an operation. And we were working with some early tablet-like computers that…We looked at a number of potential things. We were prototyping with PC 10 [form factor]. There's a small square form factor board for what you can do; have PC components. Intel 486 type stuff. And they were prototyping with that. We were trying to find somebody who could build us an actual ruggedized computer; probably a form factor about like this. Most of our testing was done on the desktop. So we did a lot of testing that way, but we did have a couple of early notebooks or laptops. I don't remember who made them.

Weber: You had them made?

Tomlinson: We had them made. These were not the ruggedized ones, but we could carry them around and look at the map and…

Weber: Did you have any prototypes made for…

Tomlinson: We had one prototype of what we thought it might look like. I don't think it was functional but it had the case and the screen. And it might've been functional. We only had one though, so we couldn't really…

Weber: And then after that?

Tomlinson: Well, we worked on the logistics anchor desk, which was a way of getting a situational awareness kind of display of various logistics issues. We got data feeds, mostly from databases, such as I talked of before. There's a more or less standard planning format that's used called Time Phased… TPFDD, I forget exactly what it stands for. [Time Phased Force Deployment Data]. But it's basically a line by line listing of, in varying levels of detail, all the things that need to be moved logistically, from personnel to bulk things like oil and water and tanks and treads and repair parts and all nine classes of supply. And this is a way of bringing all that information together into a single display, so you could figure out what your logistics situation was; how much stuff that you have here on hand, and so on.

And that sort of segued into what became Cougar. But I'm missing the intermediate name. ALP. These are a sequence of agent-based planning tools to assist with the logistics problem. And the idea here with using agent-based technology was to distribute the computing to places that were relatively close to, say, a supply depot or a unit in the field, so that you could keep track of the local situation, have some computing power there to know what your current situation was, and then interact with agents elsewhere and arrange for transportation.

For example, you have a supply depot here, you need supplies there, you need to arrange the transportation, do the scheduling, come up with a plan for what to do, and then start watching the plan as it executes. Because every plan changes. No plan is accurate once it starts executing. Re-plan; and that was the whole overall notion. We were working on the agents for this in the infrastructure, the software framework for them to run in. It was a multi-vendor collaboration. We were mostly concentrated on the infrastructure to support various planning pieces; so other companies were coming up with algorithms for doing this and that. It evolved to the point where we were in fact doing a significant share of that. Some of the people either dropped out or decided to go elsewhere with what they were doing. And as I say, there were, well there were two programs. One was ALP and one was UltraLog. The agent system we were working with came to be called Cougar. Cougar's still around. It's open-source. And that lasted for a number of years. It was two fairly good- sized programs that we went through there. And at the end of UltraLog, these things--

Weber: Which was when roughly?

Tomlinson: Let's see, when would UltraLog have finished? It wasn't too long ago. It was probably 2000, 2001, something like that. Then what? We're getting to the hard stuff to remember, all the current things.

Weber: So at the time you created email, did you anticipate how big it would become?

Tomlinson: Well no, not really. The problem is that while I felt that everybody who had access to the network and a computer would have no reason not to use it because it was clearly very useful. The fact is that at the time there was a very small community of people who had that access, maybe 1,000 individuals, something of that sort. And so while you could communicate with email with these 1,000 people, there were millions of other people that you could not communicate with. And so to some extent my vision was limited by the size of that community, and I said, "Everybody's going to use it"; but only 1,000 people [did].

And what it really took for it to become as universal as it has is the growth of networking in general and the availability of computers. And in some sense one drove the other. There were people who wanted to use applications like email, and for them buying a computer and hooking it up with a modem to a network of some kind was an almost essential thing for them to do in terms of their business or their approach to how to run their life. And that, of course, led to demand for lower-priced computers, more accessible networking. And the fact that then these people were connecting up meant that the community with which you could communicate became that much larger. So instead of 1,000 people you might have 10,000 or 100,000 or 1,000,000. And as that grew it was this regenerative effect in which the demand for computers and networks drove their price down, because once they existed they became cookie-cutter products that could be turned out with mass production techniques, and so it just sort of went around in an ever expanding spiral to encompass more and more people with the ability to use email.

Weber: And then after 2001 or so, any notable projects you want to talk about?

Tomlinson: Well, the one that stands out is the little bit of work I did in mobile networking, which you said is one of your interests, as well as computers in general. We have a project we've been working on now for a couple of years of a self-organizing mobile network using small radios, that by virtue of their large number and their low cost means they can have a large number of them. Every individual can have one or two sitting around in their pockets or on the table. And the number of these allows them to organize themselves into a reliable network that does not require a lot of advanced planning. It's all radio-based. And using some of the delay tolerant techniques that have been developed in the last few years, you can even go out of range of other radios for a while and come back and continue communication. So it's a very interesting project that I've worked on recently.

And we're doing some work now in the healthcare arena where we're trying to gather together information on medical records in an anonymized way so that you can try to gather statistics about how things are working, whether treatments are working well, and follow a patient from what observations were made, what treatments he received, what the outcome was and to try to say, "Well these observations here and these treatments, you're getting much better outcomes than if you go use these other treatments." [We] try to analyze this data in a way that does not presume any kind of causality. You don't say that getting your aspirin within one day of presenting with an MI [myocardial infarct] cardiac problem is good or bad. Right now it's assumed that it is good to do this, because it has good effects, but you don't necessarily

know that in advance, and you get some accidental trials by somebody [who] comes in and they do this. It may not have been important for what they were there for, but they had other conditions.

Weber: But using data collected by ordinary physicians or by researchers?

Tomlinson: We're partnering with some people who have have gotten into this business. So we have some data that we can de-identify and use.

Weber: And particularly when you came in the '60s, there was a certain tension between military research and certainly the anti-war movement, and I presume the lifestyles of a lot of the people that worked here. Did that play out within BBN?

Tomlinson: Yes, sure. There have always been people here who would choose to not work on projects sponsored by the Department of Defense. Most of what we've done has not been very directly associated with the Defense Department. DARPA itself has changed from being DARPA to ARPA to DARPA to ARPA a couple of times, depending whether they're trying to emphasize the advanced nature of what they do or the defense nature of what they do. But most of that research is pretty general purpose, in the sense that it applies to a large range of possible problems. It happens to be sponsored by the Defense Department and it certainly has defense applications for the most part; although sometimes you wonder when you look at some of the projects.

But most people can find a place within that spectrum where they're comfortable, and those that don't, they find something else. They f work in the Education Group, which we had for a while, which is now somewhat diminished. Or they can find some commercial projects, or they can go elsewhere. And that's happened on a number of occasions. And certainly in the '60s and '70s, in the Vietnam era, there certainly wasn't very much tension. We were viewed by outsiders as having a stronger part, a larger role in the defense industry than I think the people here thought we did. Because as I say, most of what we were working on was very general purpose, it all had incidental military significance. But we were sponsored by the Department of Defense and that's what tended to attract interest. We did have a couple of occasions in which there was perceived to be some kind of a threat that really never materialized.

Weber: But a lot of the people here looked like students or hippies or ordinary people at the time-- correct?

Tomlinson: Well there's always been a broad spectrum of people here, and it was true then and it's true now. There's the people who come in without shoes-- they may put on shoes on a really, really cold day and look like they just got out of bed and they haven't groomed in any particular way before they came here. And there are those you never see without a suit. And depending upon what their job is, it doesn't really matter. Obviously in some situations where you are in fact interacting with a client, and that client has their own expectations of how they're going to be treated as a client, they might be upset at being approached by such an individual. But for the most part those issues just don't really exist.

Weber: Okay. Because partly I'm thinking of things I've heard at SRI and some real culture clashes and wondering if those sorts of things… it doesn't sound like it was…

Tomlinson: I have to admit that I'm probably relatively immune to such things. I tend not to notice them a lot. So I'm probably not the best spokesman for that aspect of things here.

Weber: Anything else you would like to say?

Tomlinson: No, I think I've… I've enjoyed this visit. I hope to come up with some material that will be instrumental in keeping your endeavor going and help other people understand what computers are about, and what they were about, and what they will be about. I think it's very exciting to have this interest being shown in these kinds of activities.

Weber: Great. Well thank you. And any advice you would give to young people interested in computing?

Tomlinson: Well you have to be interested. I think it's very hard to do anything with computers if you're not interested. You need to remember that you don't really have to go looking for problems; there are a lot of problems out there, they will find you. And what you need to know is how to figure out what it is about those problems that you know how to deal with. Know which parts of those problems have been dealt with by others. You need to do research. You need to know what else has already happened so you don't repeat the same mistakes. And if you've got that sense of what you can do, what has been done, and the problem is what needs to be solved, you can figure out what the difference is, how much more has to be done. And that's what you need to work on, when you want to work with computers, or anything in general even. Computers are not particular and not unique in that regard. They're just… they're cool.

Weber: Thank you.

END OF THE INTERVIEW

