An Interview with

JACK DENNIS

OH 177

Conducted by Judy O'Neill

on

31 October 1989

Cambridge, MA

Charles Babbage Institute
Center for the History of Information Processing
University of Minnesota, Minneapolis

Jack Dennis Interview
31 October 1989

Abstract

Dennis describes his educational background and work in time-sharing computer systems at the Massachusetts Institute of Technology (MIT). The interview focuses on time-sharing. Dennis discusses the TX0 computer at MIT, the work of John McCarthy on time-sharing, and the influence of the Information Processing Techniques Office of the Advanced Research Projects Agency (later the Defense Advanced Research Projects Agency) on the development of time-sharing. Dennis also recalls the competition between various firms, including Digital Equipment Corporation, General Electric, Burroughs, and International Business Machines, to manufacture time-sharing systems. He describes the development of MULTICS at General Electric.

JACK DENNIS INTERVIEW

DATE: 31 October 1989 INTERVIEWER: Judy O'Neill

LOCATION: Cambridge, MA

O'NEILL: As you know our primary interest is with DARPA and in particular in the history of time-sharing. How did you get interested in computing and in the computer field, and what experience you had with computers before your PDP-1 time-sharing system?

DENNIS: PDP-1 time-sharing system, that started when the machine was delivered to MIT. That was about 1961, wasn't it? Do you have the date?

O'NEILL: I do have the date. I think it was a little bit earlier than that.

DENNIS: Maybe 1960.

O'NEILL: I recall '59 or '60.

DENNIS: We were probably planning for it during '59. Now, the PDP-1 was brought into the research laboratory of electronics. They had a room over in building 26. And the room chosen for the PDP-1 was right next door to the TX0 computer, which I had already been responsible for for probably a year or so. The TX0 had been brought to the electrical engineering department from Lincoln Laboratory where it was built. At that time I had been working on some rather major improvements on the TX0 hardware that would support a more extensive software system. So I was involved in improving the TX0, making additions to it, both hardware and software. Now to go back before that to my experience with computing, I received all of my college education at MIT, starting as a freshman in 1949, and I received my doctorate in 1958 in the fall. Immediately after completing the doctorate I joined the faculty, first as instructor, then the next year as assistant professor. So almost immediately after I joined the faculty, I became associated with the TX0 and the activities surrounding the TX0. I worked with Tom Stockham -- the name may mean something to you. He was on the MIT faculty at that time, and we jointly put together one of the earliest symbolic

interactive debuggers that was called FLIT [laughs]. That became a model for the DDT program which was later done for the PDP-1. So Tom Stockham and I did MICRO FLIT. Now going back prior to that time, my education prior to my graduate education was in electrical engineering. The first computer course that I had was from Charles Adams around 1954. As a Masters Degree student I was on the cooperative program in electrical engineering, and my co-op assignments were with the Cambridge Air Force research laboratory. I had some acquaintance with computers there because I saw computer hardware used in connection with various radar projects that were going on at Bedford airport. So my first real contact with computers was with my co-op assignments. Those were special purpose computers, not general purpose. My first contact with general purpose computers was the Whirlwind machine, and the course taught by Charles Adams. When I started graduate school, I started writing programs for Whirlwind, and during graduate school I wrote a very extensive program to solve the transportation problem in machine code. Later I transferred that program to the IBM 709, so I had that experience behind me.

O'NEILL: As I understand it, the Whirlwind had displays and input devices that allowed the operators to have some interaction with the machine. As a programmer did you have any of that?

DENNIS: That interactive work was mostly behind closed doors. I knew of two activities that used the interactive support: One was the work on the air defense system, Cape Cod System, as it was called at that time. I knew that that was going on, but I didn't know much about it. The second was Doug Ross' work on the AED project. Now as an ordinary programmer using Whirlwind, we had something called a scope post-mortem, which made it possible to present the contents of the machine's memory on the display cathode ray tube in position in the control room of the machine and had a camera set up to take pictures of the display. The standard way of debugging programs, if you couldn't figure out what was wrong by any other technique, was to dump memory through this display camera option and then you could get the film later and look at it through a microfilm reader [laughs]. Which was pretty tedious...

O'NEILL: Sounds a little painful.

DENNIS: If you went during the day, operation of the machine was in control of the operators, so the most you

could do was present your paper tapes to the operator, typically Mike Solamita who was the chief operator of Whirlwind computer, it seemed to me, during the time I used the machine. But if you learned the ropes of the machine and would come around in the evening, or very early morning hours, you could get to run the machine yourself, which was a big kick.

O'NEILL: Was it more efficient to do it yourself, or did you just enjoy it?

DENNIS: It was both. It was more efficient because you could decrease your turn-around time substantially. It wasn't more efficient, because you probably made more mistakes in running the machine than the operators did. But you could say, oh shucks, I missed that, and go off, punch a few more holes in your tape and then come back to the computer room and run your program again.

O'NEILL: So you had a lot of early exposure to wanting to better use the machine and get a little closer to it.

DENNIS: Well, you know, when LCS had their 25th anniversary I gave a little talk, and one of the comments I made was that the nature of programs that we write today isn't that much different from the programs that we wrote for Whirlwind, but we have fantastically better tools for writing them, both in terms of programming languages and even more dramatically, the means of supporting interaction with the machine. I now have a Sun to work with on my desk.

O'NEILL: Those are nice [laughs].

DENNIS: At first I had to fiddle with paper tapes, punches and Flexowriters and things like that which were far more primitive. This change has been the biggest advance in computing.

O'NEILL: I haven't seen the paper that you gave at the 25th anniversary, has it been published?

DENNIS: Let's see... Oh, that has not been published yet, but it's in the works. I haven't done my piece of it [laughs].

I believe Professor Al Meyer is in charge, but other people are doing some of the editorial work. I think there are two parts to it. One is a first volume which is to be a record volume of the proceedings of the colloquium. That's what I was referring to. There's a later volume that is supposed to contain more carefully worked out papers to which I would like to contribute, but I haven't been able to set aside the time to do it.

O'NEILL: Okay.

DENNIS: So my background is engineering, electrical engineering specifically, and very broad, because I took lots of physics courses as well. So I have quite a bit of knowledge of physical sciences, which is pretty unusual for people in computer science these days. People educated in computer science typically do not have a very strong engineering background. That has made it possible for me to work on both hardware and software and feel comfortable on both sides.

O'NEILL: Was that more usual at the time?

DENNIS: It was more usual at the time, but still my kind of experience was unusual. There are other people who had the experience in the electrical engineering side, like Professor Fano who was in charge of the project but didn't have much experience in programming or software. Corbató, of course, is someone who has had physics background, so he knew the physical sciences from that side, rather than the engineering side, and of course, also has the software experience.

O'NEILL: Let's talk about when the PDP-1 showed up.

DENNIS: Well, first of all you understand that the plans to make the PDP-1 a time-sharing computer had been in the works for some time before that.

O'NEILL: That was what I was trying to get to. How was the time-sharing system decided on?

DENNIS: You know about the long range computer... What is the exact name?

Computer Planning Study, or something...

O'NEILL: Right. Long Range Study Group, no. Long Range Committee for the...

DENNIS: Computer Study Group, it may be [laughs].

O'NEILL: I have one of the reports in here; we can ...

DENNIS: There are two reports. There was a short report and a long report. I hope you have both of them.

O'NEILL: Long Range Computation Study Group. I have the short report here.

DENNIS: The long form of the report is, I guess, harder to interpret because it's harder to figure out the various pieces of the report, how much influence they have on what happened following that. But I think, is that April 1960?

O'NEILL: 1961.

DENNIS: Okay. The reason I was invited to participate in that committee was because of my interest in computer systems, and I was teaching a course about that time, a graduate course, 6.535, which was called something like Architecture of Computer Systems. That is the name I might choose today, I'm not sure it was the name at that time. But I was involved in trying to understand the nature of computer systems and what requirements were put on them. What was unusual about my approach to this subject was that I wanted to understand more precisely ... I'm not sure how to say this... in a more precise way, the nature of the interface provided by a computer system for the purpose of writing programs. To point out an example, at that time computers already had moderately elaborate operating systems -- nowhere as near as elaborate as they are now -- but it was evident to me that the facilities provided by the operating system were messing up the language that people were using to talk to computers. Because the operating

system facilities had to be looked at as an extension of the capabilities provided by programming language, and therefore in order to write programs for the machine, you had to not only understand the programming language, you had to understand the operating system as well. That was an additional burden that people really didn't recognize, or at least tolerated when perhaps in an ideal world, they shouldn't have to tolerate it.

O'NEILL: Were these particular programming languages that you're referring to, or just in general?

DENNIS: Well in general, but the situation has not changed significantly. We still have programming languages like Fortran, and LISP. Any programming language you name is not complete because it does not allow you to get at the facilities. It does not allow you to express within the semantics of the language the services provided by an operating system, the most notorious one being Input/Output or parallelism, or the idea of interrupt. People have tried very ad hoc ways of incorporating interrupts into programming languages, but they're always a mess. They're inelegant and do not have a clean semantics. So I would say that if there has been a fundamental direction to my work at MIT, it has been to try as best I can to overcome that problem. And so that was the direction of my thinking. My thoughts in that direction began from that time of my involvement in teaching this course, and I'm not quite sure when that was. I first started teaching 6.25 and 6.251; I'm not sure.

O'NEILL: Was that the course that was later turned into the book, *Machines, Languages, and Computation*?

DENNIS: No. That's another bit of history, which is not as relevant. *Machines, Languages, and Computation* is a book on computer science theory. It's not so relevant to the present discussion.

O'NEILL: Okay. So you were invited to participate in the Long Range Study Group because of your general interest in computing and...

DENNIS: Computer systems, particularly.

O'NEILL: At that time you had background with the TX0? Am I getting the timings right?

DENNIS: I'm not sure. I probably... I don't know. The timing doesn't seem quite right to me [laughs].

O'NEILL: I'm getting a little confused on the timings myself, so let's take a minute. Had you already worked out the time-sharing system on the PDP-1 before the Long Range Study Group?

DENNIS: I was in process of designing and building that time-sharing system during the time that the Long Range Study Group was meeting.

[INTERRUPTION]

DENNIS: Let me say one thing. With the TX0 I recognized the problem. People were using the TX0 interactively on a hands-on basis. I recognized some problems with the TX0 because it didn't have any auxiliary storage. There was no place to store a program, you had to load everything from paper tape. So very early in working with the TX0 I decided it needed some auxiliary storage. So RLE supported the purchase of a tape transport, and I designed the control system for it. Gordon Bell [laughs] helped me, and we worked together on that project. That's an interesting sideline. Gordon Bell was working on speech analysis, using the TX0 at the time. What seems amazing to me is how many things happened in the short time after I completed my thesis for the doctorate in the summer of 1958. Until then I was working absolutely flat out on the thesis. So I could not have started anything with the TX0 until fall of 1958. My guess is that in the fall of '58 I probably started right in working with the TX0. The first thing I did was to write an assembly program for the TX0, a macro assembler.

O'NEILL: A good thing to do, I'm sure, if you were working in machine language [laughs].

DENNIS: Well, the only programming tool that came with the TX0 from Lincoln Laboratory was an assembler, written by Munroe Weinstein, I believe, which was very primitive. In fact there was no assembly language listing of the

assembler itself. The only way to figure out what it was doing was to have the contents of memory printed out in octal code. The first thing I had to do was disassemble the assembler and figure out how it worked [laughs].

O'NEILL: Sounds very challenging [laughs].

DENNIS: Well, it's of the same order as disassembling things now. It's not an impossible task.

O'NEILL: When you were back working on the PDP-1 were you familiar with the BBN group that was working on a time-sharing system there?

DENNIS: Yes. In fact... have you interviewed John McCarthy?

O'NEILL: I haven't personally, but he'll be interviewed as part of the project.

DENNIS: John McCarthy was the premiere advocate of time-sharing. I was considering a whole bunch of improvements to make to TX0 to make it a more useful machine. John McCarthy came along and said, "Gee, why don't you make it a time-sharing machine?" So I immediately started thinking about how I was going to make the TX0 into a time-sharing machine. Within a year there was the proposal of bringing the PDP-1 in, and so we started thinking, well how am I doing to make the PDP-1 a time-sharing machine [laughs]? John McCarthy inspired both the work here at MIT, and at BBN. Through McCarthy, and Fredkin I guess, we learned about the PDP-1 at BBN and we learned about BBN plans, and we coordinated plans to the extent that we both bought the same auxiliary storage drum through Digital. The drum used a nifty idea of Fredkin's; it could swap 4,096 words between drum and magnetic core memory in one revolution. The PDP-1 time-sharing system could switch between users in slightly more than this time.

O'NEILL: Did you have a lot of interaction with the people at Digital?

DENNIS: Oh yes, there was quite a bit of interaction. Let's see, I don't know how to characterize it. For example, there was a lot of software written for the PDP-1, which was inspired by related software from the TX0. The PDP-1 software was being written by undergraduates and students who joined the community surrounding the PDP-1. For example, the macro assembler which I had written inspired the macro which was written for the PDP-1, and that became a product of Digital.

O'NEILL: So DEC was very open to what was going on?

DENNIS: Yes. In fact, Allan Kotok was one of the people involved with the PDP-1. He went on to design the instruction set of the PDP-10. Many of the hardware features we put into the PDP-1 reappeared later in the PDP-11/45. I'm not familiar with the details of how that happened.

O'NEILL: Did DEC have technical people here working with you at all?

DENNIS: No.

O'NEILL: Did you interface with salespeople at DEC?

DENNIS: No. If we decided that we wanted something, we would call up DEC and say, we want this, and there would be somebody that we should talk to.

O'NEILL: I see.

DENNIS: That was in the era in which DEC was selling machines by building special interfaces or providing special services to customers. They provided a lot of information and guidance for installing the drum, for example. DEC did not have a standard drum system for the machine, and as I recall, we bought the physical drum from Vermont Research, which was a supplier to DEC. We got information about Vermont Research from DEC, but then all of the

arrangements to requisition the hardware and install it were done by us. We used a lot of DEC equipment. We used logic modules from DEC, we bought their cabinets, we bought their power supplies and things like that.

O'NEILL: DEC was one of the first manufacturers to market time-sharing, and I was wondering how much they picked that up directly from what had been going on with the early machines.

DENNIS: Not very directly. There was no interest in DEC, for example, in copying our time-sharing system.

O'NEILL: Okay.

DENNIS: The sequence of events which happened there is kind of interesting. One of the people who worked with me on the guts of the time-sharing system for the PDP-1 was Peter Deutsch. Have you run across that name?

O'NEILL: No, I haven't.

DENNIS: He is the son of a physics professor here, who started coming in and playing with the TX0 when he was in high school. He was pointing out to the graduate students the errors in their programs [laughs]. On the PDP-1 we had 4,000 words -- the total amount of core memory on the machine at first. And it ran with those 4,000 words of memory in the original time-sharing system. The way the time-sharing system worked is there was a small part of that memory that had a little kernel of an operating system in it. It occupied about 500 words of memory. I wrote that. I wrote the original version of that, and carefully designed it to provide the basic services of the electric typewriter interface, interrupt handling for quite a variety of input/output equipment in that 500 words. Peter Deutsch became very familiar with the kernel, and took it with him to California. And the same system was almost identically implemented on a SDS940 machine at Berkeley. I think the system at Berkeley was one of the earliest time-sharing systems installed in a college outside of MIT, other than Dartmouth, I guess. Dartmouth had been a leader in implementing an early time-sharing system based on a GE235 machine. I can't recall any other university projects. The Dartmouth system was somewhat restrictive because it basically would serve multiple users using the BASIC

language, whereas the one done at Berkeley was similar to the MIT systems, which allowed general purpose use of the machine. The user could choose his own language or subsystem to work with.

O'NEILL: When you were on the Long Range Study Committee, was there a systematic attempt to study whatever other systems were available or was there just a general knowledge of what was going on?

DENNIS: What the Committee was trying to do was define the characteristics of a computer that would satisfactorily support time-sharing service for our community. So the committee itself did not specifically go around and visit manufacturers or try to learn about other systems. The Committee assumed that each of its members had considerable knowledge from their own contacts, which we did. There is a close relationship between time-sharing and the concept of multi-programming. What was important to us was the nature of previous multi-programming systems and the ways in which those multi-programming systems failed to provide adequate support for time-sharing.

O'NEILL: So you thought going into the Study Group that it was to specify the machine requirements for a time-sharing system?

DENNIS: Yes.

O'NEILL: So time-sharing was already firmly imbedded as the way to go to provide computation?

DENNIS: Yes. The committee was never asking the question, is time-sharing the right way to go. We were assuming that.

O'NEILL: Okay.

TAPE 1/SIDE 2

DENNIS: A lot of the discussion was connected with parallelism, and multi-processing, and multi-processor architecture, and memory management. A lot of time in the Committee was spent on these subjects, and also the characteristics of an instruction set architecture for a suitable machine.

O'NEILL: Okay, so you were going into fairly specific areas of the machine itself.

DENNIS: Yes. A number of the people on the Committee felt that a very good way of proceeding would be to build our own machine.

O'NEILL: Yes, I think that was listed as one of the alternatives in the report. I guess another alternative was to get the STRETCH?

DENNIS: Oh, the STRETCH ... was STRETCH proposed?

O'NEILL: Well, STRETCH was proposed in the other report actually.

DENNIS: I don't think the committee ever proposed it. There was a lot of discussion when Project MAC was first formed. In fact in connection with its forming, Licklider proposed bringing in an ANSQ7 machine, because these were being dumped by the government.

O'NEILL: That was the Q32.

DENNIS: And there was a lot of tension surrounding that, because here was a guy from whom we were going to get all of our money proposing what we thought was a ridiculous idea [laughs].

O'NEILL: Why did you think it was ridiculous?

DENNIS: It was an old machine, it had very limited memory and addressing capabilities. It was very much unaligned with the requirements that the committee had written down.

O'NEILL: That leads us into the ARPA funding that comes along after the committee report has been finished. Between the time the report was finished and the ARPA funding came around, did you see any progress on the report itself in terms of talking to manufacturers about what they had available or taking any other steps?

DENNIS: It was only after the summer session of 1963, the Summer Study of 1963, and funding of the project, that we undertook our visits to manufacturers. Once the project was funded, we had the responsibility for making a choice. I think with the funding of the project, a decision was made that we would have a commercial company, a computer vendor. We would obtain our hardware as commercial hardware from a vendor in the industry. So the next step was to decide who that vendor would be.

O'NEILL: Did you personally visit manufacturers?

DENNIS: There were four of us who formed a committee that was given the responsibility to do that. It included Corbató, Ted Glaser, Robert Graham, and myself.

O'NEILL: What were your visits like? Was there a lot of interest in your project? Was there a lot of interest in providing you with a machine?

DENNIS: There was considerable interest in the project. In fact, in general I'd say all the manufacturers we visited expressed considerable interest. I think their interest in many cases was just keeping up with what was going on at MIT and understanding our viewpoint, not necessarily feeling that they were the ones to help us out. Of course, most of the manufacturers were willing to sell us any of their standard products [laughs]. Some of them saw that their products were not very suitable, and not a good match to our needs. Two manufacturers were more seriously

interested, IBM and General Electric. IBM just put on all the force they could muster. One particular personality in IBM was Gene Amdahl, who just couldn't believe that the new IBM system... See what was happening is that IBM System 360 had not been announced at the time we were visiting. The 360 was announced in April of 1964. IBM wanted to believe that that was a forward looking computer system that would solve everybody's problems. After all, they had put years and years of effort of their best people into figuring out what IBM System 360 should be. And here we are telling IBM, a couple of months before their major announcement, that their architecture does not meet our needs. That was it. So IBM put in a fantastic amount of effort to try to convince us that System 360 really was what we wanted. Gene Amdahl was one of the people in IBM whom I had a long private conversation with on what our needs were and what his view about what computation was like.

O'NEILL: Did they not want to do time-sharing, or did they think they could do it on their machine?

DENNIS: Oh, they were happy to accept our interest in time-sharing, even though they didn't believe it in their guts, but they were happy to sell their machines. So they looked at it as, oh you want to do time-sharing, you've got all this government money to do time-sharing. Sure, our machine is good for that. That was their viewpoint. Or maybe to say it more accurately: "We are determined to prove to you that our machine is good for that, because our machine is the best one that we could possibly [laughs]..."

O'NEILL: Can you remember back at the time, did you think that time-sharing was a general solution for computing, or did you see it as the solution for MIT, and maybe other universities? Did you have a broader view?

DENNIS: I don't think it's either of those two. Computers, even at that time, were into all sorts of things like industrial process control, for which no one would think of using a time-sharing computer, at least not how we meant time-sharing. By the way, there was this ambiguity, because the term time-sharing appeared in the literature earlier where it simply meant division of the computer processor's capability among different activities. It didn't imply the human interaction at all.

O'NEILL: It was more a multi-processing idea.

DENNIS: Yes. And the requirements for that are considerably less strenuous than providing support for an interactive environment. So there is that problem. For example, the 360 is a pretty good machine for multi-programming, and multi-programming capability is a part of what's necessary to do time-sharing with, so they felt they had the problem solved. But they didn't.

O'NEILL: I've read some things that indicate that the GE proposal was much closer to what MIT had in mind.

DENNIS: When we visited GE, they were working on the GE635. The responsible engineer was John Couleur. Do you have that name?

O'NEILL: No. One of the next areas for our study is looking at industry and trying to get some people there whom we can interview.

DENNIS: Well, John Couleur is a very important person in this business, because John was able to work effectively with Ted Glaser. John understood the nature of what we expected from the hardware and realized the 635 was as close as anything in the industry to it. He was willing almost immediately to start thinking about how they could make the machine better for us, which was very different from IBM's approach. IBM's approach was to say our machine is the one you want, period [laughs].

O'NEILL: So GE was much more flexible.

DENNIS: We stopped trying to talk with IBM about modifications. What do you want modifications for? Our machine is what you want. GE, because of John, said to us, ah, here's a new, interesting area that we could apply our basic architecture to. Maybe we can work out how to fix the addressing mechanism of the 635 to match with the ideas that these MIT people are interested in pursuing. John was open to this, excited about it, and the company let

him do it. There were two ingredients, both of which were missing at IBM. At IBM, the technical people weren't willing to understand and work with us, and the company as a whole wouldn't back any significant changes.

Visits to other companies were of a somewhat different character. They would show us their wares, describe their products, we'd see their laboratories, all of which is very interesting and very impressive to us, but we would see that their machines just don't match up with our needs. In particular, we were very much sold on the idea of having a multi-processor architecture. So we would go to these companies and ask them, well you're not offering us a multi-processor; what would you do if we want a multi-processor computer? Now, Corbató was interested in multi-processing because of the reliability aspect; what happens if the machine goes down. If you have two processors, then you can continue running the machine, even though one has failed. I was interested in multi-processing because I felt and believed that it would be a more efficient way of getting greater capacity from the machine. I recognized the reliability merit of it also. My argument was that having several processors would make the memory work harder.

O'NEILL: I see. More memory references per processor. The Project MAC IBM 7094 or 7090... the Computation Center had the 7090, and Project MAC got the 7094, I believe.

DENNIS: I'm not sure of those numbers myself.

O'NEILL: In any case, it seems that the first Project MAC machine was always seen as sort of an intermediate step, that a new machine would be forthcoming from a manufacturer. So that first machine was never seriously considered, it was just a stop gap?

DENNIS: Well, you'll have to ask Corby about that because the work on the 7090 in the Computation Center, again, was inspired by pressure from McCarthy to get a more convenient way for him to do his AI research [laughs]. He was demanding better access to the computer and encouraging people to do it. The first actual time-sharing on the 7090 was actually set up by Herb Teager. I think Corbató, initially, was reluctant, but with pressure from McCarthy

and pressure from Teager, he recognized that this was going to be very important. He was going to have to figure out how to deal with it. So then he got involved. In contrast to Teager, Corbató wanted to view the whole thing as a university-wide service, whereas Teager was more looking at the time-sharing arrangement to serve individual scientists, researchers who could benefit from the expanded capabilities of the computer. So you understand the difference? Corbató was looking at it as a service to a large community, and Teager was looking at how can we make progress in these various fields which need better interaction with the computer. So Corbató won out, and Teager left MIT to go to Boston University as a result of that conflict. When Teager left it also meant that the support for graphics and post-interaction was given a secondary role in the computation center until Doug Ross and his people came and put the Kluge on the machine. I remember these things happening over a span of two or three years.

O'NEILL: It's amazing how much was going on. Did you have a lot of interaction with people from ARPA? You mentioned Licklider coming around and talking about giving you a machine.

DENNIS: No, not personally. In fact, the only interaction with ARPA people... well, I had some interactions... Larry Roberts was a user of the TX0. In fact, he did his handprinted character recognition program on the TX0, so I knew him at that time. We both appeared in the CBS program "Thinking Machine"... the one that also had Doug Ross' play, Western, generated by the Computer.

O'NEILL: I haven't see that, I've heard of it, but I haven't seen it.

DENNIS: Well, you have to see it [laughs]. Both Larry Roberts and I have bit parts in this film [laughs]. Ivan Sutherland was also an early user of the TX0. He had a neat interactive program that he wrote for the TX0, prior to his going to the TX2 and doing Sketchpad.

O'NEILL: Oh, okay. So a preliminary to Sketchpad. Was there ever any concern with doing command and control research, or having military relevance or any of those kinds of issues within the MIT community? I mean, was ARPA viewed as the military, or was it just funding?

DENNIS: Oh, ARPA was generally viewed as the major - this is over the whole history of ARPA support - as the major supporting government agency in computer science, and we tend not to think of DARPA as a defense establishment, because only until I guess the '70s was there any hint of putting military requirements into the whole business of contracting with DARPA. There was a time in which the MULTICS effort was threatened to be shut down, around 1967, because it hadn't become operational yet, and the government was putting all this money in and it seemed like they weren't getting anything for it. So the whole MULTICS effort was hanging by a thread for a while around 1967. It was at that point the DARPA/IPTO became more insistent that there be sort of operational justifications. The nature of the research was always basic or fundamental research; we had never gotten into any development work for DARPA. Our work has been detached from military requirements, generally. But we always had to be able to make the argument that this would be supportive of some military need.

O'NEILL: But in a sort of abstract way, not really very tied to military needs. Okay.

DENNIS: But in the early days, the first seven years of Project MAC, there was not even that requirement. There were two fundamental reasons for supporting computer science research. One was development of time-sharing, and the second one was getting the graduate students educated in computer science. A general way to strengthen technology in the country. They almost didn't care what we worked on [laughs].

O'NEILL: Sounds like a good position to be in [laughs].

DENNIS: It was good years.

O'NEILL: I know you were on the MULTICS planning committee and were obviously involved in the GE choice. Did you actively work on MULTICS after that?

DENNIS: No. By April of 1964, when the choice was made to go with GE, I had been working on the conceptual

addressing mechanism that would be used in MULTICS. I wrote a couple of papers having to do with this work. Then the MULTICS team carried on from that. My choice was either to devote my energy to the MULTICS effort, which would have required all my time, or to devote my time to other things. I decided not to participate with the MULTICS team because I wouldn't be able to explore my ideas. The reason is the work on MULTICS had to be frozen into a design very quickly, and the design that was chosen was not what I would have done.

O'NEILL: Were you aware of that before you made the decision?

DENNIS: Yes. I knew the direction they were going. I wasn't entirely happy with it. I later realized that it was done basically for pragmatic reasons, which were necessary in order to get the job done.

O'NEILL: Were you still working on Project MAC projects?

DENNIS: Oh, yes. I joined Project MAC in '63, and formed a group which originally was called the Machine Structures Group, and shortly thereafter, the Computations Structures Group. The name change was in recognition of the fact that it's not just hardware we were thinking about, but the organization of a system to support programming. So for the rest of my career in the Computation Structures Group I've been focusing on those issues without being tied to a particular project, or system building project. I think it was the right choice for me.

O'NEILL: When you were involved after the GE choice was made, was that when Bell Labs got involved as well? Were you involved in any of those negotiations?

DENNIS: No.

O'NEILL: There's another whole side story there.

DENNIS: Basically my connection with MULTICS stopped in spring of '64.

O'NEILL: Okay, that early. Pretty much right after the GE decision was made.

DENNIS: Let's see, there was a session in one of the joint computer conferences where the MULTICS plans were presented, I think it was 1965, I'm not sure of the date.

O'NEILL: Yes, I believe that's right.

DENNIS: I did not present a paper. I was not invited to present a paper there because I was not a project participant. However, I organized the first symposium on operating system principles which was held in 1967 in Tennessee. This grew out of my being part of an ACM special interest committee, called Special Interest Committee on Time-sharing.

O'NEILL: I didn't realize they had one on time-sharing.

DENNIS: Yes. Well, I suppose it was to a large degree because of my influence that we decided that time-sharing was not an appropriate area for an ACM committee, so we changed it to Committee on Operating Systems, and then it became the Special Interest Group on Operating Systems of the ACM. It was about the time of the transition when I organized this conference.

O'NEILL: That was the Gatlinburg conference?

DENNIS: That was the Gatlinburg conference in 1967. I and my co-chairman organized it. We tried to contact all the people we knew to get papers for this, and by a few weeks before the meeting we only had twelve papers [laughs]. So I said, well, gee we've got to have more papers than that, or else we're not going to be able to fill up the program. So I got together with Bob Daley, and we wrote this paper on Processes and Virtual memory in MULTICS. I also wrote the paper on communications. Both of those were afterthoughts because we didn't have enough papers. The paper I wrote with Daley had very strong connections with MULTICS, was basically intended to be an explanation of

what I felt were the key mechanisms in the MULTICS system.

O'NEILL: We've covered most of the questions that I had prepared. I did want to have an emphasis on the years right before DARPA's funding, and then when that came around and into MULTICS. Do you have any general kind of comments you'd like to make on this whole business?

DENNIS: Well, okay. MULTICS suffered from a couple of things. One was the fact that the pragmatic choices which had to be made in order to continue the project were such that MULTICS ended up being an inefficient machine from a hardware point of view. That's sad, because it probably prevented MULTICS from becoming a widely accepted product. I suppose the sadness is that maybe it could've been made much more efficient if a different sort of inner design had been chosen. But this is speculation. I guess the other thing is if a better inner core had been chosen for MULTICS, perhaps the software system would not have been as complex as it turned out to be. But on the other hand, maybe that was preordained also. It's hard to tell.

O'NEILL: It was certainly an ambitious project.

DENNIS: Certainly. What actually happened is that microprocessors came along, and distributed computing came along, and that turned out to be so economically attractive that the advantages MULTICS had to offer in terms of being able to build sub-systems, and all the protection capabilities in it, and so forth, the things that I was emphasizing at the time, did not have sufficient value to the user. Now, my view is that what has happened is that this simply injected a ten-year or twenty-year pause in the evolution of computer systems. Now that we're going back to workstations and servers, all of the same issues are in front of us again, and solutions have yet to be worked out. The solutions will be along the lines we were thinking about back in the early '60s.

O'NEILL: Now there is more of an imperative to do it.

DENNIS: There's more of an imperative to... Well, I don't know. Everything in this industry is done in direct

response to user needs on a short-term basis, not a long-term basis. Just like financing these days; everything is short-term rather than the long-term. And also the computer science community doesn't agree with itself on what are the right directions to go. So it's very difficult to set in motion a direction-setting project [laughs]. There could be a project now which would be as significant, in my view, as MULTICS was, but it would be impossible to do because you wouldn't get a group of people to agree, now, that that was the right thing to do. It was possible in 1963. It was possible in 1963 to get practically all the faculty in computer science agreed that this was the right thing to do. You wouldn't be able to get that now.

O'NEILL: Even within MIT you wouldn't be able to?

DENNIS: Even with MIT you wouldn't be able to get that.

O'NEILL: Did you find dissension in '63 in other communities, other universities?

DENNIS: Well, there were reactionaries around. They said, you know, I don't need time-sharing, the old way is the right way, and so forth.

O'NEILL: Was that a widespread reaction?

DENNIS: Around the country it was fairly widespread. Read the time-sharing literature, and the panel discussions held a conferences, and so forth. People were knocking the time-sharing idea left and right. But at MIT within the computer science community we were all very much in favor of it.

[INTERRUPTION]

TAPE 2/SIDE 1

DENNIS: Let's discuss the ideas of segmentation and paging. The MULTICS machine, the GE 645, is the first machine in which these two ideas were combined and implemented in one piece of hardware. Let me point out what had gone on before. The idea of paging a memory, which is that of dividing the address space into equal size blocks which could be brought into core memory on demand, is attributed to the Atlas Machine at Manchester, was done in support of a multi-programming concept. It was a mechanism to support the concurrent operation of several different programs on the machine at the same time, a single processor machine. Prior to that, the Burroughs Company had built a series of machines starting with the B5000. The B5000, I think there are articles on it that date from about 1958.

O'NEILL: I didn't realize they were that early.

DENNIS: I don't have an exact date for the introduction of the B5000 machine. Well, there are many interesting things about the B5000. First of all it was a multi-processor machine, the first commercially successful machine that was a multi-processor. It had an operating system that could run simultaneously on all processors. That's something which, in order to achieve, would have to be fairly sophisticated in both the design of the hardware and in the design of the operating system. This was a really major, major achievement for the Burroughs Company, bringing out the B5000. Now the B5000 was thought of as a machine to support the ALGOL 60 programming language, and therefore it was equipped with a stack. But it also had another mechanism that involved the use of things called descriptors, which were basically pointers to areas of memory which could be used for storing segments of programs and arrays of data, in a way that the memory for these objects could be managed effectively by the machine. Each segment had an arbitrary length -- arbitrary number of words. The machine and its operating system had the problem of fitting all these arbitrarily sized segments of memory into the physical memory of the machine -- a pretty hard thing to do. In fact, the Burroughs B5000 was a fairly inefficient machine in part because of this problem of dealing with memory management issues generated by the feature of segmentation in the machine. In MULTICS we wanted to provide each user with a sufficiently large address space that he could link his programs to any programs that were stored in the machine. The use of segmentation was a means to achieve that very large address space. The use of paging was a mechanism to make it operate efficiently, particularly on a multi-processor machine. My major contribution to MULTICS was the recommendation that they combine the two ideas for these purposes -- to achieve a large address

space and also to make the storage allocation mechanism efficient when used on a multi-processor configuration.

O'NEILL: So that allowed the user to be moved around through pages, rather than having to worry about the whole segment line?

DENNIS: Well... okay. Each segment was broken up into pages, so that instead of consuming... a segment might be 67,892 words long, okay, and to find that size space within the machine might be a very difficult problem. However, to find thousand-word blocks here, here, here, and here, is a relatively easy thing to do. You can have confidence that that process can continue and you don't have a deterioration setting in as the computer runs and space becomes more fragmented because of the odd-sized pieces that you're trying to allocate.

O'NEILL: Were you concerned about the problems of actually managing the page locations?

DENNIS: I wasn't concerned about that problem because I regarded that one as a relatively easy one to solve. I was advocating providing the user with the very large address space which was made possible by the concept of segmentation. The difference between MULTICS and what I would like MULTICS to have been is that each process in MULTICS has a separate, very large address space. I wanted all processes to operate in the same, hugely large address space, so that processes could talk to each other without having to have their messages translated. See, every time one process talks to another in MULTICS, the process has to have its language translated to talk in the language of the other process, so to speak. That makes for inefficient communication between the processes. So interprocess communication in MULTICS is more expensive and more complicated than it would have been otherwise. Also because of the way segmentation was done in MULTICS, and these separate address spaces for each process, one result of that is you cannot allow a process to run forever. It has to clean up its address space eventually, and that interferes with the elegance of programming MULTICS supports. So you can do elegant programming in MULTICS, provided you don't run out of memory [laughs]. But if you're going to keep yourself from running out of memory, you either have to not run very long, or you have to clean up your address space, which interferes with the quality of programming.

O'NEILL: So you always have to be aware of your size, in other words.

DENNIS: Yes. That's the limitation of MULTICS which would not have been there if we had adopted the more universal idea we talked about. Now, that universal idea has been implemented in a machine by IBM. The IBM System 38 implements the idea of globally valid unique identifiers for objects in the system. That has been carried forward now into system AS400, also of IBM. However, they have put way too much complication into software layers built on top of this. Again, IBM's usual thing of trying to solve everybody's problem, rather than counting on third-party software houses. What you really have to do as a manufacturer these days, is to build a sub-stratum on which other people can build software. IBM has been unwilling to do that, because IBM is looking at the field and saying, ah, we're not going to make our money on hardware anymore because everybody has already spent as much as they're going to on hardware, so we have to make our money on software. If we're going to make our money on software, then we want to make sure that it is we who are providing the software. By doing that they're ruling themselves out of a large part of the market, because people are preferring to buy platforms on which they can run everybody else's software. That's the way the industry is going, and why IBM is not doing so well. Okay, I guess that's it. See if you can sort that out [laughs].

O'NEILL: Yes. I have plenty to think about. Thank you.

END OF INTERVIEW