Oral History of Malcolm Douglas (Doug) McIlroy
Part 2 of 2

Interviewed by:
David C. Brock

Audio Recorded November 8, 2019
Etna, NH

CHM Reference number: X9183.2020

Brock: Okay, great. Well, thanks, Doug, for continuing the interview. I appreciate it a lot. Well, as you saw from the last transcript, we finished at just about the beginnings of the Unix story. And I even wanted to begin with a question of maybe taking even one step back from there. And what I was wondering about was after that NATO software engineering conference, where you introduced your ideas about software components, and then the subsequent sort of abrupt departure of Bell Labs from Multics, how were you personally thinking about what you might work on next? After those two things happened?

McIlroy: I have no particular recollection of choosing a definite direction to go in. There were always small things to work on. And in fact, I think that perhaps describes the way I worked at Bell Labs anyways. I did a whole lot of little things, and very few big things. And these might be a math problem one day, a small program another day. And in the inter-- I certainly in the interim between the pullout from Multics and the advent of Unix, I certainly was writing programs because I remember with some distaste using the GE-635, and its operating system. And what programs-- but at this remove, I can't specifically remember what I was programming at the time. One thing would have been ROFF.

Brock: Could you talk about ROFF and why you were working on that, and--

McIlroy: Okay, ROFF, of course, is a descendent of Jerry [Jerome H.] Saltzer's RUNOFF. And we certainly needed one for Multics. And my recollection is-- and it turns out everybody has a different recollection on this-- is that Bob Morris and "Person X," whom I don't remember simply made a RUNOFF, I think, in MAD [Michigan Algorithm Decoder], which was the programming language at MIT. But MAD didn't run on a 635. Or it didn't even run on Multics. So for some reason, I thought it would be fun to do it again in BCPL.

Brock: Okay.

McIlroy: The name ROFF was applied, I believe, by Bob Morris, because there would be simultaneously on CTSS, the old RUNOFF, and the new program. So ROFF was just a contraction. And I called mine, which is written in BCPL, ROFF, only I went-- I augmented the Saltzer model. Introduced number registers and adding to num-- and some arithmetic on number registers, and a few more commands, and particularly header and footer things. And that program persisted around Bell Labs for quite a long time. It ran on a 635, I'm sure, until it was replaced. And then that got transliterated again into machine language by [Dennis] Ritchie and [Ken] Thompson for the PDP-7. And once again for the PDP-11. I do not recall whether they did it in machine language for the PDP-11 or in B, which would have been a rather easy modification of the BCPL. And then at the same time, Joe Ossanna was pursuing the goal of having a word processing system-- which was the fig leaf under which we got funding for the PDP-11-- was working on NROFF.

Brock: And what would you say is the distinguishing characteristic between ROFF and NROFF?

McIlroy: NROFF has-- gives-- well, the most important one to me is that NROFF had macros.

Brock: Ah.

McIlroy: Subsequently having seen Joe do it on-- I put macros into the ROFF that was running in the comp. center as well, but Joe gets the credit for putting it into one of these things in the first place. And Joe introduced the whole vocabulary of "escape sequences." Which generalized when you came to TROFF, the kinds of escape sequences you had in ROFF were-- there was half-spacing up and down available on some of the output devices. I had no way to talk about that. And then it-- then when you go to font changes and size changes, all of that just fit right into the syntax that Joe already had in NROFF.

Brock: I see. And some of the other text editors that were around at the time, QED and TECO, these also had a kind of macros, didn't they?

McIlroy: QED didn't.

Brock: Oh, it did not. Hm.

McIlroy: Well, at least-- QED, as Ken implemented it, did not. He was building on something that he'd seen at Berkeley. But I'm not aware that that had macros.

Brock: Okay, I'm not sure either. Huh! And was this-- was your work on ROFF, was this the first time that you had gotten into, by necessity, the details of the whole world of page layout and typesetting and this typographical world, or was it an easy--

McIlroy: I hadn't done any of that before. But at the ROFF level, there's not much to be done. <laughter> A little bit of indenting, a little bit of vertical spacing, a little bit of centering, and that's it.

Brock: And would you say that the reasons why there was the necessity for having ROFF, for having RUNOFF, you know, to create literary documents, rather than maybe program documents, if you will? Or program texts. Is this the-- I guess I'm wondering about this push to make the computer a writing tool for writing beyond software.

McIlroy: What-- the first things like RUNOFF, were first used for writing manuals. Same thing is true of ROFF.

Brock: Okay, so the first is?

Brock: And because manuals had a-- and perhaps that's where the idea for macros came in, because the manuals had a stereotype form, so you wanted to put in these bits of boilerplate that the macros could do for you. And I'm not-- oh, and I think that Joe clearly had at the outset, the idea that somebody was going to come along and make a higher level package on top of the low level ROFF for making document layouts for publication. And often the drafts will be typewritten. Publication will be typeset. And the measurements are going to change between the two. If you'll hide that in a macro package, so you'll have a version of the macro package that does typewritten. And the same macros for the particular format of the journal, or if you want to publish it in several places, you can make the changes just in the macro definitions and not at every place, every paragraph where, which is affected by your style sheet choice. So one of the macros he definitely had in mind was the idea of style sheets.

Brock: Right, which makes it more along the lines of almost like the markup language approach to the world. Yeah, yeah.

McIlroy: Well, it is a markup language.

Brock: Yeah, I guess so! <laughs> Not just like it, but it. <laughs>

McIlroy: I don't think we used that word in those days, but it is.

Brock: Yeah.

McIlroy: Nowadays it's markup versus WYSIWYG.

Brock: Would you say that-- so as you worked on that effort in this period kind of right after the Multics effort was-- Bell Labs withdrew from it, were you at all concerned about how that departure from the Multics effort was going to affect the research programs of the other people in your group?

McIlroy: Well, yes, because there were several who really wanted to keep working on operating systems, even though I think most of them felt freed. So we had this-- particularly Joe Ossanna would have been the ringleader, Ken Thompson was right in there. This push to try to get some machines on which we could build an operating system. That started right off.

Brock: That was the immediate response.

McIlroy: And the first proposal that Ossanna came up with was to get a PDP-10. Well, this story has been told so often.

Brock: Well, please.

McIlroy: And higher management, once burned, said, "We don't do operating systems anymore." That did not deter our guys.

Brock: Was that one person in particular? Or was it a group of people who--

McIlroy: I think it probably went from Sam [Samuel] Morgan up through the Vice President at least.

Brock: Okay. And so there's the essential tension of how to continue to pursue work on operating systems.

McIlroy: Yes. And then came the fig leaf. The idea that clearly word processing by then was a coming thing. And it would clearly be taken up in the typing pools, for example, if not every secretary's office. So the rationale for getting the PDP-11 quite explicitly said that we would develop a word processing system. I think it was believed that we would make a standalone word processing system-- oh, but to do that, we would first make an operating system, so that we would be able to program the machine easily. But I suspect I don't think we ever said this, but I believe higher management believed that we would make some standalone-- we would have the operating system in our lab, but it would support a freestanding word processing system that could be distributed here and there in the Labs. There is a story afoot, and I don't think it's quite true, that we had in mind from the start that the Patent Department would be using it.

Brock: Not so.

McIlroy: The Patent-- shortly after NROFF came up, Joe Ossanna heard that the Patent Department had just installed a commercial word processing system.

Brock: Hm!

McIlroy: And he went over there to talk to them to see if he could talk them out of this thing. But they were already using it. And it happened to be based on the PDP-11.

Brock: Oh!

McIlroy: The thing that made the sale was they asked him, "One of the things we do in the Patent Department is file patent applications. We write lots of letters, too. And patent applications have to have line numbers. And the commercial system didn't do that. Can your system do it?" Joe said, "No, not today. But it will-- but it can tomorrow." And they said, "Okay, do that, and we'll try it." And tomorrow, indeed, he had line numbers in there, and the secretaries started dialing into, in the Patent Department, starting dialing into our research computer and suddenly the amount that the file system grew every day was jumped! <laughter> Because they put in 90,000 characters a day, something like that.

Brock: Right, right, right. <laughs>

McIlroy: And a fun day in the lab after the Patent Department came in was since they had already entered a lot of text into the commercial system, could we get that into ours? Because it happened to be a PDP-11, their documents were on DECtape, so we could bring the DECtape, mount it, try to print it out, it was gibberish. It had, obviously, been encoded somehow, and the immediate idea that sprang to mind was, "Let's see if it's encoded in some-- in chunks somewhere." So we quick put together a little bit-wise autocorrelation program. And up pops seven as a peak in the autocorrelation. Seven is the number of bits in an ASCII character. Whereas, eight is the number of bits in a byte. And they had done this silly, almost useless, data compression of just squeezing every sev-- every seven bits they would put in a character, so that made every-- you had to go several words before it got realigned with byte boundary. And decoding was done.

Brock: And was this a product that was offered by DEC or something? Or do you recall?

McIlroy: I think the name of the product was Atex, but I'm not sure. It's outside, but it was not from a computer vendor. It was from a software vendor.

Brock: Okay.

McIlroy: But they resold the DEC machine as part of it.

Brock: Right.

McIlroy: Or required that the patent department buy a DEC machine. I'm not sure whether the Atex machine got taken out, and the Patent got their own, but eventually they got off our machine and did it themselves.

Brock: And it seems like while you describe the document processing or word processing as a fig leaf for this work on operating systems, it just seems to me that from the years of continuation of work about building tools for dealing with text and writing, and--

McIlroy: Oh, there's no doubt, it was a central theme of that lab.

Brock: It seems like it was also a passion. You know?

McIlroy: Yeah, yeah.

Brock: So could you-- you know, that passion for working with text and writing and word processing, do you think that was something that people in the group already had somewhat of that orientation for-- so that would lead the group to choose word processing as a kind of direction for the project that would allow the operating systems work as well. I'm wondering-- I guess it's a kind of-- you can see what I'm struggling for.

McIlroy: Yeah, I think that there's a predilection to enjoy that work on the part of many people. I don't think it had gelled into an identifiable theme as it did later on by the time of the advent of Unix. But people that came-- Lee McMahon, who joined in from Visual and Acoustics Research quite early on, was trained as a Jesuit priest.

Brock: Hm!

McIlroy: And had literary leanings. And I think he came-- I think he was attracted by the word processing. I like the crossword puzzles and that kind of thing. And it's-- when I learned that the Visual and Acoustics people had been able to get hold of [Merriam-] Webster's Dictionary, I grabbed one! We also learned about Webster's Unabridged wordlist, which the Air Force had transcribed, and was possible to get that tape. So we started collecting dictionaries. That didn't happen until the PDP-11 was installed. And wordplay, one of-- I remember a tour de force program that Dennis [Ritchie] wrote early on in the PDP-11, as soon as we got a disk drive. Disk drives weren't very big. The first disk as I recall had just enough space to hold the wordlist of Webster's Unabridged. And Dennis in a tour de force of moving things back and forth between tapes and a small amount of core memory generated a list of all the anagrams in Webster's Unabridged Dictionary. So since the dictionary took up the whole disk, he didn't have much word space. The word space was DEC tapes. I was impressed with that, that accomplishment at that time. <laughter>

Brock: And so, yeah, it sounds like there were these latent orientations, or incipient orientations that once that became a focus of the group, it seems like it blossomed incredibly.

McIlroy: Yes, Lee McMahon and Bob Morris, early on, got the Federalist Papers on-- a transcript of them, and were playing the game that [Frederick] Mosteller did at Harvard of--

Brock: I'm not familiar with this game.

McIlroy: Trying to assess the style in a hope of nailing down attribution to the Federalist Papers, a game that's been played ever since they were written! <laughter> I think people are pretty sure of it now, by common consensus anyways. But one of the things is his statistical analysis of the style of the various documents, to see which of the three authors wrote them.

Brock: In terms of vocabulary mostly, or--

McIlroy: Vocabulary.

Brock: Oh, I guess-

McIlroy: The number-- simple things like counting the density of articles. A little bit about vocabulary, word length. And these kinds of measures then much later popped up in the Writer's Workbench.

Brock: Oh! For-- as applied to software. In the--

McIlroy: Do you know about the Writer's Workbench.

Brock: Oh, Writer's Workbench! I was thinking about the Programmer's-- I'm sorry, yeah! No, I don't know the Writer's Workbench. I'm afraid I don't.

McIlroy: Lorinda Cherry in my department sat down in her office and started writing a program. Oh! Among the other things that we got were-- what's it called? Kučera and Francis, a million words of English text, but-- the Brown Corpus is what it's called [Brown University Standard Corpus of Present-Day American English].

Brock: Okay.

McIlroy: The Brown Corpus was an early statistical attempt to get statistical insight into English. And it was a collection of newspaper articles, magazine articles, chapters of books, for a million words. And among other things, it included an annotation of these texts with parts of speech.

Brock: Okay.

McIlroy: So Lorinda was interested in this, and she wondered could she automate the business of-- The Brown Corpus was hand annotated. Could she automate annotation with parts of speech? And she did a very good job of it. By and large without help of dictionaries. You know, you get-- you know all the function words. And maybe a thousand words, or something like that. And you know about prefixes and suffixes, which if it ends in "-able" and it's multisyllable, it probably is an adjective. So she wrote this thing called PARTS. But she didn't have any use for it. But it was an impressive activity. One day, and we're now talking 1980 timeframe, I think, no 1978 timeframe, Bill [William] Vesterman, a Professor of English at Rutgers called me, I'm not sure why, out of the blue and said, "I have an idea about writing style assessment for use in colleges." "Come up and talk about it." And it was to automate some of these measures which were already in use for characterizing, is this fifth grade text, sixth grade text? By the breadth of vocabulary, the sentence length, the typical word length, number of subordinate clauses. And then he had the idea of add to that stylistic things, such as the density of adjectives, passive voice, and it sounded like Lorinda's thing would be a good thing to build this on. So I put the two of them together, and in about two weeks Lorinda had something going. If Bill had called any department anywhere else in the country, he would have been sending them off on a two-year job! I don't think he ever realized how lucky he was in-- or how prescient he was in making this particular call.

Brock: Mm! <laughs>

McIlroy: So she had this thing which would do assessment and he arranged to get it-- I think it was Colorado State starting using it in their freshman English classes, giving a little bit of feedback. And Nina Macdonald, in the Unix Support Group, outside of research, glommed onto it, and she said, "Let's make this into a product." And Writer's Workbench was the result. And Writer's Workbench unified the various pieces that Vesterman had suggested. And added a few more. It would catch things. In typing, you often end up typing the same word at the end of one line, and words get doubled as you're typing. You would never do it if you were handwriting. So it would find things like doubled words. Another thing of Vesterman's was clichés that you shouldn't use. He had a long list of them! <laughter> And that was in there. And then a front end was put on to not only give the numerical results, but a little interpretation of them, and whether this was regarded as good or bad and for literary, for technical writing. And this actually made the national press. And the only time that Unix has been featured on "The Today Show" was when they called Lorinda in to talk about Writer's Workbench.

Brock: Interesting! And why do you think that was? Just because it was a more relatable application?

McIlroy: Yeah, so everybody knows about it.

Brock: Everybody knows clichés. Yeah! <laughs> And was the idea that this would be an environment in which the writer would work and then analyze their work as they went?

McIlroy: Yeah, exactly, after you got a first draft, run this thing over it. Let it tell you how--

Brock: Right, I have seen something about this.

McIlroy: At the same time, IBM had a whole department working on such things. And eventually that's what's in there, that gets into your word processor, the green underlines in the word processor.

Brock: Well, yeah, that-- and so that's kind of an apotheosis of the interest in word processing. To go back to this moment-- well, I was just interested to get clarity about-- when you began to think of the sort of concepts that would become known later as pipes. Because from some things that I read it struck me that this was an idea or a concept that you had been working with before the Unix effort began.

McIlroy: Oh, yeah, oh, yeah.

Brock: For some time.

McIlroy: Oh, yes!

Brock: Could you talk about where it came from, what the notion was? Just lead us into that.

McIlroy: Yep. If you want to read about it, it's on my website, buried in a little piece about prime number sieves.

Brock: I will read about it.

McIlroy: Right near the top of my website.

Brock: Okay.

McIlroy: Not under my publications. Anyway, goes back to 1964 or so, when Melvin-- was it Melvin Conway, somebody I've never met, but a well-known computer scientist-- wrote about "coroutines," which have now become generators in Python.

Brock: Okay.

McIlroy: It's where you establish a connection between two routines and they're constantly talking back and forth. And this model was clearly, clearly important in real life. This kind of dialogue happens between actors in real life. And I pursued the notion of coroutines. And wrote a manuscript around in 1967, when I was at Oxford. It was never published. It was an idea for putting coroutines into PL/1. But I was never fully happy with the way I did it. However, I did give a lecture in Cambridge, England, in the Spring of 1968, about the subject, where I gave several examples, including the coroutine sieve of Eratosthenes. If you haven't seen this, it's a lovely program.

Brock: I have not.

McIlroy: We'll come back to that.

Brock: Okay. <laughter>

McIlroy: It's a nice aside. And but later on at the same time, then across from the hall from me, Thompson and Ritchie and [Rudd] Canaday were discussing file systems, and coming up with the model that Ken finally built. I was hacking away at my desk at some syntax that would allow you to pipe data one way down a pipeline. I was not doing back and forth communication. And in a certain lack of vision, I didn't have-- I didn't invent an operator for connecting these things. I'd just write one after another. Unfortunately, that's insufficient for real Unix commands, because you can't tell where one command begins and the next command ends, or the previous command ends. And it needed an operator. I did ask Sandy Fraser, who was building something and I cannot remember what it was now on the GE-635, could he give me a facility for feeding the output of one program directly into another? He thought it might be possible but it would be a lot of work. When Unix came up, I started asking Ken the same thing. He didn't say it would be a lot of work and it was clear to me that it wouldn't be because there was enough infrastructure in there already, there was buffered I/O so there was an intermediate staging place between the two programs just one is writing into the same buffer that the other's reading out of, Unix already had the idea that two programs could be reading out of the same buffer at the same time. But Ken didn't see it as a productive enough idea to merit putting it in. Because this was only the facility to connect and it would have been a system call for connecting two processes but it didn't percolate up to the user level, the shell level. So one day I came in, in 1972, and I said, "And after you put this in you could use it directly in the shell with a notation like this," the notation was pretty ugly but it was clear enough. And once he saw the connection with the-- between this facility and its manifestation in the shell, Ken said, "I'll do it," and we had this memorable next day when he did it overnight and also modified many programs so that you could in fact pipe into them. Most programs before that, you had to name the input and name the output. There is no name for a pipe, so he fixed the programs so that they could talk to each other by pipes and the next day we had this really memorable frenzy of, "Look what I can do with pipes. Look what I can do with pipes." One that I still love the memory of is Joe Ossanna had made a cheap facility for doing two column texts in NROFF. What you would do is every other page, you'd set on the left side of the page or the right side of a page and then you would run a program that would just collapse these together, so he didn't have to do anything at NROFF to accomplish this, he just had to write this little program called OVERLAY. And then on pipe day he piped into OVERLAY and he piped into another OVERLAY and this could do four column text for no effort whatsoever.

<laughter>

Brock: And so that whole day was just a proof…

McIlroy: Yes.

Brock: …a realization of what could be done with this.

McIlroy: It was, it was absorbed instantly into one's outlook on programming. And by the end of the week secretaries were piping the output of NROFF into the printer.

Brock: How does this connect to-- I mean this may be a half-baked question so please ignore it if it is but how does this idea of piping, passing messages between programs, you know, is this related at all to some of this whole direction toward object oriented programming or this whole conception of programs, software being messages between objects, I mean is it…

McIlroy: Certainly message passing and piping are related. Recently in putting together a package of documents to send to people to use for the Unix celebration, I reread one of the Multics documents that I wrote and it said in input/output it is not files and devices that are important, the stream is the thing. And there's a difference, a stream of messages between two fixed communicators is a pipe, message passing in general is like message switching where every message says, "This if for Joe. This is for Sam," with a pipe, you don't have to say that, you've made the connection, it's the difference between UDP and TCP and…

Brock: Yeah right, I was just thinking circuit switching or packet switching, yeah.

McIlroy: Yes. And pipes have the virtue of being-- of very simple semantics, message passing, anything goes and it's hard to analyze, there is no proof necessary that a pipeline works, there is proof necessary that a message passing system works and these proofs which people have been working on since the '60s are really nasty.

Brock: So you had in promoting this idea starting in 1964, at one point I understand you really had the analogy to…

McIlroy: Garden hose.

Brock: …garden hose, could you talk about that?

McIlroy: That was written down in the same document where I said that "the stream is the thing," it was about doing I/O on Multics, that was only one subject in that paper and you now know about everything that I said about the subject in that paper. I don't think there's anything more to say about that.

Brock: Okay. And I guess the only other question I had about the development of your thinking about pipes was if it had at all related to your earlier work on macros?

McIlroy: No.

Brock: Okay because to me that it kind of seemed…

McIlroy: They're programming constructs but they're not particularly related.

Brock: For me a string of commands and a string of…

<overlapping conversation>

McIlroy: Well of course-- yes, what pipes are is function composition, mathematically and when one macro calls another macro, function composition, so if we abstract to that level, yes there is a relation.

Brock: Okay, but extremely different in practice.

McIlroy: Mm-hmm.

Brock: And the other question I had about pipes was if the notion of it, if the very notion of it and the utility or the power of pipes implicit in it is one of these hallmarks of what would become known as the Unix philosophy particularly this idea of small simple programs that do one thing.

McIlroy: Yes.

Brock: I mean it seems that post hoc that these two things go together very well, is there an innate relationship between the two?

McIlroy: They fed off each other, there's no doubt of that. So it was only-- I'm not sure of the date when [Brian] Kernighan and [P. J.] Plauger wrote *Programming Tools* but it was only shortly after pipes. But before, the fact that we could have such fun on pipe day is due to that we already had a bunch of small programming tools, we now had a way to compose them.

Brock: <laughs> And I guess I was-- oh I'm sorry.

McIlroy: And so when the Unix special edition of the *Bell System Technical Journal* came out in 1978, we were by then very conscious of it and the introduction which I wrote breaks down these principles.

Brock: Right, and then it had been evident that these things fit together so well. Well could you talk about when-- oh, I'm sorry.

McIlroy: I promised along the way to talk about the sieve.

<laughter>

Brock: Yes, please do.

McIlroy: Do you know the famous Sieve of Eratosthenes?

Brock: I do not.

McIlroy: Okay, Eratosthenes you do know, right?

Brock: Yes.

McIlroy: He said, "Here's how you find the prime numbers, write down all the integers, strike out every other one, take the next integer that has-- " I'm sorry, "That has not been struck out," which in this case would be three, "Strike out every third one. There's some overstrikes going on here. Then take the next one that survived this process which is five, strike out every fifth one."

Brock: Oh, oh, oh, I see, yeah.

McIlroy: "And the survivors of this process are the primes." Very cute idea.

Brock: Yeah, it's a great idea.

McIlroy: Okay, so how do you it with pipes? We make a stack of sieves like you would see in a gravel pit, at the top is a source which simply counts and sends down through this stack, two, three, four, five, six, seven, eight, nine. At the bottom is a sink and any time something comes to the sink, it's a prime. So the first thing it gets is a two, it puts above it a sieve mesh, a filter that throws out anything divisible by two and it emits two at the bottom as a prime. So now what the sink sees is all the numbers that are not divisible by two. When three comes through, it puts another mesh in.

Brock: Oh it constructs the mesh as it receives things.

McIlroy: So it builds this ever growing pipeline, it's an assembly line where we're sending incoming numbers and there are colors in that assembly line, more like in apple processing, somebody's picking out the bad apples that come along. So we have colors we're picking out. At first it's a very short line, every time another prime comes out the bottom, we hire another color who can cull that prime and you got this ever growing sequence of culling processes and it's a lovely image, I was very proud of it when I cooked it up for this talk in Cambridge and it's been a popular example ever since. I never saw it run until-- even when we put on pipes, I was afraid to try it because too many processes for this tiny Unix system, I never saw it run until I made a little coroutine package that did it all in one process on Unix. And Dennis, it would be about 1977 is my guess and Dennis used that to build the coroutine sieve and I finally saw it run years after, years after first conceiving it. Okay, that's sieves.

Brock: <laughs> Well yeah, so to go back into the sort of sequence, so your idea about pipes predates the kind of initial Unix efforts, could you talk about when you first learned that Ken Thompson and Dennis Ritchie and maybe at that time Rudd Canaday were actually creating a new operating system on this PDP-7 Graphic 2 workstation?

McIlroy: Well I was aware of their conversations which were before we got the PDP-7.

Brock: Okay. And these are conversations about what would be the desired file system.

McIlroy: Yeah, and they borrowed from Multics the idea of a hierarchal file system. People can't believe that the file systems were flat before Multics, but they were and there was no organization to them.

Brock: And was also that initial discussion this whole idea of treating files just as a stream of bytes and having the interpretation of the file be up to the programs?

McIlroy: You know, I don't know whether they were talking about it or whether Ken just did that out of simplicity because it was not until I saw the read and write interface for the PDP-7 that I became conscious of the fact that we'd given up all this trash.

Brock: <laughs> Well how did-- oh, I'm sorry.

McIlroy: Come to think of it, I do not recall much discussion in Multics of record structured file systems. Multics had the model-- Multics at base was an immense address space, segmented, and the file systems simply put a structure on top of this really flat almost infinite address space, an address was 72-bits as I recall.[1]

Brock: And so that has also echoed in the kind of nature of the files and the-- well I guess and in the file structure of Unix.

McIlroy: And the file structure of Unix allowed for simultaneous access to files, I'm pretty sure they talked about the buffering and finally the inodes which were little descriptors fixed in space whereas the file itself was scattered all over the disk according to where they could find a free track.

Brock: Well when did or how did these discussions move then from these discussions of what one might do to actually now getting a machine to try them out on?

McIlroy: I actually do not know, I believe that it was Ken that located this used machine and jumped for it, I actually don't know that behind the scenes story.

Brock: Was it physically moving the machine or…

McIlroy: No, the machine existed in a closet and I think -- only slightly bigger than a closet, I don't think it had to be moved there. Although come to think of it, the closet was on the wrong end of the-- was on our end of the building and the machine originally belonged to the department on the other end of the building. Here I am a manager and I don't know how things happened. That's the genius of Bell Labs.

Brock: <laughs> I was curious just to ask a few questions about the Graphic-2 workstation itself and how it was used by I guess it was the Acoustics Research Department, could you tell me a little bit about what the concept was?

McIlroy: The title of that department was Visual and Acoustics Research.

Brock: Oh, thank you. Yeah, the “Visual” gets dropped off in a lot of things I read.

[1] Actually 36-bits. Correction by Doug McIlroy. N.B. https://multicians.org/daley-dennis.html

McIlroy: Yes, we think of them primarily as acoustics because that's where they've had their greatest impact. And acoustics had a lot more to do with the telephone system than visual did although a lot of the bandwidth of the telephone system went into television. And television was first demonstrated by Bell Labs. Bell Labs was not in the television business. So yes, but they did have visual people and they… We had graphical output fairly early on in our comp. center, Stromberg-Carlson machine, it was not the primary output devices, it[2] had been at Whirlwind where it just the highest bandwidth of output was graphical. I can't say why there were interested in interactive graphics over in Visual and Acoustics Research but they were so they built a Graphic-1 and this was Graphic-2, it was a vector graphics machine. I cannot tell you what they were using it for, but they had already learned apparently all they wanted to from this and were going on to bigger and better things and discarded the machine. They had an apparently unlimited budget for hardware.

Brock: And this must have been a relatively, I mean it had to have been only three, four years old at the most.

McIlroy: Yes. And I cannot recall, I don't think I ever saw a demo on that machine while it was in their department.

Brock: Did you know, and I think it was a person named William…

McIlroy: Ninke.

Brock: …Ninke who started this all off with the Graphic-1. And I recently saw a video, a digitized film from 1965 that shows evidently this kind of graphical document editor on the Graphic-1 machine, I was quite blown away and I looked up a paper of his from '65 that does talk about this idea. I wondered if anybody in your group had ever seen this thing.

McIlroy: No.

Brock: Yeah, it was very, very surprising.

McIlroy: Yes because we didn't do graphic document editing until long after that, like maybe 1982, yes.

Brock: He was dragging words around with a light pen and things like that, it was astonishing to me. But I'll have to track down William and see what he can tell us. Well then could you talk about when were you aware that Ken Thompson and with Dennis Ritchie had really been implementing some of these ideas?

McIlroy: Oh, as soon as the machine came in, I knew they were going to do that on it.

[2] That is, graphical output. Clarification by Doug McIlroy.

Brock: And was the discussion also from the earliest period about also creating a new programming language in which to implement a new operating system…

McIlroy: No.

Brock: …were those-- that was something that developed later.

McIlroy: If you're going on-- yeah, programming languages were parallel to but not in the design path of operating system.

Brock: I see.

McIlroy: The only language that they used for making the operating system was assembly language so they did have to write assemblers for it.

Brock: Right. And there's a discussion about the role of this computer game called Space Travel in this early story and I wondered what you thought of what has been kind of said about that, about the desire to have a machine on which they could, particularly Ken Thompson I suppose, have unfettered time to play Space Travel was a motivator, could you speak to that?

McIlroy: You've now hit-- we talked about one theme that you could find in our center which was text processing. Another theme in which we were much more amateur is astronomy.

Brock: Oh we had talked about that last time with the…

McIlroy: And Space Travel fits in that second theme because what Ken did was put in a model of the solar system and all its planets and moons and then you'd drive your vehicle around and try to land on them. Landing on-- you're going very fast to get from one place to another and you have to land softly when you get to a planet and the graphic was used to-- first of all planets would show as dots on the screen, as you approached them, they become discs and finally you just see a slightly curved surface because you're too close to see the whole planet. And I think the fun of the game was landing safely and beginning to learn the names of all the moons. Along with that, they did some orbital mechanics through typically a three-body problem and it's when they were doing the three-body-- you can get some really crazy orbits with three-bodies on the screen. And in the course of doing that, they discovered their naive integration techniques for solving the differential equations didn't conserve energy and if you ran very long, everything fell together. And this came up on TUHS [The Unix Heritage Society] recently, you may have seen it.

Brock: That Bob Morris figured it out.

McIlroy: Dick [Richard] Hamming.

Brock: Dick Hamming, sorry, yeah.

McIlroy: And I had not realized in one of the later posts on that thread somebody said, "This is now known as symplectic integration," and it became a big thing. Apparently in the '70s and '80s Hamming had done it as a throwaway, "Here, I've done it for you this afternoon," beforehand.

Brock: And was it the case that a version of the Space Travel game had been available on Multics or was it-- that seems to be some of what I had read that this was…

McIlroy: Well we didn't have any graphical terminals on Multics so I don't think so.

Brock: Okay. But it was…

McIlroy: Now did Ken bring it up standalone on a PDP-7 before he built the operating system, I don't know the answer to that, he may well have. You'll have to ask Ken.

Brock: Okay. I guess what I'm fishing around for is a sense of how important was that game to sort of the beginning of this group?

McIlroy: That was a good exercise in support of this-- when you're bringing up a new computer it really helps to have another one in the background, in this case it was the old 635 to prepare programs on it and then carry them over and Space Travel was done that way, come to think of it, I don't know what language Space Travel was written in. Oh no, I do know because you've collected it, it was written in assembly language.

Brock: Yes, it was.

<laughter>

Brock: It's interesting but that is part of this kind of astronomy interest that runs throughout the story of this group. Well when did you find yourself as the new operating system-- as Ken Thompson with Dennis Ritchie are putting together this new operating system, when did it really start to occupy a lot of your attention?

McIlroy: Instantly.

<laughter>

Brock: Could you expand on that? Why was that?

McIlroy: Well I saw it, it was so appealing in not having a burden of file descriptions which came-- it was also appealing in that they had, by the time I was using it, turned it into a two user system. This meant that you could begin to use it in time sharing mode which we had already enjoyed when we were preparing the Multics project. It was obviously a lot more fun to write a program for this machine than for its predecessors and I undertook to bring up TMG, bootstrap TMG from scratch on the machine. So I had a significant project that I could do and it was just such fun that I completely abandoned my connections with the comp. center machine except for the maintenance of ROFF.

Brock: Wow, and just switched entirely over into this environment. And by this time…

McIlroy: TMG is a big enough project that it took a while and once I got it up there it was self-supporting and I could build TMG to a level that it had not previously reached.

Brock: I will confess my ignorance about what TMG did, I'm sorry.

McIlroy: Okay. TMG is an invention of Bob [Robert M.] McClure, it is for writing compilers.

Brock: Oh.

McIlroy: It's slightly curious, it's none of the standard grammar models, it's—[Alfred] Aho and [Jeffrey] Ullman finally invented a name for TMG grammar, but it was very effective for entering the grammar and the translation rules. McClure sent it to me, he made it for the CDC 1604, he hand transliterated in it a machine language for the 7090, sent it to me, I debugged it because there were transcription errors but it was an interesting thing. I knew that the logic was right, all I had to find was clerical errors. And I brought it up on the 7090, used it to-- what compilers did I use it to build besides itself? Funny, I may never have. Oh, oh yes, yes there is one big-- I didn't use it on-- it was the tool that saved us on the Multics project when our contractor could not produce a PL/1 compiler. Bob Morris and I wrote one in TMG.

Brock: Okay, that's what I was wondering.

McIlroy: And that got ported across from the 7090 to the 635 by Clem [Clement F.] Pease by the remarkable strategy of making macros for the 635 to emulate the 7090.

Brock: Wow.

McIlroy: So this program in 7090 assembly code ran on the 635.

Brock: Creating an emulator out of macros essentially.

McIlroy: Yes.

Brock: That's amazing.

McIlroy: Well they were both 36-bit machines. But there were—the 7090 had an accumulator that had one instruction that it could do, sign-magnitude addition, it had another instruction that could do two's complement addition and it had another instruction that could do one's complement addition.

<laughter>

McIlroy: And all of those-- at least two of those three were used in TMG but Clem took care of them all.

Brock: So as you were working to develop TMG for Unix, I guess I was wondering about just this whole practice in the developing group of users of writing these distinct programs that accomplish a specific function and creating a shared repository of them. How did that practice come about?

McIlroy: Well the shared repository we had come to appreciate from CTSS and in fact everybody realized that CTSS, it was an afterthought at CTSS, CTSS was going to be pure timesharing, each user thought he had the whole machine or his own machine and then notion of cooperation grew, became recognized there and Multics-- and affirmed in Multics and reaffirmed in Unix.

Brock: So it was kind of a given of the timesharing approach.

McIlroy: Mm-hmm.

Brock: That makes sense. I was curious too about when-- I was uncertain and it's probably a fault of my research about just when the idea of software tools began to be articulated in the whole project, was that…

McIlroy: The word certainly came from [Brian W.] Kernighan and [P. J.] Plauger, that word was new to me when I saw their manuscript. [*Software Tools*, 1976]

Brock: Okay. But it seems kind of inherent in what was going on from the…

McIlroy: Oh yes.

Brock: Giving a name to something that had been going for a while.

McIlroy: That introduction to the 1978 BSTJ which listed these principles, started recognizing things that we had intuited all along.

Brock: Well I have some questions about that kind of philosophy that I would like to dig into after a few other questions maybe if I could. One was just a question about the simplicity of some of these early Unix programs and utilities that people were developing, it seems that simplicity maybe even over functionality even in a way was such a primary value for what people were doing, could you…

McIlroy: Well it was simplicity with an eye to generality and generality often produces function, instead of listing a bunch of functions, can you find a higher-level principle that encompasses them all?

Brock: So that was really simplicity and generality…

McIlroy: And of course as has often been observed it might not have arisen had we been able to get a big machine like a PDP-10. The fact that we had tiny machines meant we couldn't put in a bunch of functions into a program, we had to find the underlying nugget that made your general goal, that reached your general goal.

Brock: So it was really that idea of the constraint leading to…

McIlroy: That helped a lot. And Unix wouldn't have-- the universities wouldn't have bought into Unix if it hadn't been on a cheap machine and we wouldn't be here today if we had gotten a PDP-10.

Brock: Because you think it's again, it's a style that adapted to the constraint that you were faced but once the power of it…

McIlroy: But if you've got a mathematical background, this desire for succinct elegance is ingrained in the whole field of mathematics, it's not necessarily ingrained in the field of computing.

<laughter>

Brock: Well, I mean, it is abstraction…

McIlroy: Yes.

Brock: …it is, yeah, and it is this kind of mathematical aesthetic to it, it's interesting. And some people have quipped that sort of the brevity of everything…

McIlroy: Oh yes, we used two letter commands. <laughs>

Brock: You know, but this is also partly a constraint of the machine environment or what do you think?

McIlroy: Only in the ROFF family which had two letter editing markup commands and that was because they fit in one machine word but it certainly is not the reason for making the commands short in the shell because the shell dealt with strings. Maybe it kept the table of commands to be a little smaller but it already was variable length so there was no real size constraint.

Brock: So that was stylistic, just a matter of…

McIlroy: Yeah, minimize keystrokes.

Brock: <laughs> So it could be just simply distaste for typing that…

McIlroy: Yeah.

Brock: Yeah. I was interested in how other people kind of came into the effort and saw what was going on and chose to participate in it, how did that kind of accretion go on?

McIlroy: Well you've heard how I came in. Bob Morris was just part of the group at the time and he had worked on Multics and he just naturally fell into this I think for the same reasons that I did, particularly his interest in security which manifested itself pretty early on in the Unix days. With our own machine rather than a comp. center machine, we could get down inside and address things like security and play with them. In the comp. center machine, you had to stay at user level, you couldn't get down inside the operating system. So I think that would be part of what attracted Bob. I've mentioned Lee McMahon and I think we can say it's text processing that kind of drew him in. Ossanna, well, he was right there from the beginning and he was clearly the most committed of all these people to Multics. He had addressed the issue of input devices on Multics and had advised the makers of the Model 37 Teletype on its character set at least, I'm not sure about what other properties and he had more diligently than most other people searched for a successor to Multics when we pulled out and so he was-- and he too took responsibility for essentially organizing the machine room when a PDP-11 came up, so he was just in there from the start. First person who-- well Lee McMahon came in from outside, Lorinda Cherry a couple of years later. I don't know how she heard about it, but she was in Visual and Acoustics Research, she came over and saw Sam Morgan said, "Could I transfer here?" And she had to agree to some kind of probationary transfer and fighting the usual thing that women have to fight, or had to fight, she just thought it looked like an interesting to place to join in and did. She had to first come, in fact she first came in not working on Unix but working with Ken Knowlton on drawing pictures of atoms, of molecules and then slipped into Unix by working with Bob Morris on TYPO. Do you know TYPO?

Brock: No.

McIlroy: One of the more interesting Unix programs. And finally earned her place in the group but she really sort of had to beat on the door to get in unlike the rest of us that just were able to drift in. Let's see, Brian [Kernighan], he didn't come in in the PDP-7 days, although he witnessed it, I don't think he was really playing much with the PDP-7. But I think it was the excitement of all the new bits of software that were coming and he could write his book and…

<laughter>

Brock: Right.

McIlroy: He had a subject to write about. Let's see, who-- that's sort of the original dramatis personae…

Brock: And Russ Canaday had been there from the start…

McIlroy: Rudd Canaday…

Brock: Oh sorry, Rudd.

McIlroy: …was never a Unix user, he was a visitor in our department when they were having the talks but he had gone back to his original department in the Labs before we built one.

Brock: Okay. And when was it evident…

McIlroy: Fred [Frederick T.] Grampp. There were later people that were actually recruited, Fred Grampp and Ed Sitar out of the comp. center. Grampp was very important, fascinating guy.

Brock: And he would have come into the effort a few years after it started?

McIlroy: Yeah, I think he probably came in around maybe '96.

Brock: Seventy-six.

McIlroy: Yeah, I'm sorry, '76.

Brock: I guess I was wondering-- and so for Ken Thompson and Dennis Ritchie this just became the centerpiece of their research program.

McIlroy: For a long time, and for Ritchie essentially the rest of his career.

Brock: Yes. I guess I was wondering when it became sort of the centerpiece of your group or was it the centerpiece of your group?

McIlroy: My group had two centers in that we had the systems people and we had the theoreticians and nice thing is that sometimes they talked to each other.

<laughter>

McIlroy: But even the theoreticians-- I think it's about the time of Fred Grampp when the whole computing science center started running its own computer center and only the numerical analysts were left by and large using the company comp. center where Fortran reigned.

Brock: I see.

McIlroy: And, can I put it-- when did we get a computer room fancier than the single PDP-11 in the attic, you know, I cannot remember when the famous Unix lab with the computer room in the back started…

Brock: I don't know the date.

McIlroy: …I can't put a date on that. It would be in the mid to late '70s and that was done at a level of management above me that we would do that because Sam Morgan recruited Fred Grampp and he gets a lot of credit for having done that and Grampp ran the back room which had-- oh okay, we never had multiple PDP-11s. So that back room, it came with a VAX and eventually we had a quite a stable of VAXen, yeah. All right, and when did the VAX happen, 1979 I think.

Brock: Okay. And so that was…

McIlroy: Oh, and Steve Johnson and Dennis Ritchie worked on making a portable C compiler so that they could transfer Unix onto another computer and they had in mind any other computer but the first one we got happened to be the Interdata which would have been probably '78 and I don't think many people got to use the Interdata because the VAX then came out, looked very attractive, and another port was done to the VAX, it was done in Holmdel by John Reiser and at Berkeley…

<break in recording>

Brock: Well we were just talking about computer science getting its own computer center…

McIlroy: Yes.

Brock: …and migrating everything over to the VAX. So that was, so in a way that was Bell Labs' computer science as a whole…

McIlroy: Yes.

Brock: …using Unix as its environment. Which was happening at the same time I guess that Unix was starting to both travel around in different places in the Bell system but also getting out particularly into universities.

McIlroy: Yeah well that had happened with the PDP-11, that really took off probably around 1974.

Brock: Okay. That early, I hadn't realized that.

McIlroy: Oh yeah.

Brock: Okay. Could you talk a little bit about when this idea of developing a new programming language in which Unix would be implemented and Dennis Ritchie's whole effort really came about?

McIlroy: Well that was an objective from the start, reading assembly language is clearly an interim state of affairs and we had this background of BCPL and Ken made B which was, pioneered some of the notations that were adopted in C. But all these previous languages had been for word-organized machines and the PDP-11 was a byte-addressable machine. And C was an attempt to bring these languages up to real compatibility with a byte-addressed machine. Most of the other machines had whatever you were talking to it was always the same size, we now had bytes, words and double words for floating point. The big issue was handling subscripts in the language which was close to assembly language, the language allowed you to manipulate addresses as and I had an-- what did subscripting mean, how could you do nice subscripting so that it worked on bytes, words and double words? And Dennis thought long and hard about the relationship of address manipulation and subscripting and finally came up with what it means to add one to an address, it depends on what kind of an address it is when you add one to it.

Brock: Right. <laughs>

McIlroy: And this grew into C.

Brock: And this relates to, actually that talk at the Unix 50th anniversary where, who was talking about it, someone was talking about the way that C handles memory as being foundational to its success, so this is very much…

McIlroy: That's correct.

Brock: …related to what we're talking about here.

McIlroy: Very much.

Brock: And it was that kind of, that flexibility or that abstraction if you will again that gave it its power and was that-- in the development of C, was that more of a sort of a solitary activity that Dennis Ritchie was doing or was it highly interactive?

McIlroy: Oh it was very much in public, he was bouncing the ideas off of people all the time.

Brock: Okay. So in the same way that it reflected the same kind of mode of operation, communal or collegial mode of operation that the entire thing did.

McIlroy: Yes.

Brock: Okay, interesting. And what about the story about giving Unix its name, I've read a couple of different accounts of that but…

McIlroy: Nobody can quite remember, and the name presumably came from either Peter Neumann or the name as heard came either from Peter Neumann or Brian Kernighan and neither one of them is quite willing to say, "I did it."

Brock: <laughs>

McIlroy: Because it was a play on Multics, everybody thought that there was a C, I mean most of us thought there was a C in the word but it was Ken who wrote it down with an X. And simply because systems often had names in all caps, Ken did that and then we regretted it for-- we spent a lot and it got turned into a trademark by the legal folks and then they leaned on us to keep writing it with all caps when we realized that it was simply a proper noun.

Brock: Right. <laughs>

McIlroy: It was not an acronym.

Brock: Right. And what about Ken Thompson's work, was he similarly interactive with other people about his ideas?

McIlroy: Yes except for his working hours.

Brock: Which were nocturnal.

McIlroy: He would typically not be there in the morning but would be there to god knows when at night. So he was working alone a lot but when he was in the lab it was communal work…

Brock: Okay.

McIlroy: …when other people were in the lab it was communal work I should say.

Brock: Let me just look at my question list here. We talked a little bit-- well it sounds like you I mean really started to develop a whole sequence of Unix programs that you were concerned with and I just wrote down a list of some that I associated with you, SPELL, DIFF, SORT, JOIN, GRAPH, SPEAK, TSORT, CALENDAR, ECHO.

McIlroy: That list of course is very short compared to Ken's list.

Brock: Well.

<laughter>

Brock: But it's-- so was it the case that these were what connects this list together is that a need arose for something…

McIlroy: Yes.

Brock: …an opportunity arose for something…

McIlroy: Yes.

Brock: …and it was part of this culture of quickly trying to develop something and…

McIlroy: You have said it better than I could.

Brock: <laughs> So that's the thematic that connects us of various, of just the tools that one…

McIlroy: Take SPEAK for example, Ossanna who couldn't resist an interesting new piece of equipment -- the phototypesetter being the preeminent one -- saw the Votrax and got one and stuck it on the machine. And Lee McMahon promptly became quite good at cooking up phoneme strings to feed to the Votrax. Did we talk about this when you were here before?

Brock: We did briefly yeah, yeah.

McIlroy: Oh okay, so. And I perceived that it was a pain to write phonetically when why couldn't-- and it was obvious that much of English is phonetic but it's full of exceptions and I sat down one night and wrote a program that allowed you to-- it was driven off a table which said if you see this letter, it's pronounced this way but if it's contained in this longer string it's pronounced another way. And this was sufficient, and this was sufficiently good that it took over, nobody ever wrote a phoneme string again.

Brock: So with some of these programs that many of the different users were using it sounds like there was a culture too of, for example, while one person may have written let's say the initial version of one of these programs that at a later time someone may have thought they had something, a different approach that was superior or something that they could…

McIlroy: That's exactly true.

Brock: …improvement or what have you and that in a sense the ownership of that program would transfer to someone else you know, was that…

McIlroy: That's correct, in fact the cry in the Unix room was, "You touched it last, it's yours." That way…

Brock: <laughs> If an issue came up with it subsequently?

McIlroy: Yes.

Brock: So was that something new or was that kind of a-- was that like a new cultural practice for creating programs or…

McIlroy: You know, I think it's-- just how it came that there was not a proprietary view of what you had done. I blame it on there was so much credit to go around that you didn't have to worry about preserving your ownership. Had I seen this kind of thing before is an interesting-- and the answer is not really.

Brock: Well it did strike me as unusual that people would be that open to have their work revised or done away with or, you know.

McIlroy: But everybody appreciated the fact that things were always improving. And I did tell you about SORT in the statement about "Wide options are available," in the first edition manual.

Brock: I don't think we did talk about that.

McIlroy: Well SORT in the first edition manual was there and it said, "Wide options are available," and I asked Ken, "What are they?" and he said, "You can get whatever you want by altering the source."

<laughter>

McIlroy: And indeed one night, I did finally put in a whole lot of options to SORT and I became the owner.

Brock: <laughs>

<break in recording>

Brock: So we were just talking about this culture of sharing and revision and that seems to have continued through the entire…

McIlroy: Certainly until I retired.

Brock: <laughs> Well, we talked a little bit about how word processing and document production had become this important dimension and initially an important sort of rationale for the effort and that certainly seems to have grown very rapidly. I was also very interested by a story that you told at the recent Unix anniversary specifically about when the phototypesetter that you just mentioned had been acquired and work that the Unix group had been doing for I think it was the head of AT&T.

McIlroy: The CEO, yeah.

Brock: The CEO of AT&T and I wondered if you could just retrace that a little bit here for this recording.

McIlroy: Charlie Brown, the CEO of AT&T apparently was vain about not wearing glasses and his vice president for public relations, I think I've got the title more or less right, got wind-- so he didn't like to give speeches from text, read speeches with his glasses on, and the vice president got wind that we did have this typesetter and asked could we typeset in large type and if so would we be willing to do it for him and it didn't seem like a bad idea to have a connection to the top. The only burden was that we would have to run the rather persnickety printer ourselves and somehow get the typeset in and send this typeset stuff to New York but they could use our computer to do the TROFFing from a terminal in New York, we didn't have to take any responsibility for that. And this did work. One exciting day his speech writers were a little bit late with the draft and they had to dispatch a helicopter to the lawn of Bell Labs to pick up the output for his forthcoming speech.

Somewhat later I discovered that they had found that having access to Unix was kind of nice and they had started to use it for other purposes than simply typesetting speeches. I found that minutes of the board of directors were being maintained on our machine and a machine where everybody was a super user, anybody could look at whatever the CEO was keeping on our machine and a machine which was very-- if there were any Unix machine in the world that people would want to break into, it was ours and that did happen occasionally. And I called New York and said, "I think this would be a good idea if-- we'd rather not have the corporate secrets on our machine, please get your own." And the price tag is something they could afford at that level. So finally there ended up being a Unix machine in the CEO's office.

Brock: And what time frame was that?

McIlroy: Oh that was probably around 1975, I could look up and see when Charlie Brown was president but…

Brock: Right. And that ran for some time.

McIlroy: I think it probably ran a couple of years.

Brock: Right. In the whole story of the group getting the phototypesetter was that something that would have been sort of in the research budget for Joe Ossanna individually?

McIlroy: Yeah by then, it's an interesting thing about the research budget. I was a department head for 21 years and never saw my budget. The budget was handled at one level above me and by 1972, I think it was that year we got the phototypesetter, higher management had decided that Unix was after all an alright research topic. And the phototypesetter fit very nicely with the original rationale, I think there was no particular trouble in getting the money for it.

Brock: And was it the case when, I guess, the next-- you got the a PDP-11 after the Graphic-2 workstation, and that would've been you just put in a request to get that for the...?

McIlroy: Well, that was a little bit different. Again, Visual and Acoustics Research comes to the aid of it. Would've been the-- I guess it would've been the end of 1970 when we ordered that machine, and they had a little money left over, and Max Mathews, who ran the outfit, had seen that Unix seemed to be a good thing and he kicked in some money.

Brock: Hm, and did that--

McIlroy: So that was-- so we got a gift from somewhere else in the labs for much of the price of that machine.

Brock: Hm, and did it come to pass that eventually Visual and Acoustics Research started to run Unix systems of their own for their work or how did--

McIlroy: Did they have-- they had, of course, a whole series of special machines that hooked directly to apparatus.

Brock: Right.

McIlroy: They must've had them, because folks like Fernando Pereira and-- knew all about Unix and were good users and although they had accounts on our machines, I'm sure they didn't do all their work on our machines. Funny I can't... But if they did have a Unix machine, I can't remember what room it was in.[3]

[3] Joe Olive, who specialized in speech synthesis [in the Visual and Acoustics Research Department], put my SPEAK program into a standalone synthesizer as a last resort when their more sophisticated dictionary-based algorithm got stumped. That machine was a micro PDP-11. Joe gave us one that I took to Australia for a demo in 1982 and was still in service in the Unix room when I retired. Clarification by Doug McIlroy.

<laughter>

Brock: Yeah, and they were the adjacent department, if you will?

McIlroy: Yeah. They were at one end of the building. Math Department was-- well, the Math Department was in the middle but mostly one level down, and we were at the other end.

Brock: And then--

McIlroy: And then, of course, they did get a Unix machine in the comp. center, the Cray.

Brock: Oh. <laughs>

McIlroy: So if you're doing big-time computation you'd be doing it on a Unix machine.

Brock: Huh. I hadn't realized that. I was curious when we talked a little bit before about the, I think it was the Writer's Workbench, but the-- which I had confused with the Programmer's <laughs> Workbench, and that-- could you talk about that effort and when it began and how it related to your department?

McIlroy: Programmer's Workbench was at Piscataway, and where there was an IBM 360 comp center, and it was apparently-- it was perceived to be something of a drag to prepare input, get output from the 360, and there was no way to use it as a time-sharing system, and a group there saw that putting a front end together with a cheap Unix machine should, would, they hoped, improve productivity. Like, you could prepare your documents, you could prepare your programs, with quick turnover and have them ship directly into the machine. Almost turned the 360 into a time-sharing machine. The people there were very good, and were very much in contact with us, and they also were interested in-- they were associated with big programming projects. So one of their first really important contributions was SCCS, the source code control system. We didn't have to use it. We were only doing small things, but that spread everywhere through the Labs and made a model for which things like Git and CVS are the present incarnations.

Brock: And that was source control systems for all of the different groups within the Bell System who were...?

McIlroy: Yeah. Mm-hm.

Brock: Okay. A common tool that people could use for source code control.

McIlroy: Yes. Yes.

Brock: I see, and so that was an example of the--

McIlroy: There were several Unix tools that were-- that they had slightly different flavor when they came from that group. But there are several of the standard Unix tools that came from them. I think particularly of FIND.

Brock: And we talked about some of the, well, it sounds almost, the question that I wrote, the, “How did Unix begin to spread within Bell Labs and the Bell system?” and it sounds like, you know, some were almost informal connections, but it almost, it almost sounds like partially a process of osmosis. I was wondering if there was ever or eventually was a more formal push?

McIlroy: Yes. Well, we have the, perhaps the first three installations are worth mentioning. We’ve already talked about how Patent got theirs, and that was a matter of salesmanship.

<laughter>

Brock: Right.

McIlroy: A group at Holmdel installed a Unix-- I’m sorry, adopted Unix as an attachment for wire centers, for analyzing all the trouble reports that came out of these switches. And in those days, most of the switches were still electromechanical, and they have a little machine over in the corner that produces immense 120-column punch cards every time they detect an error, and they’re always detecting errors, because it’s electromechanical. Maybe a fly in--

<laughter>

Brock: Right.

McIlroy: A fly caught between the relay contact, and so they write a system for taking those trouble reports and trying to pin down what-- to tell the repairman what he’s got to do.

Brock: Okay.

McIlroy: And the third, and that was at their, that export, was at their initiative, and there was one export before that into the field in a wire center in Charlotte, North Carolina, where somebody wanted an on-the-floor – a switching center, again -- for scheduling craft people’s work. They had a TV that says what you’re going to do next.

<laughter>

Brock: Okay.

McIlroy: It was a small system and apparently was quite successful, and they were expanding its application. They wanted 4K of memory once, and their management thought, “Do they really need it?" and they came to us and asked us to tell their higher management that, "Yes, it would help these people to have another 4K of memory."

Brock: <laughs> And how did that come about?

McIlroy: And how they learned about Unix I don't know. You could get that story from Ken. Ken personally went down there to install the system for them.

Brock: And so these must-- and these were in the first half of the 1970s?

McIlroy: These would've been 1973 probably.

Brock: Wow, very early on.

McIlroy: Really very early. At the same time Columbia University, Lou Katz, managed to get a university license, and by 1974-- I think it was 1974, the first meeting that grew into USENIX happened at-- was called by Lou Katz.

Brock: Hm, and did you have any insight into that whole story of the university licenses for...? Was that something that the Patent Department was...?

McIlroy: No, that's-- that was done by Western Electric.

Brock: Oh. That's interesting.

McIlroy: Oh, but I didn't finish the--

Brock: Oh, I'm sorry.

McIlroy: You wanted to know did it, you asked, "Did things finally get formalized?"

Brock: Yes.

McIlroy: And yes, Berkley Tague. Berkley, he's spelled with two "E's," not three. Berkley Tague identified Unix as a growth product within the Labs, and he lobbied to create a group called the, which became, the Unix Support Group, and I would place this probably around 1974, but...

Brock: Wow. And that was to...?

McIlroy: Yeah, and they were going to handle the distribution, and they grew finally to actually often creating things that were needed in the rest of the Labs. Writer's Workbench, I told you, was their conception out of pieces that had existed that Lorinda had built.

Brock: Right. And they were going to-- and but their remit was initially to spread Unix throughout the Labs?

McIlroy: Yes.

Brock: Yeah.

McIlroy: Yeah.

Brock: And later...?

McIlroy: Later it became a product.

Brock: Okay.

McIlroy: Yeah, System 3 and System 5.

Brock: But and to return to the--

McIlroy: And but when it was Tague's group it was specifically within the Lab.

Brock: Got it. But it sounds like some of the first travels of Unix outside the labs was through this university licensing and you said...?

McIlroy: Oh, abs-- that's the-- yeah. There were industrial licenses, but, you know, I can't name a single person who had one.[4] <laughs>

Brock: Huh.

McIlroy: Yeah, but the universities, many of them kept, you know, were in contact with us, but...

Brock: And while I just know from, I mean, I know from the history of semiconductors that, you know, Western Electric handled all the licenses for, let's say, the transistor, in an ongoing fashion, and created for the transistor extremely accessible licenses, and it seems like Western Electric did the same thing for Unix.

McIlroy: Yes.

Brock: Made extremely-- especially this research or university license.

[4] When I said this, I was thinking of the timeframe of the early university licenses. Of course, later on IBM, DEC and others bought Unix licenses, and the "Unix wars" ensued. Clarification by Doug McIlroy.

McIlroy: The universities, my recollection is, that the licensing fee was $250, which included the tapes that had to be...

<laughter>

McIlroy: But it's a, "We'll give you no help."

Brock: Right. But was this something that you had a discussion with people from the Western Electric side or was this kind of an autonomous?

McIlroy: Yes. No, there were discussions. For pricing of a commercial license. I do remember Ken coming up with an idea, "Well, what-- 10 cents a line of code."

McIlroy: And the commercial licenses ended up being around $25,000, <laughs> so...

Brock: Mm-hm. Which is still, I mean, again, sounds like a very accessible figure.

McIlroy: Mm-hm.

Brock: Hm, and--

McIlroy: But other than this input of, "What's the value of Unix?" which our guys wanted to spread it and wanted to keep the price low, except for that kind of discussion, everything else was in Western Electric. We did know Al Arms, who was the Unix licensing person, but we didn't spend a lot of time with him.

Brock: Hm. And then--

McIlroy: He would go to USENIX meetings, is my recollection.

Brock: Huh. Because it is interesting, it's a continuation of-- well, it's interesting that it's a critical decision that they made the university license so accessible, and--

McIlroy: Well, they were running scared of the Consent Decree of 1956, which said, "You are to stay in the telephone business and not branch out into a lot of other things."

Brock: Right. Most explicitly computers, right?

McIlroy: Yeah.

Brock: So, I mean, this would be, in a way, it's another example that, "Well, we're not really in the computer business if we're doing this sort of thing."

McIlroy: Yeah, mm-hm.

Brock: That's fascinating. Well, I thought maybe at this point we could spend a few moments talking about something we touched on before, about these kind of elements that by 1978 certainly were recognized as kind of the Unix approach or philosophy, and I thought if we could I could just go through some of these, some characterizations of this Unix philosophy and maybe we could talk about, well, just hear about your thoughts about the sources for or the roots of these, these various elements, and we've talked about some of this before, but there's famously, you know, write programs that do one thing and do it well. So I wondered...

McIlroy: That "do one thing" certainly started be-- you didn't have room to do more than one thing.

Brock: Right.

<laughter>

McIlroy: And then it became seen as... It's interesting. We have a whole bunch of utilities, different programs that work on files, of copy and link and print and tell the size and so, and in previous systems there'd usually be one, these would be all functions collected under one file utility, grand file utility, and that certainly happened because of size constraints originally, but we stuck with it.

Brock: And do you think it also comports with what we were talking about before of this--

McIlroy: It also, probably because there were a lot of people working on it and there was not a project or official project or organization with somebody saying, "This is your job." Nobody said, "File utilities are your job." You made one when you needed one.

Brock: <laughs> And it seems to me that it would also be quite compatible with what we were talking before, about, you know, the aesthetic of abstraction and generality.

McIlroy: Mm-hm.

Brock: And then there's this idea of write programs so that they-- write programs to work together.

McIlroy: That's--

Brock: And I wonder if that's a corollary of the first. <laughs>

McIlroy: But it's-- that is certainly written after the advent of pipes, where "work together" had a much more intimate meaning than it did before.

Brock: Mm. Right. And this one I thought was very interesting, and maybe, I don't know. I haven't seen it getting the attention that the others have, this idea about writing programs to handle text streams.

McIlroy: Yes.

Brock: Because the text stream is a kind of a universal interface. That I thought is a very interesting kind of statement, and certainly I can see how it relates to the notion of pipes, also the conception of, you know, a file is a stream of bytes. But I would just love to hear you, your thoughts on that, because that just seems very important to me.

McIlroy: Yeah, when-- and it's universal interface not just to programs, but to people. Once you get data hidden in a computer encoding that's not human readable, debugging becomes a chore. It means you have to-- if the data is going to be used by some unanticipated application, that application suddenly has to be trained on it, trained to do that data.

Brock: Right.

McIlroy: This is more a help to the programmer perhaps than the application itself, because if it's a text stream you still have to, you still have to understand the text stream.

<laughter>

Brock: Mm-hm.

McIlroy: But decoding a text stream is something you just do with running the SCANF program. Whereas decoding a bit layout had traditionally, there wasn't an equivalent of SCANF for bit layouts, so you had to hard write, write hard code for each, how to get each field out of the text. These are the kinds of considerations that led to that statement.

Brock: Hm, hm. Well, it's-- and do you think-- how important do you think that that concept of, you know, having kind of text stream--

McIlroy: Well, another important thing is that with time, data formats evolve, and adding another-- when you don't know how long the string is anyway, adding another field to it doesn't make much difference.

Brock: Mm-hm. <laughs>

McIlroy: But if it's encoded in fixed bit layout, that you'll have to go to more work to add another field.

Brock: Right.

McIlroy: Or to change or-- and one, or a simpler case, you had a field that runs from zero to a hundred and eventually you ran out of-- you had more than a hundred codes, so you had to make the field wider. Making it wider is zero work on a text stream, and it is real work on a fixed length encoding.

Brock: Hm.

McIlroy: I should've said that earlier.

Brock: And do you think that, has that kind of maxim or that approach, had proven to be very important for...?

McIlroy: Yeah, I think so.

Brock: Okay.

McIlroy: Particularly for this business of looking at data in new ways. If the data is text, the cost of entry to look at it is very low.

Brock: Right. That's really fascinating. We talked about these simple minimalistic but general programs, and another hallmark approach seems to be that this idea of--

McIlroy: Let me say one more thing--

Brock: Oh, please.

McIlroy: --about the text.

Brock: Please, please.

McIlroy: For numeric data, all you have to say about text is, "There is this number, then there's that number, then there's that number." You don't have to say, "It's 5 bits of this number; it's 17 bits of that number, and this one's floating point and this one isn't." That simplifies life a lot.

Brock: Right. Because it's just--

McIlroy: Just a number. <laughs>

Brock: Number, line break, number, line break, that sort of thing or...?

McIlroy: Mm-hm. Can be that or number, blank space.

Brock: Oh. Right.

McIlroy: Who cares how much blank-- it's a number that you can recognize as a number. Now you go on to recognize that...

Brock: Right.

McIlroy: As long as it's set off.

Brock: Hm. True. Yeah. Well, what I was going to ask, wanted to ask about next, is this idea of making as many things as possible and rendering them as files. So devices as a file.

McIlroy: Yeah.

Brock: I'm not quite sure the whole universe of things covered by files, but that seemed to be, you know, a very important simplification and abstraction. Could you talk about, a little bit about that approach?

McIlroy: Yes, and this is course at-- this was written before the file idea was pushed to the limit as it has been nowadays. But it is a realization of what I mentioned before, that the stream is the thing. You don't care whether you're reading from a device or from a telephone line. You're just getting, you're-- but for most programs, it's a source, it's just a source of data, and in Fortran, there was a separate statement for reading from cards and reading from tape, and yet what's cards one day is put on tape the next day. So yes, this is, at least at the level of just reading from devices, to the greatest extent possible, makes it look the same regardless of what kind of device it is. Unfortunately, some devices have other properties that are unavoidable. Some have sufficiently many-- are sufficiently error prone that you have to take that into account.

Brock: I see.

McIlroy: Some devices have out-of-channel signaling, and so you have to be prepared for that, and that, all of that stuff, got buried over in another corner of the system called IOCTL, IO Control, and IO Control is the black sheep of Unix, you know, and--

<laughter>

McIlroy: --you'd rather not talk about it, you know, and it's got all the uglies in it.

Brock: So it's the exception that shows the rule.

McIlroy: Yes. Mm-hm.

Brock: We talked a little bit about the quick development and testing of programs. One that I will confess that I didn't really understand is about it being a self-supporting system. Unix is a self-supporting system.

McIlroy: It meant we, if you wanted a new version of the system, you didn't have to go off to another machine and make it and bring it in here. That's all. <laughs>

Brock: Oh, oh, I see. So you could use--

McIlroy: It had-- you could recompile the system and run it.

Brock: Oh, I see. I see. Okay.

McIlroy: Very trivial nowadays.

Brock: But not at the time.

McIlroy: Not at the time. At the-- often there was a-- you would-- there was such a distinction between code and data, and you had to stand on your head to make a new compiler, because while you're editing it it's called data, and when you're compiling it it's called code.

<laughter>

Brock: So this was-- okay. I can see. Now I understand. It's that you use a current-- you build the next iteration of the system within the system and this kind of idea.

McIlroy: As the PDP-7 had been used in Visual and Acoustics, they prepared programs for it on the 7090 and they ran them on the PDP-7. It was not self-supporting.

Brock: Right. Okay. I understand. Thank you for that. We talked about how in returning to this idea of Unix spreading outside of Bell Labs, particularly to universities under the university or the research license, it seems that that was a pretty rapid spread of Unix, particularly to the universities, and I--

McIlroy: Yes. By the 6th Edition, which was 1976. This was worldwide, and we had the Lions book written then. [John Lions, *Lions' Commentary on Unix 6th Edition, with Source Code*]

Brock: I was wondering-- I was curious to hear your reflections on how did that fact, that all these universities worldwide so quickly adopted Unix, how did that fact kind of shape the future of Unix? What was the consequence of that?

McIlroy: Well, it meant that coming-- it's just like IBM used to sell hardware to universities cheaply, and everybody graduating knew IBM and so it was the old, old drug seller who knew so well today's free samples are tomorrow's clientele.

<laughter>

Brock: Right.

McIlroy: And certainly the fact that grad students typically-- I don't think undergraduates met Unix so much in universities. Grad students came out imbued with Unix and they would want it wherever they went to work. So this certainly helped the propagation of Unix into the commercial world.

Brock: Hm. Did it also lead to kind of requests or pressures coming back into you?

McIlroy: Feedback. I wouldn't say much pressure. Within the spread within the Labs, it did result in back pressures. "Can you give us this?" and particularly in interprocess communication. You could see what was happening was each project would would be being written by people who'd been trained in a different college. One did messages, one did events, one did...

<laughter>

Brock: Right.

McIlroy: Did semaphores.

Brock: Yeah.

McIlroy: And, "Oh, we just got to have those." Interprocess communication was pretty amusing all the requests we got for it. We may not have got it all. Signals we did not get right, but we certainly didn't need all these different facilities.

Brock: Right. But the--

McIlroy: Eventually Linux does have them all.

Brock: Because I suppose people can just add them. <laughs>

McIlroy: Mm-hm.

Brock: But what kind of-- was there any really important feedback that was coming from this...?

McIlroy: Yes, yes, the internet.

<laughter>

Brock: So could you expand on that?

McIlroy: Certainly one of the seminal events in Unix history was when DARPA decided that the hosts for ARPANET, for the first ARPANET, would be Unix, and Berkeley was given the assignment of making it happen.

Brock: Mm, and this was adding the networking to Unix in BSD.

McIlroy: Yes. Mm-hm.

Brock: Yeah, and that was--

McIlroy: I mean, we-- some networking was done in Bell Labs, but the one that counted is the internet.

Brock: And how did that come back in?

McIlroy: We brought in Berkeley code, put it right in the research Unix. When we did that, the size of the kernel doubled overnight.

Brock: Really?

McIlroy: Yeah, it was big deal.

Brock: How so? Was just that part of--

McIlroy: Well, I think it's partly that Berkeley didn't have quite the same aesthetic in making things small, but actually connecting to the network is non-trivial. Running the whole protocol stack is non-trivial.

Brock: Wow. I hadn't realized it was that big of a component.

McIlroy: It really was a big deal.

Brock: Wow. Huh. And well, let me see. I was interested in your reflection on, not only did-- well, this wide and fast adoption of Unix in academic computer science, both kind of builds in life-long adherence to Unix, but I was interested in how it has made Unix occupy a kind of a central place within computer science itself, as sort of the paradigmatic operating system.

McIlroy: That-- yeah.

Brock: And I’m sure there are other ways.

McIlroy: That’s absolutely true. Yeah, that if you study operating systems, you-- some flavor, usually a small flavor, of Unix, is central to the course.

Brock: Yeah. I was just curious to hear your thoughts about that.

McIlroy: And we did have things like Mount Xinu[5] or just Xinu, and Minix, which were built specifically to be teaching systems.

Brock: Do you think that is in a sense why, it seems like some of the elements of the sort of Unix philosophy, which are so ingrained with or intimately connected with Unix itself, it strikes me that in a way, it has naturalized Unix and the Unix philosophy as the natural seeming approach to operating systems or computer science, because of its role educationally. You know, it forms--

McIlroy: Well, certainly for kernels.

Brock: Yeah.

McIlroy: For the library of user programs, we don’t seem to have had a great deal of effect. All the old ones are still there, but people make big things nowadays.

Brock: I’m not--

McIlroy: Git, for example, is enormous, compared to SCCS, whereas SORT has barely changed.

Brock: Hm. And?

McIlroy: And I’m not sure why Git is so big. Yeah.

Brock: Right, I see what you're saying, that that philosophy hasn't transferred over into some of this application layer stuff.

McIlroy: Yeah, and the major and widespread applications.

Brock: Which are, right, yeah, diverge from that philosophy. I see. Another question I wanted to ask you about was about kind of the culture around the Unix group and the Unix Room. It

[5] I should have said plain “Xinu” instead of “Mount Xinu.” Xinu was a small academic model. Mt. Xinu (a reversal of Unix TM, our lawyer’s recommended usage) was a commercially supported BSD derivative. I don’t know whether my pronunciation of mt was widespread. In any event it should be spelled “mt” or “Mt.” Clarification by Doug McIlroy.

seemed like it was a very-- well, I wonder if you could describe that, what that culture was like, and its relationship to the broader Bell Labs culture. Jut in terms of the working environment.

McIlroy: Well, the fact that there was a Unix Room, was, of course, there were labs everywhere. But because what we were doing was working at terminals, it was like having a shared office. At other labs, you would have office work, and you would have lab work at the bench. But you wouldn't have people sharing all their lives. Like offices were typically private. So that was a-- I mean, we had our choice. We all had our private offices, but several people sort of lived in the Unix lab. And especially, the later lab, I think there were typically you'd find half a dozen people in there whenever you went in. So this was different from the rest of the Labs. Certainly the business we already touched on about it didn't much matter who played with the code, he who touched it last owned it, grew out of this fellowship, if you would. And we had one employee once-- generally fellowship was the right word. We had one employee once who was more of a locker-room type, and that really-- that put a little strain on the Unix Room. It didn't change it, but it's clear that you don't necessarily get it right just by putting people together.

Brock: Right, so you're talking about the kind of mix of individuals who were there that set the culture.

McIlroy: Yes.

Brock: I was wondering if there was a similarity, especially in the '60s and '70s to-- if it was similar, if the Unix Room was more similar to sort of the working culture of the computer center, or in the hacker culture, such as it was, that it would be--

McIlroy: The working culture of the computer center is already a well-organized outfit with prescribed duties. And that didn't exist at all in the Unix Room.

Brock: True.

McIlroy: We didn't have operators and customers, engineers and clerks. File clerks.

Brock: Right, right. I guess it makes it more of--

McIlroy: This was even true even after we had our own computer group. There were people who spent more time than-- most of us wouldn't go flicking switches. Mostly, because it hurts people outside. But even then, the folks that went into the computer room might work outside, too.

Brock: Right, right. Let's see. Well, I was-- I had a couple of questions about how, as you are participating in this whole Unix development, your personal work leading the department. I was curious about to hear about while this whole Unix phenomenon is happening, it seemed like you were doing a fair bit of work personally representing Bell Labs computing to outside groups.

McIlroy: We all did.

Brock: You did, yeah. <laughs> But I was interested to hear about some of your experiences, like I saw that you did quite a bit of work with the National Research Council. If just--

McIlroy: Yes, that was not an unusual thing for Bell Labs personnel to participate in NRC reports. I guess I did about three of them.

Brock: Was there anything in that that really stood out for you, that you found particularly important?

McIlroy: Well, working on the first report on computer security seemed very important. And I'm still sort of disappointed that some of our main suggestions have not yet come to pass. One of the things we talked about was the end user license agreements are unlike those for any other product. The manufacturer has no responsibility to make something that works. It's perfectly all right to put out totally untested software, right? And no repercussions on it other than maybe you get a bad reputation in the market. And security still falls to the bottom of the list of most places. The few big places like those who were running data centers, they really worry about it. But people doing the Internet of Things don't give a damn.

Brock: And so that would be almost kind of a tort reform or something.

McIlroy: Yes! And there is the-- what's it called? The Uniform-- I've forgotten the name.[6] But ordinary products that typically there's an expectation that they are fit for their purpose.

Brock: Yes.

McIlroy: And under law!

Brock: Yeah, right. <laughs>

McIlroy: And the computer industry-- or the software industry has managed to escape any legal consequences for gross error.

Brock: It's interesting that one of the primary kind of approaches to improving security is improving kind of the motivation for people to work on it. More so than any particular technical thing is just to motivate the effort.

McIlroy: Mm hm.

[6] Uniform Commercial Code. Clarification by Doug McIlroy.

Brock: Hm! That's fascinating. Well, let's see. I wanted to ask about the early '80s and the whole breakup of the Bell System. And I was curious about the impact of that on Bell Labs and your department, on the Unix effort at the Labs.

McIlroy: Well, it was too bad to have departments ripped apart and half of them go to AT&T and half of to Lucent. As it turned out, in computer science, you actually had your choice. In computer science, a minority went to AT&T. In mathematics, damn near everybody went to AT&T. And the Bell Labs math department had to be rebuilt from scratch. The style-- but I don't think it impacted the flow of work.

Brock: In your department.

McIlroy: Yeah. It did eventually, because Western Electric's market became very different. And Bell Labs eventually makes it-- got its support by making-- by developing products for Western Electric. And the split-off changed that entirely. But I retired soon enough that I didn't really see the effects. Nowadays I understand that people identify themselves as working on projects. And we didn't, generally speaking, we did not do that sort of work in my time.

Brock: Well, one thing that I forgot to ask, is was it-- well, I'll just go through the questions. It seems that from my research that it was in 1986 that Dennis Ritchie became the head of the department where you became a distinguished member of the technical staff within it. I was just curious about was that something that you had desired to do, or what was that?

McIlroy: Yes, I did, I did.

Brock: Yeah, can you talk about what your thinking was by that time?

McIlroy: There were two things. I'd done it for a long time. But I also realized that younger people in the center often had different kinds of motivations than I. In particular, they were much more oriented toward interactive graphics and so on, which I didn't much-- I was not the right person to be managing these folks. So I put in my time and got out.

Brock: So you were recognizing that your interests were diverging with some of the newer members of the technical staff.

McIlroy: Yes, yes, yes. They were just as good as the older members, but they were interested in different things.

Brock: Well, was it under Dennis Ritchie then your department began working on Plan 9? Was it--

McIlroy: Plan 9 started around 1990.

Brock: Oh, okay, so it was far after, yeah.

McIlroy: Plan 9 was-- by the time I left, my desk had Plan 9 on it.

Brock: And were you a part of that--

McIlroy: No. The only thing I contributed to Plan 9 was the ellipse drawing routine. Oh! I'm sorry! No, that wasn't the Plan 9, that was to the Blit.

Brock: I mean, it sounds like several of the-- it sounds like Dennis Ritchie and Ken Thompson, in particular, were both-- Rob Pike.

McIlroy: Pike and Thompson would be the central people.

Brock: Were the central people in Plan 9.

McIlroy: And Dennis came in-- Dennis did some very important things, but he wasn't one of the instigators.

Brock: Okay.

McIlroy: As far as I know.

Brock: But this was an idea to evolve some new operating system ideas.

McIlroy: The premise of Plan 9 was to make operating systems-- at least to make a seamless multi-computer operating system. As soon as you go down into a network in Unix, when we talked to the internet, we had a different interfaces-- it wasn't like talking to a file. We had sockets rather-- which were a completely new thing. They-- but Ken and-- Rob and Ken set out to do was to arrange-- it didn't matter where a process was. Processes didn't have to all be in one machine. They could be in one machine or multiple machines, and they’d communicate with each other exactly the same way. They did this by opening up the protocol for operating data streams. And the protocol became overt, and not something hidden inside the operating system. They did a lot of other things, too. They added things like union file systems and name spaces. And taking the notion of files much further than it had been before. Most data queries to the operating system like, "What are the properties of this file?" came back as if they were those properties had been stored on a file somewhere. They were actually synthesized on the moment, but it came out as text. These things were all-- all of this, except for the inherent distributed nature of the system, were snapped up and put into Linux, but the basic premise remains unrealized in the wider world.

Brock: Huh! That's interesting that all those things were put into Linux. Which was really, I guess sort of really gaining steam as Plan 9 was coming to fruition. Yeah.

McIlroy: Yes, at the same time.

Brock: Yeah, they're co-extensive, yeah.

McIlroy: Plan 9 even went so far as when you trans-- we had big-endian and little-endian machines, when you transferred data between the machines, it got re-endianed <laughter>, or whatever the right word is.

Brock: Oh, the operating system just took care of that. Hm.

McIlroy: Mm hm. They did a fairly remarkable job of hiding the byte order of machines.

Brock: And was this kind of the logic behind both Rob Pike and Ken Thompson's subsequent move to Google, right? Because there's a huge distributed system there? Was there a connection there?

McIlroy: No, no, I don't think so.

Brock: No?

McIlroy: I think they moved to Google because the-- as the reality of Lucent's market, compared to the Western Electric's captive market, set in, the Labs became a less free place to work. Google had a nice research outfit. However, you'll find that as far as I know, Pike hasn't published anything since he moved to Google. Whereas, he published fairly frequently at Bell Labs. So the Google culture is not the same as he knew before. But they all seemed very happy out there. Rob is now in Australia.

Brock: Well, I guess then, I wanted to talk to you a little bit about your decision to leave the Labs, and engage with Dartmouth.

McIlroy: Well, leaving the Labs was I became 65. Of course, it was legitimate to stay on, but and why did I come to Dartmouth? All along Barbara and I have said we'd like to retire to a college town, and we both grew up in college towns. And when Peter went to college, he went to Dartmouth, and we got to know the place, and it seemed like a very nice territory to live in. And I did know the Chairman at Dartmouth. And four years before I retired, I initiated the inquiry of, "Could you put up with an old guy coming in and hanging around the Department?" And they said, "Yes." And even though that Chairman died before I moved here, the next one continued to be receptive. So that's how we came here. Perhaps slightly related is the fact that the Morris family had come to Hanover, too, but had not established relations with the college. They came here because their Great Aunt had had a summer place up here and she[7] had got to know it in her childhood.

[7] Bob Morris' wife. Clarification by Doug McIlroy.

Brock: Right! <laughs> Oh, so you had friends.

McIlroy: So we did have one person, one very close friend here, just down the street a few couple of miles.

Brock: And how have you enjoyed your association with the Dartmouth Department?

McIlroy: Very much, until last year, the department gro-- this year, right, the department has grown so big that I got squeezed out of office space. I have a terminal in a lab somewhere. So that means that I'm falling out of touch a bit. But just yesterday I met with a student who I'm advising for a senior thesis.

Brock: Oh, excellent!

McIlroy: So I've still got a connection.

Brock: Excellent.

McIlroy: And I've got the run of the library, and all that stuff. And the department file system. So I don't have to pay for internet service.

Brock: <laughs> Perfect! And has it been the case-- I know for a lot of places with some similarity to Dartmouth, that Computer Science is becoming the dominant undergraduate major for people. Is it a similar--

McIlroy: The Computer Science Department is squeezed because of that. It is not so-- it is not the biggest major on campus. I think Engineering is. Because Dartmouth doesn't distinguish among engineering.

Brock: Oh, yeah?

McIlroy: There's no sub-species, it's all one big department!

Brock: Oh, okay. <laughs> But its popularity amongst-- as an undergraduate major is growing, and therefore the department is--

McIlroy: It is, yes, yes. And so you end up with courses with a half a dozen TAs, you know. But not 30 or 40 TAs like some schools.

Brock: Right, right. Well, I guess, I wanted to-- let me just check the time. I guess, if I could, I wanted to ask you, well, one, about--

McIlroy: Maybe one more thing about Dartmouth.

Brock: Yes.

McIlroy: When I joined the Labs, I thought that I would stay there a few years and then go join academia. But it was so much fun that I stayed, and I became real-- I actually realized that we had more academic freedom there. We didn't have to make grant proposals.

Brock: Yeah.

McIlroy: I get here, and at the Labs you had the-- I find that that's a really significant concern, particularly for the untenured Assistant Professors. They have to put their nose to the grindstone. And they've got to get enough publications out to make tenure. And at Bell Labs, if you had some interesting little question, well, put it on the flagpole, walk down the halls, people come out of the offices and give you ideas. Here, if you drop into an office of an Assistant Professor, they're too busy to think about anything! So it's actually less collegial than the Labs were.

Brock: I absolutely believe that. Yeah, I mean, that seems-- that is what I observe amongst people and my colleagues, and especially so, I think, for Bell Labs in the period that you enjoyed being there, which was, you know, the opposite of resource constraint, and it was just a phenomenal situation. And also, well, that you hear it time and again from people who worked at Bell Labs at that time, the scope-- the freedom of action and the scope of action that people had, and in a way, just the confidence that people would do something good that would benefit the Bell System as a whole was-- well, it was a confidence that was born of seeing it happen time and again, but just this idea of, "Keep in the back of your mind that you work for the phone company," seems to have been adequate to the task, you know?

McIlroy: That's exactly right. And the fact that you really-- if you needed help, people were always willing to give it.

Brock: Mm, well, I suppose in such a-- in some ways, in such a successful and secure environment, yeah, that people are free to share. Just as we were talking about with the people taking over this this tool or that tool. And not worrying about credit or territoriality in that fashion. You know, if you're a manager of a department, and you don't have to look at the budget, it's a very different environment to these young people having to compete against one another for the same pool of NSF funding. So it's a totally different environment.

McIlroy: That's exactly true.

Brock: I did want to ask you just a little bit about your interest in computing currently. You know, what your sort of working on today, and then just ask you to-- and then I want to close by just asking you to reflect on the kind of incredible success that Unix has had. So those were just the two things I wanted to talk about.

McIlroy: I have a lot of fun these days with-- or have had a lot of fun for the last decade or so with Haskell, which is a lovely programming language, more like math than like-- than most past languages. And in particular, revisiting stream processing. The old algorithms like the Sieve of Eratosthenes, the stream version becomes absolutely beautiful in Haskell. Two-line program. And on my website, I have a piece called, "Power Series in Ten One-Liners," Haskell programs for manipulating about formal power series. If you go look at Knuth's books, you've got a whole chapter on this subject. And most of the chapter just describes these things that I can-- that nowadays you can write in one line of code, particularly reversion of power series, taking the functional inverse, that is. One line of code versus in Knuth a half a page of-- and it's a one line of code that has no subscripts in it! That's lovely, in Knuth it's half a page of pseudocode, can't run it, full of subscripts. So this is just a remarkable improvement in expressivity. Especially in this, and a lot of fun. And I'm currently writing a little piece on macros again, which is going *way, way* back, you know? Doing LISP within a macro processor, this just illustrates that macros, on their own-- and throwing out all the non-macro accoutrements that macro processors usually contain. Macros per se are Turing complete, and making LISP will illustrate that.

Brock: Oh.

McIlroy: In a nice way.

Brock: Interesting.

McIlroy: So I'm writing that. Now the other thing you asked.

Brock: Yes! I'm just curious as someone who was there from at the before the very beginning of the whole Unix tale, you know, now seeing, well, the ubiquity of Unix in a way, either as Unix, as in-- you know, as Linux, in all the servers and also in all of these-- you know, all of the Apple-made devices, you know?

McIlroy: Well, it's very nice, personally, to know that I can use any of these systems. The first time I experienced that was the Cray. There was zero learning curve to get things running on the Cray. Of course, there's plenty of learning curves to use the Cray efficiently, but just write an ordinary C program, and run it over there! <laughter> So and whenever I go to an Apple machine, I just dive into the terminal, and I know what to do.

Brock: Right.

McIlroy: I don't have to know what all those icons on the side mean. <laughter> It's nice to-- really nice to know that the Unix model can be found everywhere. But I wonder when it becomes blown up to the extent that this was commented upon at the celebration at Bell Labs, this is like saying, "There's a little Mercedes inside of every car, because Mercedes made the first-- that Benz made the first automobile." <laughter>

Brock: Right, right.

McIlroy: It's rewarding to know that we found enough-- I shouldn't say "we"-- "they" found such a permanently useful underpinning for the way you would talk to a computer. What-- it's disappointing a bit in that Plan 9 didn't-- that the basic premise of Plan 9 remains to be done. I think we need it. And I guess I'm too much out of touch to know whether anybody else has got another approach to the same problem. I've been very impressed over the years with L4's microkernel approach, not with Mach’s microkernel approach. Their microkernel is bigger than our kernel. That just didn't seem to be right. And wondering whether the L4 analysis is inherently too inefficient to live on.

Brock: Hm.

McIlroy: But I would love to see things-- see partitioning of the kernel as L4 tries to do. The-- in one project that I consulted on within the Labs, we realized that distributed computing was safer than computing on one machine. Because things that weren't supposed to talk to each other, couldn’t! Or weren't supposed to see each other. And L4 has this attitude toward-- this design approach.

Brock: Hm. That's interesting. Well, I think I have asked all the questions that I had intended to ask. At least, that I had written for this afternoon. I'm sure I'll go away and think of a whole another list of things, but do you feel that there's some large area?

McIlroy: You've done very-- as a sideline, I could say a bit more about astronomy, but it really-- that's-- we're amateurs at that, and it's just been play for us. But essential to the history of computing, no.

Brock: Okay! <laughter> Well, maybe then I'll switch these off, and this will be a good time to break.

END OF THE INTERVIEW