Oral History of Frederick “Fritz” Trapnell

Interviewed by:
Jim Strickland

Recorded: May 27, 2014
Mountain View, California

CHM Reference number: X7177.2014

Jim Strickland: My name is Jim Strickland. And today I am interviewing Frederick Trapnell. This is March 27, 2014. And you prefer to be called “Fritz.”

Fritz Trapnell: Yes, I prefer Fritz.

Strickland: So I will do that. Fritz, let me start off, if I may, by asking a little bit about where you were born, where you grew up, and that kind of thing.

Trapnell: OK. I was born in Washington DC. My father was a naval officer. And my mother was a Californian. She was from San Diego and had moved back to Washington when his duty transferred. So I was born there.

But I actually grew up in San Diego, in Coronado mostly, though I went to high school back East. And then immediately after high school, in 1950, I ended up in the Marine Corps during the Korean War. It was as the Korean War broke out and the Marines were good enough to send me to an aviation electronics school. So that's where I first got into electronics. I spent two years on active duty and then went to college at Caltech in Pasadena.

During the summer vacations, I worked for the Charactron Division of Convair in San Diego, under the tutelage of a wonderful man and outstanding engineer, Fred Hammann. In the summer after my freshman year, I was a electronics technician; but after my sophomore year, Fred promoted me to junior engineer, and I was engaged in circuit design studies.

I did my undergraduate work in physics. I was in physics long enough to discover that I was not going to make a physicist. And so I stayed on for another year and got a master's degree in electrical engineering. So that's how I get started.

Strickland: Excellent. And I'd like to talk to you more about your dad. But maybe we can do that later.

Trapnell: OK.

Strickland: But was he an influence on your life at this point?

Trapnell: He and my mother were divorced when I was three. My mother remarried a marine officer who was a big figure in my life. And I didn't really get to know my own father until high school time when he

was on the East Coast, and he, for my sins, insisted that I go to school back East, which I did. I was in a boarding school, and I began to get to know him on vacations and during school breaks when I'd go down to Patuxent River, Maryland, where he was stationed. So I began to get to know him then. He was difficult on both his children. And it wasn't until a lot later that I got to know him very well and we got to be very close. So he was a big influence later in my life, but not in the early days.

Strickland: OK. But your stepfather, the Marine, may have been a big influence.

Trapnell: Yes, he was a big influence.

Strickland: How did he influence you?

Trapnell: Well, he was quite different from my father. He was a very physical kind of person. I learned a lot about fishing and hunting and skin diving. And he also had a lot of integrity, so I learned a lot about honesty and integrity. My father was the same way. But my stepfather had the most influence when I was young. So yes, he was a big influence during my growing up time.

Strickland: Terrific. But you ended up at school at Caltech in Pasadena. And you got a masters in electrical engineering.

Trapnell: I did.

Strickland: OK. And I think you said you had your first exposure to a computer there.

Trapnell: Yes, I did. My first exposure was a course in logical design I took as a senior, which fascinated me. And I thought, Wow! This is really an interesting application for electronics. Up to that point I had decided I was going to be involved in electronics because that's what the Marine Corps had started me off in. This was the most interesting application of electronics that I could see. And it, of course, was very new.

So when I went to graduate school, I got a research assistantship to maintain Caltech’s first digital computer, which was an ElectroData 205. They'd had an earlier IBM calculator that I can't remember the number of.

Strickland: Might it have been an IBM 610?

Trapnell: It might have been. It stood about 6 foot 10 inches high, as I recall. And I/O was paper tape; well, so was the ElectroData. So that was my introduction to computers.

Caltech also got the first Librascope LGP-30, which was designed by a graduate student at Caltech. Caltech, I think, sold the design to Librascope, and, in return, got the first machine, which is never a great thing to do. First machines are usually have problems. And that one did, though it worked pretty well. And so I did my first programming on those two machines, the Librascope LGP-30 and the ElectroData 205.

Strickland: OK. So that was both programming and the actual hardware maintenance?

Trapnell: Yeah, I had to do the actual hardware maintenance on the ElectroData machine, but I programmed both of them, I don't know if we talk about the programming. But it might be of interest.

Strickland: Well, they were both-- The ElectroData was a drum machine, and the Librascope was a drum machine.

Trapnell: And the Librascope was a drum machine as well. The ElectroData 205 was very comparable to the IBM 650. They were at the same time, similar technologies. The Librascope was a small machine. It was supposed to be a desktop, the nearest thing to a desktop.

As I recall, it had 4,096 12-bit words. And my professor asked me to write a decimal floating point package for it, which I did. And it took me, perhaps, four to six weeks. And he kept needling me about what was taking so long. And when I got through with it and began to test it, I had to tell him that of the 4,096 words that were available, I had used 4,025 of them to write this package. And he hit the ceiling. He thought this was ridiculous.

So he took my code and went away for about two weeks. And he came back, and he said, “I found an error.” And I fixed it. And it now takes 4,030 words. So that was a real challenge.

Strickland: But not exactly a sub-routine. Right?

Trapnell: Not exactly. This package filled the whole machine. So you'd key in a set of decimal numbers and the operation you wanted to perform on it. And it would do the calculation and give you the answers.

Strickland: You know that we have a Librascope, right?

Trapnell: Do you?

Strickland: You've seen that?

Trapnell: I haven't seen that one. I should take a look at that. Yes, that goes back some number of years. So that was my first introduction to computers.

Strickland: OK. So you were in software before you ever got, in effect, out of grad school. And then you went to work for IBM. How did that happen?

Trapnell: Well, the IBM recruiters came out to Caltech. And I had decided I wanted to go to work in, if you will, civilian electronics and not in military electronics. I'd been very closely associated with the military. But I was interested to see if there was really going to be an application for electronics in civilian business.

So I interviewed with AT&T Bell Labs and IBM and got job offers from both. And IBM was in Poughkeepsie, and I elected to go to Poughkeepsie. But that was a result of being recruited in college.

Strickland: And of course, IBM had the old system stuff at Endicott, and the new computer stuff was mostly being done at Poughkeepsie. Is that right?

Trapnell: That's probably true. Most of the computers-- I've forgotten where the 650 was developed because it was before my time. But it was likely Poughkeepsie. But Poughkeepsie certainly had all of the so-called big machines, the 701, 702, 704, 705, and then on and on, of course, to the transistor machines, the 7090, et cetera.

Strickland: OK. So when you were at Poughkeepsie, what did they have you do first?

Trapnell: Well, I interviewed several places. Stretch [IBM 7030] was being developed at the time. And that was an interesting proposition. But the Special Engineering Products Division appealed to me because it was involved in a lot of different kinds of products and looked like it would offer a lot of different challenges. So I elected to join that division, a very small division that didn't last very long, about three years. And then it merged into the Advanced Systems Development Division, ASDD. Anyway, that's where I started. [FT Note: The division was headed by Jerry Haddad, a tough-minded driver who had earlier been part of some key developments in computers at IBM. The development engineering arm was directed by Phil Jackson, an old IBM hand and a wonderful guy, and my first boss, who reported to Jackson, was a man by the name of Tony Sanchirico.]

And I did a number of things. One of them was some work on a B-70 program. They sent me up to, Owego, near Endicott. They had a military lab in Binghamton, as I recall, that was an adjunct to the one in Kingston. And I went up there to work on some of the bomb/NAV system on the B-70 briefly.

Then I worked on how to digitize waveforms on a cathode ray tube so that you could analyze them offline. And I designed a machine for that, got a patent for that. Then I don't remember exactly how it happened, but I got very much interested in computer communications. How does a computer talk to a telephone line?

IBM had already built the Transceiver, which was the card system over a telephone line. And that work was done in Poughkeepsie. Well, we were interested in how to get a computer to do that. The Sabre system was already being designed. And they used special controllers on their 7090 to do that. The SAGE system was in operation. And it had the same capability of talking to telephone lines, but it wasn't very general purpose. These were very specialized to those machines and to operate in certain specialized ways. And the question was, how do you get any old computer to talk to telephone lines? And more particularly, how do you get a computer to talk to 100 telephone lines concurrently? That seemed like a big number in those days.

So I got a new boss, Gene Potok, and I started the work on that problem. First of all, we were concerned with what to use for modems to modulate and demodulating signals on telephone lines. The phone company was very possessive about who put signals on their telephone lines and with what kinds of signals they were.

Strickland: AT&T was still in the monopoly state at that time.

Trapnell: AT&T was still in a monopoly, yes. They weren't nasty to us, but they were firm. And we talked to them a lot about their modems, about the performance of their modems. And in those days, 600 bits per second was the fastest thing they had.

The card transceiver operated at something like 200 bits per second. And the the four-out-of-eight frequencies signaling system it used just wasn't going to work for higher speeds. AT&T had a frequency shift modem, which operated at 600 bits per second, and they were talking about going to 1,200. That was going to be a big leap forward.

About that time, two guys at IBM in San Jose, Emil Hopner, Harold Markey, invented a new way of signaling on a telephone line, phase shift signaling, which, by the way, is standard today. But they invented that technology in San Jose. And it offered potentially much higher speeds than the phone company was going to offer. So I remember spending time out here in San Jose with them and trying to convince the phone company that maybe ought to look at the modem that we had.

And I've forgotten how that ended up. But for me, it morphed into an assignment that I wanted very much, which was to figure out how to build a general purpose controller to allow big computers, like the newly arrived 7070, seven-oh-seventy, to talk to a lot of telephone lines concurrently. The 7070 was a new machine that was coming along. 7090 was already there. I think the 7070 followed it, and then the 7050.

So I worked on the design of that and built at least one prototype of it. And out of that came the first IBM front end processor, the IBM 7750. It was a stored program controller that sat on a tape channel. And the computer could talk to it like a tape. These computers did not have interrupt systems. So you couldn't, from the outside go in and say, hey, I've got something here, which all the later computers, of course, did.

So the internal programming had to recognize that periodically it would have to poll this unit sitting outboard. But the unit outboard did all of the acquiring of data, transmitting of data, and transmitted between the computer and the 7750 transmitted buffers of – I've forgotten what size - I think they could be variable. Anyway, it was the first communications controller of its type in IBM and possibly the first front end processor product in the world.

Strickland: And it supported a lot of different device types and different communication protocols.

Trapnell: Yes, it did. It supported everything from teletype to the 11-bit signals used by the card transceiver. It supported various speeds, selectable. And you could have different lines working at different speeds. Analysis of receiving signals was all done digitally. You'd come right off the modem, straight into the 7750, and it digitally looked at the incoming signals, sampled the input.

And we had worked out the digital sampling method with three guys in San Jose, Markey and Hopner as well as John McLaughlin, who was one of the smartest guys I ever met. I think from Information Theory you can prove you need to sample at least twice per bit. They had done some work that demonstrated that if your signals are mixed with a lot of noise, you need to do it much more frequently. So we adopted a standard that bits have to be sampled at least seven times per bit. And on the 7750, by program, you could adjust each connection to the frequency used on the line it served.

Strickland: It was a controller in the sense that your job was to gather data from telephone lines and pass it to a mainframe.

Trapnell: And the other way around.

Strickland: And the other way. But it was programmable.

Trapnell: It had its own stored program.

Strickland: So in that sense it's a computer.

Trapnell: It absolutely was a stored program computer. I remember that Fred Brooks was a part of a group that was called in to do an engineering audit of this machine (we did such audits in those days), and he was very interested. As I recall, he said, “I stayed up all night reading this manual (which I had written) and this is a really interesting machine because it is so different from anything we've done before.”

It was a stored program computer, a front end computer in the classic sense that it was a little computer that sat outboard. Other people, of course, built front end computers later. CDC was famous for front end computers.

Strickland: So this must have been what, about 1959 or 1960?

Trapnell: This is 1959.

Strickland: And is that the first time you met Fred Brooks?

Trapnell: That was the first time I met Fred Brooks, yes. I mean, I knew of him. He'd been on the Stretch program, and he showed up on this audit team.

Strickland: Later, we're going to run into Bob Evans. Did you know Bob Evans at this time?

Trapnell: Not then. I knew Bob Evans later, quite well.

Strickland: OK. So 1959, '60, and you were a member of a team or you were leading a team.

Trapnell: I was leading a team to design the 7750, when I ran into a bit of a personal problem that led me to resign from IBM; I planned to move back to California. After I'd handed in my letter of resignation, IBM counter-offered me a position in Paris. And in the end, we decided to do that. So I transferred from IBM in Poughkeepsie to IBM World Trade, spent a little time in New York, and then moved to Paris as part of the IBM World Trade laboratories staff, of which there was one person when I arrived, and I made two. My boss was Hal Martin, whom I had met earlier because he was one of the recruiters that came to Caltech to recruit people when I was interested.

Strickland: But are you saying you helped start what was basically the second World Trade laboratory?

Trapnell: No. The World Trade laboratories were already established. I'm not sure if all of them were, but the one at Hursley in England had just started in about 1957 or '8. There was a Paris lab that had been in operation since before World War II. There was also a lab in Boeblingen, Germany, that I believe went back about that far as well. I'm sure the one in Sweden was already established. So there were at least four labs. And perhaps there was another one. Well, there was one in Vienna later, but I'm not sure it was there then. Anyway, there were at least four.

IBM management said, we've got these labs over here, and their building stuff for Europe. We've got to get these productive for the worldwide market because we're in a worldwide business. So the mission of the World Trade Laboratories was to bring this group of European labs into the worldwide product development operation.

So in Paris, my initial assignments were to help Hal to do some of those things. But I also had a particular interest and some background in airline reservation systems, because of my association with the Sabre reservation system and my work on the 7750. European airlines were just starting to think about automated reservations, and IBM World Trade trying to get the airlines interested, so I helped out on studies and proposals for several of systems.

I guess the most notable one was with KLM in Amsterdam, where I went and spent, I don't know, several months up there. And instead of me showing them what to do, I learned about airline reservation systems from a senior IBM systems engineer there by the name of Henk Asser. Henk was one of the smartest guys I'd ever knew. I had previously met him in Poughkeepsie, but I was fortunate enough to be able to work with him for several months. And he taught me the ins and outs of airline reservation procedures and how airline reservation systems actually had to work. So that was a very, very fruitful experience for me. Then I went over and spent a similar amount of time in Ireland, working with the IBM rep there, Dr. Paddy Doyle. We worked on a proposal for Aer Lingus.

I think eventually IBM did sell systems to both these airlines. We were trying to build airline reservations using 1410 computers at that point.

Whether those airlines built something before the 360, they might have. But the 360 made a lot of changes. The 360 system was really much better suited for that kind of thing. Whether there were actually any commercially successful systems before then, I can't say.

Anyway, I was in the Paris office of IBM World Trade laboratories from the end of 1960 until about June when Hal Martin got a new boss, Gardiner Tucker. Gardiner was really concerned about the Hursley laboratory, the management and direction there. He made some management changes and removed the

laboratory director, and asked me if I would take that position. Well, I was a 29-year-old kid and, of course, was so flattered I couldn't say no. Whether it was a wise thing for me to do, I don't know. Well, yes, I do know a little bit. I'll come back to that.

Anyway, I accepted that position and took over at Hursley, as head of the laboratory, in July 1961.

Strickland: And how big was the laboratory, and what were some of the projects that you were doing?

Trapnell: About 400 or so people. It was a substantial lab and a fairly heavy expense for IBM. The main thing they were working on was the SCAMP computer. Now SCAMP’s claim to fame was to be the first computer in IBM to use a read-only control memory to store all the control operations (micro-code) for decoding the computer’s instruction codes. Prior to that, instruction decoding was all hardwired.

Strickland: Microcode? Is this the first microcode?

Trapnell: This is microcode. So IBM SCAMP put microcode on the map for IBM, perhaps for the computer industry.

Strickland: Ah, very good. And obviously, that was a key to 360.

Trapnell: It was very important to 360. And it was the SCAMP that demonstrated its utility. IBM didn't invent this idea. I think the idea of a micro-coded computer was invented by Wilkes at Cambridge. And I think they may have built one or two. But this SCAMP machine was the first proposed product.

Now, SCAMP was not on the worldwide product list. It was potentially going to be a computer in Britain, maybe in IBM World Trade. Nobody quite knew. Along with that product, was the development of the read-only memory technology needed to store the microcode. So there was the basis for microcoding technology and a computer that demonstrated this technology. I was a 29-year-old kid, an American, walking into this scene in Britain, where almost everybody was older than I; certainly the senior managers were all older.

John Fairclough, was well known there; he and I were the same age. John's has passed away, unfortunately. But he and I were the same age, and he was one of the younger managers there. So I was not welcomed – Let's put it that way. Fortunately, that didn't last long; I was accepted pretty soon.

I asked to have the SCAMP product audited – looked at as a worldwide product. Was this going to be a worldwide product? Well, we had people in from the US and Europe, and a small group, of which I was a

member on the audit committee, to decide whether this thing had a possibility of becoming a worldwide product. And we turned it down. We said no.

It didn't fit with the 7000 series. It didn't fit with the 1400 series. It didn't fit with anything else that was coming down the pike or beginning to be introduced at the time. So we had to kill it. And this of course was a severe blow to the Hursley lab and to John Fairclough, the SCAMP Project Manager. It was one of the hardest things I had to do.

The SPREAD committee [Systems, Programming, Research, Engineering and Development; a high level group formed in 1962 to evaluate current systems and recommend future plans], was the foundation committee for laying out the ground rules for the 360 series. Fortunately they were just getting started, and with the help of Gardiner Tucker, my boss, we got John Fairclough on that committee. So one big personal step forward for me was to put John in the middle of the advanced planning for the 360. He was detached from the lab for two or three months and came to the US to serve on that committee full time.

Strickland: And had he worked on SCAMP? And he knew microcode?

Trapnell: Yes, since he had been manager of the SCAMP project, he was very familiar with all its technology and all of the stuff that had gone into it. So he brought with him the knowledge of read-only memories and microcode. And he was able to convince the committee that this was the way to go in the design of all but the very highest performance computers. So that's how, I think, the read-only memory and the microcode got introduced in the 360, because John was there.

In the meanwhile, we began doing feasibility studies to support for some of the ideas coming out of that committee. I split off the read-only memory group from the SCAMP project (painful) and set it up as a separate technology activity under the senior manager Peter Atkinson, and under him, Tony Proudman, who was a key developer. Tony was brilliant, a leader in the evolution of read-only memories at IBM, and was the inventor, I think, of the balanced capacitor read only store (BCROS).

In addition to their development work on the transformer read-only store, used in SCAMP and later the 360 model 40, they began work on the balanced capacitor read-only store, which promised the higher speeds needed in the models 50, 60, and 65. So this little team was inventing and refining the technologies needed to make microcode work across most of the new product line.

And a man named Tony Peacock was one of the bright young computer architectural people who had been instrumental in SCAMP. He was even younger than John and I and he was brilliant too. He had a little team that did feasibility studies in support of design ideas coming out of SPREAD committee for arithmetic units. So we kind of got our lab into the middle of the 360 evolution during that period.

Strickland: Have you ever mused on what would have happened if Fairclough with his SCAMP and micro-program knowledge hadn't gone to SPREAD?

Trapnell: I haven’t thought much about that, but microcode would have been a longer time coming.

Strickland: Well, it sounds like serendipity. It sounds like a really excellent choice. So, you were running the Hursley lab. And we're in '62 now. And SPREAD says, we're going do this.

Trapnell: We're going to do this.

Strickland: And Hursley will get development responsibility for the model 40.

Trapnell: I think John lobbied very hard to get development responsibility for the model 40, and he succeeded. That was the point, I think, where John made good friends with Bob Evans. John and Bob were very close throughout the rest of their careers, as far as I remember. They were close friends, and it was a good thing for us because Bob was put in overall charge of the development of 360.

We [the Hursley Lab] got the development responsibility for model 40, under John Fairclough, and also for initiating the development of all of the read-only stores that went into most of the 360 models that used microcode.

Strickland: Really?

Trapnell: The ROS we developed for the model 40 came straight out of the SCAMP design. It was a transformer read-only store (TROS) – very reliable, very stable, and huge signal-to-noise ratio. Fortunately, we managed to get it into the controllers, and because it was so reliable, that design made the disk, tape, teleprocessing, etc. controllers – extremely reliable. We had some technologies under

development that weren't so reliable. And some others that were very fast. One of them was the balance capacitor read-only store, or BCROS, which was faster, more complex and more expensive per bit than a TROS. It was invented and initially developed at Hursley and adopted by the model 50, 60 and 65.

Strickland: Did you do the CCROS [card capacitor read only storage] for the model 30?

Trapnell: Yes, that’s the one we found to be not so reliable – the signal to noise ratio was too low. An important element in any system where you're trying to detect signals is, how big is the signal compared to all the noise that comes with it. Well, I think they eventually improved the signal-to-noise ratio of the CCROS, but at the time we were getting 1 1/2 to 1. That means the signal is 1 1/2 times as big as the noise. That means you have to work hard and pay a lot of money to build amplifiers to reliably detect that signal and not be confused by the noise.

With a transformer read-only store, the signal-to-noise ratio was about six to one. You know, it's so big that you can't miss it. It just knocks you off the table when that signal comes through. So that's what made the transformer read-only store so useful. But the transformer read-only store was not as fast as the BCROSS, and you couldn't change the stored data by just slipping a unit-record card in and out, as you could with the CCROS.

But the Endicott laboratory was sold on the CCROS idea you could quickly change the microcode by just punching up a new card on an ordinary IBM keypunch and using it to replace an existing one in the memory. Well, it turned out it was much harder to make the memory work reliably. You had to seal all the CCROS units, make Faraday cages out of them, so that the external noise didn't get in there. So eventually, they paid a fairly heavy price to use that CCROS, which I had previously warned management about, but they didn’t pay much heed. My military service radar experience gave me an instinctive feel for signal-to-noise ratio issues, which engineers who only worked on digital equipment did not have. But about the time I left IBM in 1967, Charlie Branscomb, a senior engineering vice-president with responsibility for the 360 line, said that to me, you know, you were right.

Strickland: Anyway, your folks at Hursley now were on the path to developing the 360..

Trapnell: Oh, yeah. We were on the path now. And so that's how it got started. And we developed the model 40 and released it, and this was the first time a product had been released into Poughkeepsie manufacturing from a foreign location. We did it using data over telephones lines and a lot of airline travel. And we got it built in both Poughkeepsie and in Essonne in France. I think it was Essonne, but maybe it was Montpelier. Both were in France, and I've forgotten which it was.

But we released it into production. And it was a pretty successful machine.

Strickland: OK. Let's jump back just a little bit. We get to my favorite date in history, April 7, 1964. And 360 is announced. And they announce, what, six different models. And help my memory here. But my memory is they announced that we would have software, not even sure we said operating system, but we would have software that would support scores of different tasks going on. And there would be a control program that would manage all this, and so on.

And my memory is we only announced one. That is 360 would have this software, period. Was that the plan or is my memory bad?

Trapnell: Well, so you were a spectator from one point of view; I was a spectator from another. At Hursley we were not directly involved at that point in the software for the 360. We had some work going on peripherally, including some work that eventually led up to the PL/1 compiler, if anybody can remember that. After I left, the work was done there.

But we had some work peripherally going on, feasibility studies. And yes, I think the original idea was one operating system, top to bottom. Well, that idea went on the rocks of not enough memory at the bottom to support the things that you had to do at the top. It just wasn't possible to take one system and have it actually cover the whole range of memories. And that problem, in my opinion, cost the IBM company an incredible amount of money.

Strickland: An incredible amount of money.

Trapnell: So let me tell you from my point of view how that struggle went. At some point, the folks in Endicott, with the small machine [Model 30], decided they had to start on their own software system. They started work on it in the closet, because there was supposed to be only one software system for the whole 360 series. But they had learned from their experience with the 1401 series that it was going to be really hard to get big software run on a small machine. They had a lot of experience with what those machines had to do and what their customers expected. And they just looked at this idea of a single operating system being designed down there in Poughkeepsie, of all places (little love was lost between Endicott and Poughkeepsie anyway), and said, you know, this is probably not going to work.

So they started, I think, on their own. Jim Frame headed the programming there under the overall guidance of John Haanstra, the division VP, who had long regarded the 1401 and its derivatives as better for his class of customers than the upcoming 360 series.

So they started on their own developing some ideas about how the operating system for the 360 would work for a small machine. That eventually became DOS. In late 1964, I moved back from Hursley to White Plains as deputy director of the IBM WorldTrade Laboratories. My boss by then, Byron Havens,

had moved his office over to Nice – he knew what he was doing. He operated from Nice in France, and I went to White Plains.

Strickland: But you're still in World Trade.

Trapnell: Yes, I'm still in World Trade, but I'm located in White Plains running the White Plains office of IBM World Trade laboratories. It was a very small office. I mean, there were six or eight people, and I was representing World Trade labs on this side of the ocean.

Next door to me one day, in the conference room in the building I was in, there was this enormous meeting going on. I didn't know anything about it; I didn't know what it was. But I could hear the temperature rising as people began to shout at each other and almost throw things.

By this time, IBM had already decided that the single OS could not be made effective on machines with 8K bytes of memory or less. So Endicott had created DOS to run on the 8K and smaller machines. Now the argument in the next door conference room was heating up because the Poughkeepsie folks were revealing to the Endicott folks that they couldn't support the 16k memory size either and that Endicott was going to have some other way to build a 16k OS too. Well, I remember stepping to the door of that conference room and listening to that discussion; it was pretty bloody. And this was before I had any idea I was going to be involved in OS 360.

At any rate, out of that came the extension of DOS up to 16k.

About that time, the Systems Development Division was formed under John Haanstra. And John asked me to go to Poughkeepsie and run OS 360. Fred Brooks was leaving to go to University of North Carolina, and they wanted me to take it over.

Well, I had done some programming. I had been involved a little bit in the management of programming. But now I was going to get thrown right in the middle of this huge undertaking. And so I said yes. And off I went to Poughkeepsie.

Well, I got there, only to discover that the major struggle, was getting the system to run in 32k – shades of the of the argument I had listened to in White Plains. Anyway, we faced the question of how to get the OS to be effective in 32K bytes and also in bigger machines that had 128, 256, and even 512K. The sizes of these larger memories were just unheard of in those days; even big computers had only 32K X 36 bit words, but the system clearly had to perform at the 256k level. And to try to get that same system to run in 32k was an incredible struggle.

I claim that fighting the problem of getting OS 360 to run well in a 32K memory cost us an extra year in the schedule. And we couldn't get anybody to listen to us about giving up on the 32K memory. The only people who gave up on it were the customers, and they did so quickly and happily once the OS was released to them.

The struggle seemed unending at the time. For example, I remember sitting in my office at Poughkeepsie quite late in one evening, 8 o'clock maybe, having a conversation with Gene Lindstrom in San Jose. Gene and his team were building the OS assemblers in San Jose. The assembler that was supposed to run on 32K memory computer could actually only use something less than half of that. I personally controlled the major allocation of memory. As I recall, I allotted the control program 15K bytes and I allowed the “application” programs, like the assembler 14 ½K, and I kept I had 1½K of fudge in my pocket.

So when I allocated the sizes out, I gave the assembler 14 ½K, and Gene was pleading with me to give him more memory.

Well, in order to fit a big program, like an assembler, in 14 ½K, you have to bring it in stages, called phases. You have to load a phase, execute it, and then you bring another one. And you have to leave the results of executed phase, the inter-phase logic, behind for the next one to work on, which further reduces the memory available.

Well Gene said to me, he now could get 95 loads of the assembler into 14 ½K, but one phase was 15 1/2k. He wanted to know if he had to split it.

Anyway, we agonized for about an hour to an hour and a half. Do we have to do it? And finally, I said, yeah, you've got to split it.

So they split it. And he called me back the next day. And he said, well, now it's divided into two phases, one of which was 14 ½K and the other one was 15K. Do he have to split that one too?

I've forgotten how exactly we solved that. The point was that all of those applications, all of the compilers and utilities were up against the same issue. This was a very sophisticated assembler, so this case was probably as bad as any. But it shows you what we were up against.

So what happened with the memory problem was, as we shipped the stuff into the field, the customers quickly figured out that 32k just didn't give you anything. And the model 40 customers, for whom this lower end operating system was really aimed, quickly added either another 32k to make 64K or moved all the way up to 128k, just like that. And they didn't bat an eyelash. And then they got the advantage of all

the other stuff. So like I said earlier, it was only IBM management that couldn't figure out that 32k wasn't going to be very effective.

Strickland: It really was both a problem and an opportunity because we benefited from the revenue of all that extra memory. And it wasn't that bad. Customers understood the problem. But if I can, let's go back to you leave Hursley, and John Fairclough takes over your job.

Trapnell: Yes, John took over my job.

Strickland: And you come to Poughkeepsie in the middle of 1960.

Trapnell: I came back in the fall of '64 to the World Trade headquarters job in White Plains and went to OS 360 in Poughkeepsie in early '65.

Strickland: '65. And according to Emerson Pugh Bob Evans asked for you by name.

Trapnell: I never heard that before I read it in Emerson Pugh's book.

Strickland: OK. But be that as it may, you got the project. And how many programmers are on that? Because we're going to be talking about Fred Brooks now. So how many programmers were on the project at this time?

Trapnell: Well, certainly hundreds. I don't know why I think it might have approached 1,000 across all the locations and components for which I had overall responsibility. In Poughkeepsie, I imagine it was 400 to 500. It was a really big program. We were pretty sure it was the largest commercial software undertaking ever attempted up to then. We had, as I recall, some 30 odd different components of the operating system, which included all the compilers and the utility programs and the variations of the control program. We had the PCP principal control program, which was the basic 32K one. And then MFT, which was the multi-programming operating system that was based on fixed tasks. And then later we eventually got MVT [multi-programming with a variable number of tasks] working.

Strickland: When you were there the decision had already been made to separate out and make PCP available first?

Trapnell: Yes, that was made before I got there. I believe Carl Reynolds, Director of Programming for the Data Systems Division and my boss, made the decision to get PCP out first. And that's what we focused on, so that it could support all those model 40s.

Strickland: And I have to add that a lot of people kept their old equipment longer and paid rental on it because they weren't able to do the work they wanted with PCP. But at least the 360s were there, and they did get started. PCP did what it had to do.

Trapnell: That is true. I remember going to a meeting in White Plains where they were ranting and raving about the delays in the OS schedule. You know, what could you say? Hey, here's what's happening. But at the time, John Opel was on the Group staff; he was a man for whom I had the greatest admiration and one of the smartest guys I ever met. He later became President of IBM. So after the meeting, John sat down with me and I think Carl Reynolds – just the three of us. And he says, you know, we can cry all the way to the bank on this delay. But it means our customers are paying us more rental on all those old machines, which was higher than the rental they will pay on their 360s.

Strickland: So it's early in '65. And we're supposed to be delivering MVT at the end in '65.

Trapnell: I think the big control program was called VMS [Variable Memory system] at that point. There was no MVT yet, and I'll talk about how we got there. They called it VMS before I arrived.

Strickland: But the decision had been made to split off DOS, first of all. Not to split off, but to have that solve the problem for the low end, and to create PCP as an interim step. And so those are going on. And you've got about 500 people. And this is the time, I think, that led Fred Brooks to write his book, *The Mythical Man-Month*, to promote the idea that you can't simply add more programmers to a big software project and thereby shorten the schedule..

Trapnell: Right.

Strickland: Was that a slight overstatement?

Trapnell: No, that's pretty close.

Strickland: Is that your experience?

Trapnell: I agree with him. I didn't understand that at the time. I don't think he did either, by the way, at the time. I think it's on reflection this is where you get to. So it was very tempting to add more people at any stage and imagine that you can get more output sooner. But unless you do it very carefully, adding more people can destroy the integrity of the organization and the relationships in it, and thereby make the organization dysfunctional. And until it regroups and puts itself back together, you've lost a lot of time. So

that's really what he's driving at, and it's true. Unless you discover an obvious gap, you ought to start out with the staff you plan to go with and stick with it.

Strickland: Any other observations from just that year of 1965 about leading a huge programming project?

Trapnell: I don't know whether I have any general observations. I think at the time we had so little understanding of software process in the software business. Software was thought to be a craft, an art, not an engineering practice.

Watts Humphrey, who was my boss during the last year of my tour on OS 360, later wrote a couple of books about process, promoting the idea that developing software is an engineering practice, and you ought to follow engineering principles. But this came along much later and certainly wasn't what we practiced.

Tom Simpson, a very smart IBM field programming guy, wrote a lot of interesting software for the 360 and the 7000 series before that. He made an interesting observation. When he visited Poughkeepsie during the height of our struggle, he found that there were excellent external specifications for the various components of the operating system, compilers, assemblers, utilities, control program facilities, etc.; and clearly these external specs had been written by very capable programmers.

But then, the capable programmers disappeared, and kids, just out of college, were doing the coding. We were asking kids to write the code, and he thought it was no wonder we got into trouble. He was a big advocate of using experienced people to write all the code too.

So that's one thing. Another observation is consistent with what Brooks wrote. It's easy for management to keep pushing hard to get things to happen faster. And it can because you make a lot of mistakes. For example, when rushed, it's easy to make a mistake in design, and a mistake in design that is discovered in testing is really hard to fix because you've got to go back and fix the design and the code, and then retest everything. I guess everybody in the software business does it, even today.

In general, not nearly enough time was spent on the front end of the project deciding what's the right thing to do and what is necessary. That's a very hard thing to do anyway. But it means you've got the right specifications that not only programmers can understand, but that people who are going to be using and selling and dealing with the customers can understand too. It allows them to say, hey, this isn't going to work, you know.

That whole combination of things is darn hard to do. But that's what good external specifications do and good engineering practice would say. Once you get that right you can code, and coding is pretty easy stuff, frankly, if you know exactly what you're doing and why. It took me a long time to learn that external and internal specs are vital, because once you've got those nailed down you can go home and code in your bathtub and it'll come out just fine.

Strickland: TSS, or time sharing, was another big problem. Were you involved in that?

Trapnell: No, I was not. TSS was done on the side to support, as I recall, a couple of different, mostly university locations. MIT and Michigan figure big in my memory about why we did TSS. And in a way, we thought it took resources away from OS that we wished we'd had. We eventually did add a time sharing facility to OS, called TSO.

Strickland: Time sharing option, part of VMS?

Trapnell: VMS, yeah, I guess it was. I know that was on the drawing board when I left, but I don't think we'd done anything with it. An interesting story, if you want to talk about it, is the VMS/MVT story.

Strickland: OK.

Trapnell: VMS envisaged a completely flexible use of memory. Remember that PCP only had one partition. MFT had multiple memory partitions, and you could run different programs in different partitions.

Strickland: And you genned [generated] them and they were fixed.

Trapnell: That's right. You genned them, and they were fixed. Well, MVT allowed you to have partitions that were variable. But VMS was planned to provide a fluid memory allocation, a la 370, running on the 360 architecture, which had no dynamic relocation hardware like the 370.

Now, as we approached completing PCP, we began to get serious about VMS, which was being developed by a small group under Larry Cohn, a brilliant software engineer. Scott Locken was the head of the control program, and Cohn worked for Locken.

Cohn had known that for VMS to work required the control program to flexibly load and relocate components anywhere in memory. And in turn, this required that the utilities, assemblers, and compilers as well as their generated code, all had to follow some fairly complex rules of address handling and register allocation.

Well, in the rush to get the OS up and running, these rules were ignored – set aside for a later time may better express it. But with PCP done, the pressure was just as strong to get VMS and the large-system software operational. And as a consequence, there was no time to go back and rework components to make them VMS compatible. They weren’t built for it. And so it became clear that VMS wasn't going to be feasible. And in fact, it wasn't feasible until you had the full relocation hardware that came along with the 370 – wherein the hardware does it for you. So a big contribution of 370 was to solve that problem.

Anyway, Larry Cohn was very unhappy. We had a big come-to-Jesus meeting about what to do about VMS. And I can remember sitting in my office in Poughkeepsie saying, if we can't do VMS, can we do something where we have task spaces can be moved and resized so that it's flexible, not fixed like MFT, and get most of what we want? Well, we argued about that and debated whether it could be done, and I think that was the first discussion I ever had about an MVT.

Eventually MVT replaced VMS in our plan, and certainly for the big systems, it was a very important. But it was a compromise. We just couldn't do what we wanted or had originally been planned with the time we had. So that's how MVT got on the map.

Strickland: So ultimately the idea of having to abandon the VMS and go to MVT started in your office?

Trapnell: As far as I know, yes.

Strickland: And did it end there? You made the decision?

Trapnell: We carried that decision up the line. They could have fired me and tried to carry on with VMS, but they didn't. Yes, I think we fought for it and said, that's what we're going to do, and you're going to have to go back and tell the customers that it's going to be this way instead of that way.

And we had to do quite a lot of that because six months or so after I arrived it became clear that we simply could not deliver the planned plethora of product on the schedules promised. It wasn't going to work. And so we had to sit down and try to take apart everything that had to be done and look at all the relationships between the components and figure out what we were going to do first and how long it was going to take.

We learned an important lesson about planning complex programs that comprise multiple projects some of which are started early and others that are planned to start when people become available from the earlier ones. That is, later projects can only start when earlier projects complete. We had long since learned to apply time contingency to the planned execution time of each project in order to have

confidence in its schedule. This contingency was based on the complexity of the project, and typical contingencies might be 20% to 40% of expected project time.

However, to accurately estimate the completion date for a project that could not start until people and other resources became available from other projects, so we also had to apply contingency to its start time as well, to account for unanticipated delays encountered in those earlier projects. Upper management had a hard time accepting, much less agreeing to, the size of the planning contingencies that resulted, but they were essential to setting accurate schedules for such a big, complex program.

My small staff included a wonderful woman, Pat Goggins. She was a retired army officer, a WAC, and was an old hand programmer and a really good planner. Larry Foster, who's mentioned in the Emerson Pugh book, was also on my staff there along with Bill Crowley and Ted Climas , a well-known name in IBM.

Ted wasn't much of a planner, but he was doing a lot of the interfacing to the outside world. He's the only guy I ever knew who, when you ask him a question and he's unsure of the answer, starts immediately launching off into what, if you're not careful, you think is the answer. And then you realize that you're getting 30 seconds of gobbledygook, while he's thinking up the answer, and then all of a sudden, the answer starts to arrive. And he didn't even give you a clue when the answer arrived. He was a wonderfully talented guy, and unfortunately he passed away some years ago.

Trapnell: Anyway, we started to lay out realistic schedules covering over 12 different major software releases. And boy, the times stretched out. The time of arrival for MVT was in release 12, and release 12 was 2 1/2 years later than the original VMS release date. We laid all this out and started to take this up the management line.

And of course, there was all kinds of flack the Group Staff. They couldn't believe it. And eventually, we had a meeting in a conference room in Poughkeepsie to review these schedules for top management. And lo and behold, Vin Learson, the Senior Vice President of IBM responsible for the product divisions, showed up for that meeting. I had known Learson slightly, and he could be frightening, but this was the first time I came face to face with him in a confrontational situation.

And so I laid out our plan and explained everything and walked through all the details – 30 or 40 components all scheduled out over a three or four year period. And yes, everything was later than what we had committed to customers.

After I finished, Huck Finn, one of the group staff guys, asked pointedly, why does that Fortran take so long? I had just explained why. So I looked him in the eye and said, because I say that's how long it

takes. There was silence; nobody said a word. When we walked out of there, that schedule became official. And who knows how many IBM sales people had to explain it to their customers.

As far as I know, we never missed schedules on those releases after that. That was probably in 1966, and I left IBM before all those releases were done, but I don't think we missed any more. Once we'd laid out schedules so they could get done, we got it done.

Strickland: Now, often IBM executives get a chance to explain not just to other IBM executives, but to our customers what's going on. Did you get in any of those?

Trapnell: Oh, yes. Yeah, I did a number of those. The most memorable ones were at the Share meetings. And the most memorable one of those was in Toronto, I think.

Strickland: Share was an organization of big technical computing customers.

Trapnell: Yes, it was the 7000, 7090, 7094, 360 model 65s, et cetera. And the big issue at the time was COBOL, Fortran, and PL/I. And of course, Share had participated in the design of PL/I, which was an idea to try to create a single language to use for both technical and commercial programming. We'd created a hardware architecture that bridged the scientific and commercial realms that everybody could accept. And now they wanted to try to find a language that would do the same thing.

Well, no one will ever know whether that was possible – I certainly don't. But Share was involved, IBM was involved, and a lot of smart people were involved. Hursley lab was right in the middle of it. (I had left there long before.) They built the first PL/1 compiler.

In the process of coming up with this language, it got really elaborate – as Fred Brooks would say, with gargoyles and all. And I think it really suffered from not having the integrity of design that comes from somebody or a very small group of people who are really controlling the design with a clear vision of what they're trying to do. That's my impression anyway, and I was close enough to probably have a pretty good impression. But I don't think there was anything like that integrity. People added stuff on because it seemed like a good idea. I don't know what's happened to PL/1 or whether anybody uses it anymore at all.

Strickland: Actually, as a quick aside, I ran into it because I worked at Santa Theresa labs. And Santa Theresa lab had responsibility for PL/1 at the time I was there, and then ended up sending it back to Hursley. But yes, it certainly never did replace Fortran or COBOL.

Trapnell: No, and I don't know whether it’s still in use. Fortran and COBOL certainly are.

Anyway, the big issue at the Share meeting in Toronto was, what are you going to do about COBOL? Is IBM going to continue to support it or is IBM going to drop it?

I was working off talking points that were provided to me by policymakers. And I think I got up and suggested we were probably going to drop COBOL. And people hit the ceiling. So I got on the phone back to headquarters, and I said, you know, this isn't going to fly. So I had to go back out and correct myself and said, we're not going to drop COBOL, but we're going to keep going with PL/1. That was quite painful.

We also had a lot of trouble adequately testing software. I remember going to one of the Share meetings and having somebody say to me, I just installed PCP, and I did XYZ on the keyboard, and the machine crashed. Didn't you guys test anything? I can't tell you how many hours and hours of testing we in development did, how much stuff we tested; and IBM Product Test was working at it too. So this guy just says, the first thing he tried was – some weak point he probably knew about – and boom! The machine crashed.

So we had lots of trouble with testing. And a very big problem, it's true with any big system, is the fact that, if you aren't careful, is easy to have the code get worse rather than better when you add changes. That is, a change to correct a problem introduces a different problem or problems. It's called regression. Code regression caused its share of problems in OS, but we also had very strict control. Scotty Locken, who worked for me as the head of the operating system, was conscious of this. He had very strict rules on what changes could be added and how they should be added, et cetera. And so we suffered from regression, but not as badly as we might have.

As overall manager of OS 360, I went back to Hursley on a visit and listened to their discussion about the development of the PL/1 compiler. They were having incredible regression problems. Having learned from Locken, I said, tell me about your regression problems. And they took me out and showed me a 6’ X 6’ board full of little white and yellow cards, each representing an outstanding problem. The white ones were for problems discovered by new tests; the yellow cards were for problems that had previously passed testing but now did not. And I said, wow! There must have been a couple of hundred cards, and most of them were yellow.

So I asked, how do you control the code? They had big steel trays of cards representing the source code that were kept in a secure place, a special room. You have to go in and make the changes in there.

I asked, when you make the changes, does anybody pay attention to what you're doing? Oh, yes, yes. We look those over very carefully.

I said, do you lock the room at night? They said, no.

I said, how do you know people aren't going in there in the middle of the night and making changes that you don't know about. Well, they said, we don't know for sure.

I said, look, you've got to fix this. You either have got to have somebody on duty all night long or you've got to lock that room up so that nobody can get to it in the middle of the night because your programmers are going to go in there and try to fix things.

And so they did it – they locked the room at night. And the yellow cards just dropped to practically zero. It was amazing.

Strickland: But I only changed one card.

Trapnell: Yeah, I only changed one card. And it dragged the whole system down. It's still a problem. You've got to be real careful about regression. It's a nasty problem in software development.

Strickland: Well, you reset the plan. And we ended up meeting the schedules. And IBM certainly suffered some reputation, but really not financially, except we put a whole lot of money into building the operating system. And now, we're at what? About the end of '66 maybe?

Trapnell: Yes.

Strickland: And you're going to leave IBM pretty soon.

Trapnell: Yeah, I left in June of '67. In retrospect, I'm sure I was burned out. It had been a really tough go, and I didn't recognize that. You've got to be more mature than I was then, for sure, to recognize that it was me having a problem.

And I looked at the jobs I might move to. And I thought, I don't see any jobs up there that appeal to me much. I thought the Systems Managers, which were the next logical step up from where I was, had a terrible job. They had to fight the political battles all the time. And I just didn't like that.

What I didn't understand was myself and didn’t realize that, I don't like politics. I'm a pretty good engineer. And if I had been smart, I think I would've said, look, I've done enough of this job. I would like to get back into some good, solid technical design work, especially architectural systems design, which was really

what I was good at. And if I'd done that, things might have taken a different course. But I wasn't mature enough, I think, to see that.

So I decided to leave IBM. I had made good friends in England when I was there and had pretty good connection with Tom Hudson, who had been Director of IBM UK, and we decided to start a consulting company. So I left IBM and went to England in 1967 and we formed a little consulting company, called TC Hudson Associates. I did most of my work on operating systems, software, and hardware with computer companies, telephone companies, and the UK government. I was there for five years.

Strickland: So is that like '67 to '72 roughly?

Trapnell: Yeah, '72. And in '72 I decided I ought to bring my family back to California, where I started and my then wife was from. So I took a job with a little company called Wavetek down in San Diego. The chairman there was Joel Naïve, a technically brilliant and demanding entrepreneur, and John Thornton, the quintessential businessman, was President; I went to work for John. Wavetek was quite a well known instrument manufacture, still important in small instruments – frequency generators, signal generators, and stuff like that.

But Joel had decided to go into the computer audio response business, designing and building units like the IBM 7770. So Wavetek got hold of a design done at the University of Utah, I think, in Logan, for a 7770 copycat that was compatible with a 7770, and they built and sold several of those. They also wanted to develop a much more general purpose kind of audio response unit.

So I came in to run this little division composed of 10 or 15 people, a couple of engineers, and me. We decided to base our design on the PDP 11, which had just come out, a really nifty machine. So it was, in a way, a front end processor, like the 7750. It had its own computer, in this case a PDP 11, plus the logic to handle all the audio stuff, either standing alone or on behalf of an IBM 360 or 370 to which it was attached. I designed the control program for it, which highly specialized to the audio-response application, very application centered in doing audio response. I designed it, we built it, and it worked.

Our old audio-response unit worked connected to a 360 channel, which we understood very well. But we wanted the new system to work attached to the 370 channel, which differed from 360 in some interesting ways that we did not yet comprehend.

One of Wavetek’s most important customers was Rohr Aircraft, which built components for aircraft, military stuff. We connected to the Rohr 370 computers for testing because we couldn't afford our own 370s. And I remember one Sunday morning, I think it was, that we hit a real bug in our system on the 370 channel. It simply wouldn't work. The channel was doing something we didn’t understand, something we could not get around, and I wondered what in the world do we do now.

In desperation, I picked up a pay phone outside the Rohr plant and called Watts Humphrey at home, an old friend and my last boss at IBM. Now he was head of the Endicott lab, and I called him and said, Watts, I need some help. I'm working for this little company. We're trying to attach to a 370 channel, and we are having trouble because we think we understand the 360 architecture, but there's something else going on here that we need to know about. I fully expected him to say, well, that's nice, and then to hang up on me.

Strickland: Yeah, good luck.

Trapnell: But he didn't. He said, OK. He took my phone number and said, I'll call you back. And so he did, in about half an hour. This is Sunday morning. And he called me back and said, I'm going to put you in touch with so and so, to see if he can't help you. And he gave me the name and home phone number of an IBM development engineer.

So I got in touch with him and explained to him what we were doing and what we were seeing . He says, oh yeah. If you want to do this, then you've got to do x and y and z, instead of the u,v,w you were doing.

I made notes like crazy, and we were able, in a matter of 15 minutes, to change our software to do x and y and z that he specified. And it all worked. I really appreciated the help from Watts, the engineer, and IBM, at a time when IBM’s reputation was to be extremely secretive about all their interfaces.

Strickland: Well, yeah. That's terrific for you and very gentlemanly of Watts.

Trapnell: Gentlemanly of IBM to do that because they might very well have just said, you know, it's tough luck if you can't figure it out.

Strickland: And so you got your audio response unit working.

Trapnell: Yep, we did. But we never sold many of those units. It was 20 years before its time. In 1973 or 1974, only a few banks were ready to move ahead with on-line audio-response banking. 20 years later, it was all the rage.

We didn't have a big enough base, and we had some competition. An outfit by the name of Periphonics also built audio response systems. They were a bit ahead of us in time and in capability.

And IBM, of course, still had the 7770. As far as I know, I don't think they exploited that very much at the time. I don't think they thought it was a very big business. I think they might have sold a couple of hundred 7770s. And that was nothing for IBM.

So we were 20 years ahead of our time, and we were a little company. And essentially, we ran out of money and gave it up. So I spent nearly a year doing some consulting based in San Diego.

And then I got an offer to join TRW. TRW, bless its cotton socks. You think of TRW with rockets and spacecraft and all that sort of thing. Well, they wanted to get in the retail point-of-sale (POS) systems business. And they bought a company that was building POS systems for the May Department Stores.

The May Company owned this outfit, which had a development lab in San Diego. And May Company decided to get out of the POS business because it was too complicated for their management to run. So they sold it to TRW.

Garrett Fitzgibbons, who I had known slightly in IBM, had come in as TRW’s Retail Systems’ director of development and manufacturing. He hired me to run development in San Diego. The May Company, a department store company, already had an older POS system, but they wanted a more effective one. So we designed and built a new system and struggled to get it installed.

Well I believe, the retail business was modestly successful for TRW; but they eventually sold it to Fujitsu, who I think is possibly still in this retail POS business. I don't know.

Strickland: IBM ended up selling its retail systems too to a Japanese company. And I think it was Toshiba

Trapnell: Anyway in 1978, I got an offer to come to Amdahl as vice president of software, and I did in 1978.

Strickland: OK, well Gene Amdahl was a famous IBM guy, architect of 360 and so on. And in 1970, he started his own company. But for a number of reasons, Fujitsu ended up getting heavily involved in Amdahl, so at the time you joined, Fujitsu was a big shareholder.

Trapnell: Already a 49% owner. The way Gene tells the story, he couldn't get anybody else interested in supporting this idea of compatible mainframes. And eventually, he went to Fujitsu to see if they would do it, and they agreed to, because they were interested in that market themselves.

Strickland: But also by this time, the machines at least are successful. I don't know about the finances, but he's making a big impact.

Trapnell: The Amdahl 470 was a very successful machine in its day.

Strickland: And his machine was running, at that time, MVS?

Trapnell: Yes, MVS, that's right.

Strickland: OK. So you come into that environment. And what was your job?

Trapnell: **A** Senior Vice President, Harold Shattuck, a stern but fair early employee of Amdahl whom I had met in IBM, hired me as Vice President of Software. The question was what should we do in software? We had several software products that emulated hardware features of the 370. A software product called MVS/SE had been developed before I got there. We also developed a VM assist, called VM/PE. These products filled in the compatibility gaps between the Amdahl and IBM hardware. These projects were headed by Nomi Williams, a cheerful, ebullient software veteran who had done control program development for CDC – a superb manager who could forge cooperative teams out of talented but recalcitrant programmers. (Full disclosure: She later became my wife.)

Down in the organization where nobody could see it, under a brilliant engineering manager named John Hiles, we were also developing a UNIX for the 370. And that was quite interesting to the telephone company, AT&T, because they were UNIX folks. If you remember, they invented it. And they were big on it. And IBM had several UNIX installations on 370s because they also had a UNIX – I’ve forgotten what they called it. ATX? Anyway, we called ours UTS.

Strickland: The UNIX that they ended up putting out on a smaller, like a RISC system, was called AIX. But I don't recall UNIX on a large system.

Trapnell: Yeah, they had UNIX on a large system too. But it turned out that AT&T liked Amdahl UTS better, I think because of the features and the flexibility and perhaps because it wasn’t from IBM. And we sold a number of Amdahl systems that ran UTS to AT&T, perhaps 30 or 40, and at the rate of maybe 10 a year, maybe even a little more from time to time.

And from that point of view, it wasn't small for for Amdahl, but it was a small project, maybe 15 people. And for 15 people to sell 10 or 15 big systems per year it was worth it. So that was another thing that we did in software.

Strickland: Was Fujitsu a help or a problem or just a non-factor for you or for UTS and you?

Trapnell: I don't think they were very much of a factor. They were sort of interested in what we were doing, but they were much more interested in the MVS side of things.

Strickland: Was Gene Amdahl a factor for you personally?

Trapnell: Gene Amdahl was only a brief factor for me personally at Amdahl, because he left a week after I joined. He had some personal reasons for doing so. Though he was not a factor for me, his leaving had a big effect on the company. Amdahl had recently brought in Jack Lewis as president. Jack had been a financial guy at IBM, and he was a great counter-balance to Gene, who was a risk-taking technical power house. When Gene left, Jack became the CEO and the center of company leadership, and the character of the Amdahl Corporation reflected that.

Gene was also a factor in something we attempted to do in software, which was to create an operating system that was an alternative to MVS. In the last days before he left, he was enthusiastic about embarking on this venture, and that enthusiasm kicked off the project and sustained it for some time after he left. It was called Aspen. We put together a small group to design and implement it. We thought a lot more interesting and flexible than MVS, and it had an MVS emulation capability – you could run MVS programs under it. But it really foundered because IBM violently objected to the MVS facility in Aspen, didn't believe it wasn't stolen, which it wasn't. It was designed and written by one of our people who had never even worked for IBM.

But IBM had had trouble with code that was stolen. I don't have any firsthand evidence, but we heard rumors that Fujitsu got in a lot of difficulty with IBM.

Strickland: Yeah, there was something there. I don't remember the details.

Trapnell: I don't either, but I've even heard about examples of things that were found in the Fujitsu code that were giveaways.

Strickland: I think there was even a settlement of some kind.

Trapnell: Yes, there might have been. Anyway, IBM threatened Amdahl with legal proceedings, so we dropped that project. I'm not so sure that developing Aspen was as wise an idea as we originally thought. But Gene Amdahl was very keen about it. And maybe if he'd stayed he would have guided it through, but

I don't know. We had a very difficult time with it. For Amdahl, it was expensive; whereas UTS paid for itself.

In 1987, Amdahl got a new president, Joe Zemke, while Jack Lewis remained as CEO. And Zemke decided to expand UTS and tried to make a big operating system out of it, expand it so it would be used by lots of other people. I disagreed with this, because UNIX was too specialized to have broad appeal. It was great for AT&T and possibly some universities, but not for the general mainframe customer. So I ended up not running software anymore, and I left Amdahl a couple years later. The expanded UTS ultimately turned out unsuccessful.

Strickland: So they wanted to make UTS more of a large system.

Trapnell: Well, it was a large system all right. It just wasn't universal. They were going to put a lot of facilities into it that MVS had. I'm not entirely clear about the design. I was no longer in the management of the program; I could only watch from the periphery. So I can't be very specific. But I know they had several hundred programmers working on enhancing UTS to make it a more general operating system.

And it really didn't work because the market just wasn't there. Nobody wanted to buy a full blown Unix system, unless you were hooked on UNIX in the first place, which the phone company was. It made sense as a limited project for AT&T, but not as a general purpose operating system.

Strickland: So what were you doing at the time that that project, somebody else was working that?

Trapnell: I moved over to a staff position under Bill O'Connell, an Amdahl Vice-president who I had known at IBM, and worked on a number of things, the most interesting one of which didn't see the light of day either. It was a programming system called Huron that was invented by a a really bright engineer at Amdahl, called Helge Knudson.

It was a fourth generation programming and operating system, which I always thought was rather better than any of the data bases or fourth generation programming systems that I saw at the time. It vastly simplified programming for database operations. It had its own unique database and its own programming language, which offered a total of about 20 operation codes from which you could create a full range of commercial programs. It was that simple – hard to believe. And when I explained to Zemke the simplicity of Huron, he found it so hard to believe that he refused to listen to me.

Strickland: Let me get out my green card here.

Trapnell: Right. But Huron had about 20 operation codes. And you could do it all. I know that because I did the programming to prove it to myself. I always thought, the ultimate test of an application language and system was whether you could write a bill-of-materials processor with it. Bill-of-materials processors. have to handle deeply recursive operations. And if you can do that with a system like Huron, I think, it will do just about anything. And I wrote a bill of materials processor that satisfied me. Yep, you can do that with Huron.

But Amdahl were never carried it forward. They really had a difficult time promoting anything except its large mainframes. I think that was part of the culture, the executive culture and the sales culture. There was so much money to be made in just selling big boxes; why should they bother with a minor software product. It was just a peripheral side issue. However to make something like a Huron go needed a big push, because a new way of programming applications was a hard sell. Perhaps a company like IBM could have made a go of it, but not Amdahl. And as far as I know, nothing ever came of Huron.

Strickland: Well, let's see. So in 1989, was it, you said you left Amdahl?

Trapnell: I left Amdahl in '89.

Strickland: And you were staff assignment at that time?

Trapnell: I was, and I was a bit discouraged with the whole computer business and decided to retire. I thought I was going to retire and write a book. I got very interested in writing and did a lot of studying about writing and wrote the book – a novel about an American pilot in the Battle of Britain. And I also took time to design the house that we live in now. I had an architect, but I became very in the layout of floor plans and specifying fixtures, fittings, and finishes – a lot of rewarding work.

I also tried to start a consulting business. By that time, I had pretty well figured out that software process was a really important and needed to be understood by a lot of people. And I wasn't the only one that figured that out.

I don't know if you've ever heard of clean-room programming, which a couple of guys at IBM invented and wrote about. It called for very strict disciplines in design and coding, some of which are powerful in developing error-free code, but not all are practical in a normal software shop.

I'd become convinced that to produce good code out of the box required good external and internal specifications, and the specifications need to thoroughly reviewed to winnow out mistakes. Many programmers don’t like to do it, but as Fred Brooks points out, writing down design decisions makes you

think carefully about them. Mistakes in specifications are easy, cheap, and quick to correct; mistakes in design that are carried into code and then discovered later in testing can cause havoc.

Clearly written specifications also allow you to show people what you're thinking about, what to expect from you. You may have to build a mock-up of displays and human interfaces to show what those will look like. But all that stuff really pays off because everybody on the team knows what to expect, and if you get the specs right, coding is easy – straightforward. Coding is difficult when you are doing the design as you code.

But you must not try do everything in the first release one of a product. Keep the first release as simple as possible. Deliver a first release that's going to be useful and allows you to get the product in the hands of customers and in the care of your support people. Then follow it as soon as you can with enhancements that improve it. This gives your customers and your sales people confidence that the product is going someplace. At the beginning of the project, you have to think about releases two and three. We learned that in OS, and the process of writing specifications was very interesting to me. It still is.

Strickland: OK. So author and home designer and consultant from '89 to '95.

Strickland: And then in '95, you joined Tandem.

Trapnell: Yes. I had developed a tool for writing pseudo-code internal specifications using Microsoft Word. It was written in Word macros. So you could sit down in front of Word, bring this up, and write pretty good pseudo-code conveniently, with all the formatting and such taken care of. I went over to Tandem one day to show it to them. They weren’t particularly interested my pseudo-code writer, but it of got me in the door. And Barry Hills, a Director there and a great guy, hired me as a program manager on programs that had nothing to do with it.

I worked on several different programs. The first one was an internal system for Tandem to manage its own software and prepare it for release – a tool to run their software business. Later, just before it was taken over by Compaq, Tandem attempted to get into the minicomputer business. And I was over with that group for a while – nothing significant there – and that evaporated when Compaq took us over, because they were minicomputer specialists.

Strickland: OK. So just to go back a second. Tandem was about a 20-year-old company when you joined them. And they had made all their money and still, I think, were pretty successful in doing nonstop systems for the banking industry, and probably some others.

Trapnell: Yeah, it's banking and hospitals and stock exchanges – anyplace where you can't afford to have the computer go down at all. Tandem systems are expensive. They're not a cheap way to do this, but they give you seamless and virtually instantaneous recovery from hardware failure, which you don’t get from a mainframe. The guys that buy them will tell you, we buy Tandem because we like to go home at night and on the weekends and get some sleep and not expect a call in the middle of the night. They also have an excellent database built to be as completely bullet proof as possible, probably better in this regard than any database system in the world, and to provide automatic recovery for anything that goes wrong.

Well, the Tandem hardware, software, and their database are today the core of the HP, Nonstop division; rather it is the Tandem philosophy carried over, but more modern, updated, faster, and bigger. Literally with a Nonstop system, if something goes wrong, a red light that goes on in the unit that's failed, and the system calls the central customer engineering and reports the lost unit. A service guy can go out and fix it then and there or, he can wait until the next day or Monday morning.

Strickland: Because the backup system or systems keeps going?

Trapnell: Because the whole thing is so completely backed up. And if they're set up for it, a failure here, say in Cupertino, can be backed up instantly on a machine, for example, in Chicago. It's automatic! They really have worked that out probably as well as anybody in the world for commercial systems. I'm sure some military stuff is more sophisticated.

But this was the Nonstop specialty, and they charge for it, yet there are still lots of customers who won't do anything else.

Strickland: OK. So that was '95. But in '97, Compaq acquires Tandem.

Trapnell: By that time, I was working on the minicomputer side of Tandem, which we gave up in favor of Compaq. And I took over a group that was involved in, what they called, solutions – things like big data. Data mining was what we called it; big data is what it seems to be called today. These were data systems for searching big collections of files, largely for banking and finance systems.

We also developed an automatic banking system.

Strickland: ATM?

Trapnell: Yes, a special ATM with a number of interesting features. I stayed as Director of Solutions for a while, and then I wanted to back off on the time I was working. So I decided to try my hand at technical writing.

By that time, I'd written that book we talked about. And I introduced myself to the head of Nonstop technical writing, Dave Cartwright, a friendly and thoughtful ex-pat from England, and said, I'd like to work for you. And I believe I can help you a lot, and read this if you want to see whether I can write or not. So hired me and put me to work on dynamic link libraries.

Nonstop was belatedly getting into dynamic link libraries (DLLs). And I was given the assignment of writing something about them. Darrell High was the chief architect on DLLs, and he had written a very thorough specification. Darrell's a pleasant but slightly dour brilliant guy – an elaborate writer who took great care with all the details right down to the nuts and bolts. It was an exercise in diligence to read his document, but it covered every aspect of the design, so we knew what we had to do.

When I first got onto this project, nobody in Nonstop had any idea what a DLL was, and the design was not yet ready for a manual yet. So I thought, OK, I'm going to write a paper and explain what a DLL is. Well, it was a big hit because in about 15 pages you could really understand what the heck is going on with DLLs. Dave Cartwright even gave me an award for writing this paper.

Then I decided, while they are still designing DLLs I was going to write the manual for them. It’s a good idea anyway to write the operating manual while you're designing, because it is after all the critical external specification, and you're going to find a lot of interesting things as you write it down. And we did.

I know I learned a lot from the designers, and I turned up a number of problems – inconsistencies and such in the design, which they were happy to find. It helped them. And the manual was published as the main customer document about the system. I wrote all the programming examples of things you could do and how to do them. And a lot of my friends have said, I've used your examples and proved it works just like that. That's was good to hear.

So I did some technical writing.

Strickland: OK. So that's now still with Tandem.

Trapnell: Nonstop has now become Compaq. In 2001, shortly after Nonstop became HP, the program manager of the DLL development program left the company; and Carol Minor, the Director to whom the DLL program reported, was looking for a new program manager. Carol was sober, dedicated veteran of Tandem with excellent strategic judgment. So I met with her and said, how about me? I know quite a bit

about DLLs, and I'm ready to go back to work full-time. She hired me. So I took over the DLL program, all the software to make the DLLs work on Nonstop.

Strickland: This is how many people? Ballpark?

Trapnell: Maybe 25 to 30 people involved across all departments. And we had a very successful DLL design. Nonstop, by that time, had really finally got its act together on process. They had learned that you cannot force schedules down engineers' throats. You have to plan carefully. You have to follow a process of specifications and reviews because it's those reviews and the modifications that come out of that that get you to the idea of writing error-free code, which is what ought to happen.

Testing is important. It's vital. But it's not the way to debug software. Software should be debugged by programmer care in writing and peer reviews. And through no effort of mine, Nonstop had got their software process pretty well under control. I think Pauline Nist, development vice president, was the instigator and mover of this, and I was a supportive bystander; I watched it happen. And I admired the fact that these guys actually figured this stuff out – stuff I'd been talking about for a long time. They really adopted it into the culture.

So we had a very successful DLL development program. It took a couple of years. I was program manager there, introduced it. And it went out. It did not have a lot of problems. It just went in very smoothly.

Partly because of the process I just described, but also largely because of Darrell High’s clear and complete specification, we knew exactly what we had to do. And he had thought it all the way through, and that provided integrity for the whole thing. That integrity, the completeness of the specification, and following the process made it just go really easy.

Strickland: OK, well we've got Tandem now being taken over by HP. It's still Nonstop.

Trapnell: Nonstop. It's still the Nonstop division of HP.

Strickland: And you are leading a group of about 25 or 30 folks. What year are we in now?

Trapnell: I'm thinking that it must be about 2005.

Strickland: That's just about the time that Carly Fiorina was ousted.

Trapnell: Yeah, she was ousted while I was there.

Strickland: Did that have any effect on you?

Trapnell: No, it didn't affect us directly. She was trying to do a job that I think she was hired for, which was to make HP more market sensitive. There were a lot of HP people who didn't like her. Her style was not very compatible with what they did. They had been used to an entirely different style. On the other hand, she definitely was trying to do something that we, coming from the outside, thought HP needed. At least I did. It needed to be done. OK, it wasn't easy, and some think she wasn't successful. But an awful lot of the products in the subsequent years – small servers, desktops and laptops – came out of the old Compaq, not out of HP, and those products made an awful lot of the money. A lot of people like to say the Compaq acquisition was a big mistake. It's not clear to me that this was true. She pushed HP in the direction they hired her for, though she was probably not the right kind of person to be in there for the long haul. Maybe they could have done a better job of picking somebody different. I don't know.

Strickland: But you did work for Mark Hurd for a year.

Trapnell: Worked for Mark Hurd. And he was there for a year or two. That's correct. And he was a different kind of a guy, very oriented towards the business. And coming from the National Cash Register, he had a good computer background.

Strickland: And even a software background I think.

Trapnell: Yeah, exactly. Though he got the Nonstop division into some stuff that I don't think they've yet recovered from, which was trying to compete with Teradata. They tried to do some things that Teradata had done with their huge file systems. And Teradata had been at this a long time and was quite successful in their own right. It was a tough market to crack into.

Strickland: Database boxes.

Trapnell: Database boxes, yeah. But I went on to run another program to put better communication processors on the Tandem systems.

Strickland: Oh really? Back to the beginning, eh?

Trapnell: Full circle. Very interesting, because when I was asked to do this, somebody knew that I'd worked on the 7750. And they said, you know what, we want to ask you to come back and do is again. So

I was program manager on the communication processor, and the idea was to have small computers, front end computers, that did the work of managing the communications, and take that off of the main processor.

Strickland: Wintel type or were you going to make your own operating system?

Trapnell: We were doing our own. I don't remember whose processor it was. But it was a separate processor, with its own stored program – very sophisticated. It was not a little machine at all, in terms of memory and software. This little box off the mainframe was quite sophisticated in managing the communications and protocols, and all of the things that communication systems have to do.

So that was the idea. And they eventually shipped it after I retired early in 2007 – this time, for real.

Strickland: You made it stick this time.

Trapnell: Made it stick.

Strickland: Good. And so now, obviously 2007 until now, you are retired. And I saw that you did a theater review.

Trapnell: My wife did.

Strickland: Your wife did.

Trapnell: That's my wife.

Strickland: And your wife's name is Nomi. And you've been married for . . .

Trapnell: Thirty-three years.

Strickland: Thirty-three years. Wonderful. And I want to ask you about the book you wrote about your dad. When did you do that?

Trapnell: When? Well, this started about four years ago. I didn't want to do it, but a friend convinced me that if I didn't write this book, nobody was going to. And it is a pretty interesting untold story about naval aviation.

My oldest daughter, Dana Tibbitts, and I decided to do this together. So we've been working on it for about four years. My father was quite a well known aviator and test pilot in the Navy. Virtually all of his shore duty was in test flying and flight test engineering. He retired as a vice admiral, and the airfield at the Naval Air Station Patuxent River, Maryland is named after him.

So he was quite well known in the Navy, but not so well known outside it. So we decided to write a book to explain his life and what it was he did. And so he got into flight testing in the 1930 and when not on duty at sea, was in flight test —until 1950.

In 1932 to '34, he was in the airplane unit that flew on and off the Macon and the Akron before her, and he was stationed here at Sunnyvale when the Macon was here. From 1934 to 1940, he was scouting and patrolling. And then as World War II was about to break, had broken in Europe already, he went back as head of flight test – senior flight test officer for the Navy – at Anacostia, DC. And he struggled for the next three years to bring into service all of the airplanes that the Navy and Marines used in World War II, in particular, the Corsair and the Hellcat.

Some of them were not only tested but actually redesigned by the flight test unit he commanded. He was in the South Pacific for the last two years of the war and then came back and was head of the Naval Air Test Center at Patuxent River.

Patuxent was set up as a test center during the war. He went back to head it up, and was there for four years. His major contribution there was in getting the Navy into jets. Naval aviation was a whole new game with jet propulsion, a whole bunch of compromises were needed in airplane design, ship design and so on; and he was really in the middle of it all. It was an interesting time.

He left there in 1950 and became skipper of a big aircraft carrier, and then made admiral, and retired early. He had a heart condition that retired him at age 50, a bit young. But he lived happily until he was 73.

Strickland: Did he leave materials for you that you used in your book?

Trapnell: He left nothing, well, almost nothing. I've got his log books. He didn't write. He was not a writer. I have one document, one 60-page document he wrote on how to manage a sea plane on the water – docking mooring, dealing with crosswinds, all this stuff. He was an avid sailor so this was second nature

to him. He probably used this document in training his squadrons. It is one of the few things he wrote that we have; we had to do a lot of digging.

Strickland: I would think that'd be a pretty difficult book to write then.

Trapnell: Well, I don't know how to compare it to other books. I know I spent a lot of time at the National Archives and at the Naval History and Heritage Center, digging through old reports, flight test reports, and some time at the museum and U.S. Naval Test Pilot School at Patuxent.

He was instrumental in getting the Test Pilot School started, which is a wonderful school. They've graduated over 4,000 people. Most of the Navy and Marine astronauts are test pilot school graduates. And it has helped to populate the Navy with people who are really knowledgeable about airplanes and how and why they fly the way they do.

Strickland: Well, I infer that your granddaughter's the one who sort of prompted you to do this, and you did it with her.

Trapnell: No, she just decided to sign on with me. She's, of course, long since grown and has kids of her own that are grown. But she knew him, and she decided she wanted to be part of writing this. So she and I cooperated on it.

Strickland: And that book is out now and available, right?

Trapnell: The book is not yet out and available, but it is to be published by the Naval Institute Press in the spring of 2015.

This has been interesting because a lot of what you've ask me about are things I haven't really thought about for so long. I've been much more involved in aviation, and well, a lot in software stuff, some of the software process things. But going back in history to IBM times has been quite an interesting adventure.

Strickland: Well, as we wind down now, is there anything that we didn't get into, that you'd like to get into?

Trapnell: I don't think of it right now. I think we pretty well hit all the major bases.

Strickland: You know, I give tours here. And when I give a tour, I say, hey, that 360 code, that's still running today. So I'll bet there's a lot of your code, that OS 360 code, is still running today.

Trapnell: There probably is.

Strickland: And that's got to be a heck of an achievement.

Trapnell: Long time. Yeah, when we built it, I remember, the first release of PCP was 440,000 bytes. And that was the biggest piece of commercial code, we think, that had ever been written. 440,000 bytes. I mean, today it's nothing. It's just amazing how things have changed. And yet, we had so much trouble with trying to fit it into small memories, as we've talked about. Yeah, it was a huge piece of code.

Strickland: Well, just a couple of kind of summary questions. What do you think is the biggest contribution you've made?

Trapnell: Isn't that an interesting question? I never thought of myself as making contributions.

I guess the thing that was most innovative was the IBM 7750. As far as I know, and Fred Brooks agrees with me, this was the first product that connected a stored program (front end) computer to a host computer, wherein the front end handled special processes needed to support the overall host computer mission. That made it the world's first front end processor product.

Also I had some communication patents, There were four or five patents, on which I was the inventor, that had to do mostly with computer communications. Those were important contributions, and I received an IBM Invention award.

I think getting the Hursley lab on the IBM worldwide map was probably an important contribution. It took heavy effort and temporarily cost me some health issues, but we got it done. In the early 1960s Hursley became a major IBM development Center and one of the important computer development centers in the world. And that, I think, was a contribution.

It seemed like a small local decision at the time, but in retrospect I think that at Hursley, my splitting the read-only store (ROS) development out from the computer project made a an important contribution, because it allowed a separate technology group to focus on developing the memories needed for the full range of 360 machines. Without that, it is likely that the Spread committee would not have pushed this technology into as many products, and microcode as a standard for computer control might not have

matured so early. And OS 360, yeah, I think getting that out the door was a contribution, and we have already talked about it.

Also, building that little control program for the PDP-11 at Wavetek was pretty challenging and interesting. It was a different sort of software architecture like nothing I had seen before. It used some state-machine techniques that I knew about, but I'd never seen them implemented. It came out very nicely.

I think getting the Nonstop DLL program out was significant. It was incredibly assisted by having the right design and the processes already there. A lot of people were complimentary about how smoothly it went..

Getting our book published will be a contribution. Also, designing our house was worthwhile too – a place and a location we love.

Strickland: Any comments about some of the people you worked for? Like, who do you think are some of the really good executives that you worked for, respected most?

Trapnell: Well, that's an interesting question. John Haanstra perhaps made the biggest impression on me. He was direct and forceful and persuasive – a powerful executive, who later left IBM and became president of GE Computers. Bob Evans, of course, left an impression everywhere he went. He was a very forceful guy, as was well known. I had a lot of arguments with Bob. And I have to say, he let me win at least the ones that were most important to me.

In the course of developing the 360, there was a certain machine we needed in order to build SLT cards. At Hursley, we were developing read-only memories, and we had to build the cards for those. IBM built only 11 of those machines, and I made sure to get ours allocated pretty early.

While in a fit of cost saving, Evans tried to eliminate my machine at one point, and I hit the ceiling saying, you cannot take this away. We are 3,000 miles from any other place that can make SLT cards, so we can't do our job without this equipment. Then we argued a little bit, but he eventually gave in on that.

Later, he also wanted to include some facilities in OS that I was pretty obstinate about not allowing. We were so overwhelmed that I just refused to take on anything extra. It was some stuff that they had developed in Federal Systems Division when Bob running it, and they really wanted to incorporate it into OS. But we were under tremendous stress. Looking back on it, I wonder whether we made the right decision. But I thought it was right, I just refused to pay any attention to it and we just said, no. And that was the final answer. Anyway, Bob and I had those kind of arguments.

Who else? Chuck Branscomb, whom I liked a lot. I thought he was a really good guy. Watts Humphrey, I worked for. He was always very nice, a very polite guy, very structured, a bit more structured than I used to deal with, even though I'm pretty structured too. I don't think we ever crossed swords, not in the way I did with Evans.

John Fairclough worked for me. He was a really good guy, and we got along very well. I got along very well with all those guys in England. They were a really good group. And they stayed right in the middle of IBM’s computer and software development from then on. As far as I know, they're still in there.

I knew Tom Watson from a distance – enough to chat with him easily. I knew Tom’s brother Dick Watson very well. He was not nearly as effective a leader as Tom – not as strong a personality, but a nice man.

Strickland: And he was running World Trade when you were there.

Trapnell: That's right. And he liked to talk to engineers. It suited him to occasionally dig down and find out what's going on with engineers. And I was one he knew, and I could speak English. He definitely was a mentor, in a way. I knew I could appeal to him personally if I wanted, and once I almost did. When I was trying to get the model 40 project established at Hursley, one of the things I needed was the budget for it. Now, World Trade didn't have that budget, all the money to fund developing the 360 product line was over in the domestic Data Systems Division.

So we weren't going to get the budget. We had the responsibility, but no budget. So I went to my boss in White Plains, Byron Havens, who had succeeded Gardiner Tucker, and said, look, we've got to get the budget settled. And he said, we haven't got the money. And I said, this is Hursley’s project. What am I supposed to do? He said, I think you ought to go to Poughkeepsie and ask them for the money. Fat chance, I thought – Data Systems in Poughkeepsie was a completely different organization.

Nevertheless, I went to see Max Paley, the development manager for the Data Systems Division in Poughkeepsie, who worked for Bob Evans, and Max had the DSD product stuff. I had just flown across the Atlantic and was pretty tired, I remember. I called up Max that afternoon and said, Max, I've got to talk to you urgently. He said, OK. Come and see me at 8 o'clock tomorrow morning. So I got to Poughkeepsie at 8 the next morning, went into Max's office, and started the conversation saying, Max, I need $1 million. And he ended it by saying, I haven't got $1 million.

So I went back down to see my boss, Byron, that afternoon. And I said, Max doesn't have the money. And Byron said a bit remorsefully, well, I guess that's it – no money, no project. And I said, look, everybody has agreed that the Hursley lab should do this project, that it's capable of doing it, and somewhere there's money to build this machine. But I can't get it. And I guess I don't accept this. And I want to use the Open Door to Dick Watson and explain the situation to him.

Well, that electrified everybody. You remember about the IBM Open Door policy.

Strickland: Oh, yes.

Trapnell: The Open Door allowed you to go over your own boss’s head to appeal a decision. So I invoked the Open Door. And before I could even exercise it, things started to happen. And we got the money. We got the money transferred to Hursley within a matter of days. And so Dick Watson helped me indirectly. Because I knew we would accept me, he would be happy to see me, and he probably would have solved the problem if it had been necessary – it wasn’t.

Strickland: Well, and he helped you solve the problem with something that you feel was one of your most important contributions. I think that's terrific.

Trapnell: That's true.

Strickland: Well, Fritz, it's just been a great pleasure for me. It really has.

Trapnell: Good. I've enjoyed it.

Strickland: I hope that sometime I get to show you around the museum and kind of point to some of the things you were talking about. And, Fritz, I certainly enjoyed our conversation.

Trapnell: Well, thank you very much. I enjoyed it too. And I thought you were very good at it.

Strickland: When historians read this 30 years from now, they'll learn a lot about running a big programming project or a small one for that matter.

Trapnell: Well, good.

END OF INTERVIEW

