An Interview with

FERNANDO J. CORBATO

OH 162

Conducted by Arthur L. Norberg

on

18 April 1989
14 November 1990

Cambridge, MA

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

Fernando J. Corbato Interview
18 April 1989
14 November 1990

Abstract

Corbato discusses computer science research, especially time-sharing, at the Massachusetts Institute of Technology (MIT). Topics in the first session include: Phil Morse and the establishment of the Computation Center, Corbato's management of the Computation Center, the development of the WHIRLWIND computer, John McCarthy and research on time-sharing, cooperation between International Business Machines (IBM) and MIT, and J. C. R. Licklider and the development of Project MAC. Topics in the second session include: time-sharing, the development of MULTICS by the General Electric (GE) Computer Division, IBM's reaction to MIT working with GE, the development of CTSS, the development of UNIX in cooperation with Bell Labs, interaction with the Information Processing Techniques Office of the Defense Advanced Research Projects Agency, interaction with Honeywell after they purchased GE's Computer Division, and the transformation of Project MAC into the Laboratory for Computer Science.

FERNANDO J. CORBATO INTERVIEW

DATE: 18 April 1989 INTERVIEWER: Arthur L. Norberg

LOCATION: Cambridge, MA

NORBERG: Today is April 18, 1989. I am in the office of Professor Fernando Corbato of the Massachusetts Institute of Technology in Cambridge, Massachusetts. We are here to talk about the timesharing developments and work in CTSS and Project MAC. It is my understanding that the Computation Laboratory began in 1956. Is that correct, or is it a little bit earlier than that?

CORBATO: No, that is about right. It was called the Computation Center, and it was organized by Phil Morse, who was a professor of physics. He died a couple years ago. Let's see, there is this tape which J.A.N. Lee just did...

NORBERG: Yes, I was present for about half of that.

CORBATO: Yes, that is one resource. Another resource which exists is a videotaping session which occurred about five years ago.

NORBERG: Yes, 1983 I think. The one that Pool and...

CORBATO: Pool and Bob Solomon did.

NORBERG: Yes, I haven't seen that, but I know of its existence.

CORBATO: It was two different days. One was on the computing environment in the Cambridge area which led to timesharing. The other was on the... Well, one was on computing in general and the other was on timesharing. I think that was the dichotomy. So there are two different tapes. I think I loaned one of my tapes out and I have not gotten it back.

NORBERG: It seems to me they are at the Computer Museum now as well, aren't they?

CORBATO: I would hope so.

NORBERG: I think so. I can go over there tomorrow and look at those because I was going to go over there anyway.

CORBATO: Bob Solomon, who is out at the University of Massachusetts, was kind of the inheritor of the archives. There were some caveats about how they were not going to be shown to anybody because they were raw tapes. They never edited them. But I have looked at them, and they are fine as far as I am concerned. Anyway, you asked about 1956.

NORBERG: Right. I am curious about your activities in the Computation Laboratory in that period.

CORBATO: Yes, I think you have to have some of the background. Computation was not recognized as being a major thrust here until WHIRLWIND. It really began with WHIRLWIND. Before that there were fragmentary things. There had been Van Bush's analog devices in the thirties, and there was some wartime work apparently trying to develop better analog devices that Frank Reintjes and others were involved in. This is touched on briefly in a history of the department that was done.

NORBERG: Yes, Lindgren and Wildes' book.

CORBATO: Yes, it was. In the middle of the war, they came up with this notion of trying to build an aircraft simulator. The Servo Lab Group, led by Jay Forrester, took on the role. The saga is documented elsewhere. It evolved into the WHIRLWIND computer rings (?), and switched from an analog design to a digital design. The war, meanwhile, had finished. They had not quite gotten it going, and they, in turn, carried it forward under ONR sponsorship and first began to operate it, to my recollection, about 1950. It was a very primitive thing, but it was

fundamentally a von Neumann machine, except that it had parallel access to the memory and therefore was faster, and was designed around the premise that it could be a real time machine. So it was fast for its day -- 24 microsecond cycle time. It also used a form of storage tube that was crude, home designed; it was expensive, but more reliable. Phil Morse, who was kind of an entrepreneurial person, saw this as an opportunity to start a computing climate. So he, I guess, approached ONR and raised the question of starting up some interdisciplinary research assistantships, about a dozen among graduate students, which he, in turn, pointed in the direction of working on the implications of modern computation. Morse himself was a physicist who had done hand calculations. He knew full well the grimness of hard calculations. I was one of those research assistants. I was a physics student. There were also students from math.

NORBERG: This would be roughly 1952, say?

CORBATO: 1951. We first cut our teeth by exploring the then-existing punch card equipment, a little bit of which was on campus, and some of the card machines. The late Sam Caldwell and Frank Verzuh had done some work with a machine that IBM made called the 602A, which was a calculator which did primitive arithmetic between the card stations as the cards were moving by. It would take the data off one card, compute and punch it in the next card. All right, so I and quite a number of other graduate students got a chance to start exploring WHIRLWIND as it was just coming up.

NORBERG: When you say quite a number, how many do you mean?

CORBATO: Well, there was a dozen of us who were on this program, plus probably another dozen or two who were involved in research projects where, for some reason or another, they had gotten pointed at using a computer. This is a nonclassified use of WHIRLWIND. The management of WHIRLWIND under the leadership of Morse sort of threw it open for grabs, as I recall but maybe I am wrong, in order to have an arbitrary use within the campus to help seed the uses of computation.

NORBERG: Do you think this was Morse's intent, or did this just sort of happen?

CORBATO: No, Morse's intent was to try to pursue the advantage of computation (?). He also started OR and had his hands in other fires too. So in the early 1950s, we began to use WHIRLWIND hard. We were very rationed in our ability to use it: we could only use it for a half-hour or an hour during the day -- about five o'clock -- and about two hours -- from two to four in the middle of the morning. So that meant some crazy hours. It meant you were working with difficult equipment, punch paper tape and all that, and we were facing all the growing pains of the industry. It was not an industry yet. The equipment was very erratic and difficult. The reliability was in error; the machine would crash every 20 minutes on the average. IBM meanwhile had begun to work on the 701, 702 computer. That was their first commercial product. It evolved into the 704 computer; in about the mid-1950s IBM also decided that they wanted to establish some university centers of computation which they would help support in some way. I would assume that their motive was primarily to create people who knew their machines and wanted to use them. So everybody had a self-interest involved. Morse saw this as a chance to capture one of those centers, so he proposed to IBM that they let MIT run one such center. To make it more attractive to them, he proposed that it be a regional center of the New England colleges. So the deal he cut with IBM was that they would install a 704 computer (which I guess got installed in 1957, but the deal was cut in 1956), while we in turn would make one shift of it -- that was an eight hour shift available to MIT, one shift of it available to the New England colleges, and IBM would continue to retain the use of one shift, which they would use for their local support office. So Morse in turn organized the Consortium of the New England Colleges, including most of the major players and a lot of the minor ones. He got NSF support to help him to run that program. As I recall, the assistantships of ONR were phasing out so he got some new ones from IBM. I believe that was the source. IBM in turn set up a support office of people to be a liaison with us. Then IBM in turn helped us augment the building that was just being constructed to include space for the center. So MIT came out with a nice package. That space has since been taken over as the physics reading room, if you ever go in that direction. In fact, the way he was able to sneak it in (because the building was nearly completed), was that to the consternation of the architects, who had built one of these faddish building of the day -- it was on stilts -- he convinced them that they had to fill in the ground floor, which was air before, with a floor. They were angry; (laugh) they tried to hide it by making the outer wall in blue tile to make it look like sky, and they put in no windows,

except only a high transom. It turned out to be a nightmare to open and shut. (laugh) That is the legacy of that. So in about 1957, moreover, the Air Force and ONR and all the parties concerned were running out of steam paying for WHIRLWIND, so that it was planned to phase it down and out. I guess Morse just knew it was coming; that was one of his reasons for trying to work with IBM. So this became the replacement. It took a couple of years to get going, it turns out, but even though [WHIRLWIND] was a pretty darn good computer it was difficult to maintain and expensive. It had acted as a prototype of Air Defense. I guess, in effect, I knew this more afterwards than at the time. It spawned off Lincoln, and later MITRE, and the whole SAGE system of deployment. So that was how we got to where we were.

NORBERG: Now, can you go back to your role in the Computation Center in 1956 through 1959-60?

CORBATO: I had been a graduate student in physics. I wandered around a little bit, and then sort of homed in on molecular physics working in John Slater's group. John Slater was my advisor. I got into an energy band calculation, used WHIRLWIND -- I used it very hard -- and I gradually got more interested in the calculator than I did in the problem. By the time I was nearly finished with my doctorate, Phil Morse approached me as he was organizing the center, and said, "Would you like to help manage the center and be a research associate in it?" I thought about it a little while and agreed. So in effect I stayed on as a kind of post-doc. My specific role was to be the person who helped allocate and administer the research assistantships, which Morse had realized we would need to augment the use of, because Morse was fairly wise in managing. He was a very shrewd manager. He saw that the goal was to not only to get access to a computer, but to augment it with a staff of people who had interest in the research aspects of computing, both at the graduate level and at the research staff level. So in fact the center actually had some research contracts too. It had associated faculty and research contracts for several years.

NORBERG: What sort of research contracts?

CORBATO: The research was in numerical analysis, I think, on one occasion. Eventually it was in machine architecture, and eventually timesharing. The research faculty that were involved were people like Dean Arden, who

had been on our faculty; John McCarthy. (In fact, John McCarthy had been at Dartmouth many years and had become sufficiently interested in the environment of MIT that he moved to MIT from Dartmouth. He switched.) Herb Teager was also a key player in the early days, and Marvin Minsky, who was associated with John McCarthy.

NORBERG: Were these people all either faculty who were affiliated with the center, or were they like yourself, research associates in the center?

CORBATO: The ones that I mentioned were all faculty. In the case of contracts that they managed to obtain, some of their salary support came from those contracts. Also in some instances the contracts had research assistants written into them, and the research assistants associated with them were supported in that way.

NORBERG: Did you have any of these contracts at the time?

CORBATO: No, I never had. I did not get any. I was working under a kind of a blanket contract that Phil Morse got for the center. I was working on them; I did not have them. My role evolved with time. I was initially just handling one wing of the effort. Then at a later time the associate director left.

NORBERG: Who was that?

CORBATO: Frank Verzuh. I in turn replaced him. I do not know if my title changed or not; I forget. It might have been assistant or associate director, but I took on the role of basically being Morse's acting director. Morse's style of management was he would put in one day a week at it. He would show up one day a week, and of course, if there was a crisis or a problem he was on the phone. So that meant the person who was running it for him had a lot of leeway to do things, and a lot of responsibility to make things happen, because we were also running the Computing Center with a fairly big staff, operators and programmers...

NORBERG: I guess I am a little confused. I did not realize that there was a difference between the Computation

Center and the Computing Center.

CORBATO: I am sorry. It was the same. I use the word loosely. Formerly it was the Computation Center. In the early days there was only one such place. For many years it had a de facto monopoly, which gradually eroded away with the advent of, basically, mini-computers.

NORBERG: So this monopoly would have lasted about ten years then, if that was the case? Don't you think?

CORBATO: It did not last quite that long, but that is approximately right. People were making special case arguments about why they had to have their own equipment.

NORBERG: Who was the first one to get another set of equipment? Do you remember?

CORBATO: I am not sure I have got the first one nailed down. One of the consequences of working with IBM was that the center would keep upgrading its equipment. IBM retained title on the equipment, and we would jawbone them until they would finally relent and give us a slightly better machine. It was never quite as early as we wanted it. So we went from a 704 to a 709, and then later we went to a 7090. The 709 that was released was bought by John Slater's group. They claimed that they could somehow run it economically themselves, giving themselves wholesale rates. Michael Barnett was the person who promised to run it. The trouble, of course, was that it was a tube machine, basically past its useful life. And even though MIT didn't pay a whopping price, they chose to charge themselves two million. They never paid it off, I understood. So it was a bad business deal.

NORBERG: Besides allocating research assistantships and managing the organization, what sorts of projects did you get involved in then that would have led to the CTSS project later on? Let me ask the question differently. It will probably be more intelligent, too. As machines like the 704 were brought into the campus and replaced WHIRLWIND 1, WHIRLWIND 2...

CORBATO: No, there was never a two.

NORBERG: Oh, okay, and replaced WHIRLWIND 1, how did you people organize the center to improve its effectiveness, increase the reliability of the machinery, increase use effectiveness in productivity, and so on?

CORBATO: Well, we tried to run a state-of-the-art center. For instance, I used to go to the so-called Share meetings. It was an organization of IBM big machine scientific users. It probably happened twice a year. You would compare notes with peers, both university and industrial. So one had an idea of what it meant to run a state-of-the-art center. In particular, what happened rapidly with that class of machine was that they evolved into a batch processing mode of operation. The initial configuration of the machine was to use a card reader and an on-line printer, on-line punch. The trouble was that the machine was paced by the speed of those. Since there was no other way of getting information in or out -- one normally needed a lot of information -- the result was that the machine worked at a crawl unless you did something about that. So the solution that came from batch processing was that all the input decks would be prerecorded on magnetic tape using an auxiliary machine, which in turn would serve as the input to the 704. It would run through a sequence of jobs and regardless of what happened, whether it would crash or whether they ran to completion, the results would be pumped out to another magnetic tape, again for subsequent off-line printing and punching on it either on another or on that same auxiliary machine. The only trouble with that was that by batching things up it meant that the cycle time for putting in an input and receiving a result was, at a minimum, a half an hour; more often it took a couple hours. Sometimes it even took a day, because being a single queue, the delay time in getting jobs back would slide to a day. That gradually happened more frequently, although we went to great effort to try to work our way out of it by having priority queues, short jobs, and deferring long jobs to the weekend. At some point, the load built up to the point where even the short jobs were getting terrible service. At the same time people were beginning to write more and more intricate programs. As people got more ambitious writing programs, they obviously had more and more trouble not making a mistake somewhere, and the debugging process got to be more and more frustrating. So this was reaching a head. In some ways the universities were the first to notice the problem, because they were the least wealthy, and I recall that places like United Aircraft down in Hartford, which were doing a lot of military defense work, were using five 709s at one time, and we had one shift of one (laugh). So it

was pretty desperate. Well, where do you want me to go next? Should I discuss the kind of projects we were working on?

NORBERG: Yes. Well, when I asked you the question about trying to increase the productivity of the machinery and so on, you defined the problem all right -- that the problem just kept getting worse and worse because of both the intricacies of the programs that were being run, and also the increase in the number of users. Now, you were faced with this situation in about 1958-59, and you could not buy another machine.

CORBATO: That is right. We could not convince IBM to do the upkeep and give us more resources. They tried to be generous in their own way, but they had their own internal fights, I am sure.

NORBERG: Was there any attempt to find resources elsewhere?

CORBATO: The quick answer is no. We were pretty much locked into working with IBM in that center, because we were given the use of the machine, and we in turn gave... We allowed it to be used free of charge, so one of the auxiliary complications was that we could not buy our way out. The users were used to free computing time, and they did not have funds in their research budgets to pay for computing. So if we were to try to get a new machine, even if it was another IBM machine, and suddenly started charging for it, people would not have the money to pay for it. That was a situation that did not get rectified until about 1965. It was one of the things that additionally aggravated the problem. The first genesis of timesharing began when John McCarthy wrote a key memo, which suggested that one could set up a debugging terminal that could interrupt the main computer without any damage to the job in progress, do some computations, and interact with the terminal and at the same time allow the person to run a program to check out debugging problems. John has told me that he went to a UNESCO conference and heard Christopher Strachey say something very similar; however when I looked at the documents that had gotten written up, it is clear that John had a broader vision. He really recognized that you could have lots of terminals. He had a whole framework, a whole computational environment. Strachey, I think, thought of it mostly as a debugging terminal. So again, those are written documents.

NORBERG: Yes, I have read the Strachey piece. I have the McCarthy piece, for that matter.

CORBATO: Yes, now they were a vision.

TAPE 1/SIDE 2

CORBATO: In the early 1950s, there were many universities that had been involved in building computers, but by the mid-1950s, as the commercial world began to build up, the university activities were dying down. The companies, however, took a kind of closed-box view of computing equipment -- they didn't want you messing around with their circuits, or they refused to repair it anymore. So, in fact, to get any changes made to the IBM equipment we had to go through a long, circuitous route submitting, in the jargon of IBM, a "request for price quotation" -- RPQ -- which meant that you wrote down your engineering change idea that you would like. They in turn would send it back to engineering, and depending on whether they wanted to or not, would quote you a price which was either outrageous or modest. It was a combination of IBM arguing within itself between engineering wanting to do it and marketing pushing it or not. So that was the path we had to argue for to get any changes made to the 704. We began to do that, and the local support groups within IBM were quite helpful, but the engineering people were not. They were more preoccupied with trying to build machines that went out to the marketplace. But fortunately, enough of the other aerospace type companies had asked for special changes that we were, by and large, able to find previously done changes which we could ask to have done for us too.

NORBERG: I see. Can I interrupt with two questions then? One of them is, how early did these requests for price quotations begin to be sent to IBM?

CORBATO: Well, somewhere around 1958-59. After the McCarthy memo. Well, we might have asked for one or two on that special...

NORBERG: Yes, that was going to be my second question.

CORBATO: I think we had a special one or two right from the beginning, but the ones focused on timesharing did not begin until after the McCarthy memo. The key ingredients that we foresaw -- John had correctly identified from the beginning. I believe they are in the first memo -- it was necessary, to make things work, to have some way of attaching the typewriters; some way of getting essentially a timer clock into the processor so that one could start the processor off on some job, and still have a way of recapturing the processor before the job finished. In those days the way most computers were organized the job would run to completion and then would return to the supervisor program. So you wanted, essentially, an egg-timer that would grab it back, and one you could set to some fairly tight timing mark -- a tenth of a second -- so that you could run a job for a quick burst, because that was important. If you were going to multiplex between a lot of different jobs, you needed to jump around a lot. Another requirement was that we have some notion of memory bounds registers. Computers were designed, in those days, on the assumption that there would be only one program and that all parts of that program would be equally friendly and well-behaved. There wasn't this "we versus they" attitude that you really ought to have when you build an operating system, particularly the supervisor programs, which had evolved in software, but the hardware people did not recognize a need for that distinction yet. So they would allow user programs to do everything that a supervising program could do. In particular, the user programs could make two mistakes. One, they could trample on somebody else's program that might be in memory at the same time and get out of their bounds, either reading or writing, both sinful -- reading for privacy; writing for destruction. Secondly, they could issue privileged instructions, such as input-output to any device on the machine, which could create chaos. In a lot of the early design of timesharing systems, the easy way out was to say that the only person to issue input-output instructions should be the supervisor. So we had RPQs to try to withdraw I/O control, to have a mode of the machine where the users could not issue I/O instructions, and secondly, could not go beyond its bound registers. Also, the word length was such that there were a lot of unused instructions in the instruction field, the IBM instruction format, and they had left undefined what would happen on all those instructions. A lot of the people had come to start using gimmicky features, which were really not features. You could not predict what would happen. It was an incomplete design, in retrospect a bad design, again, based on the fallacy that people would write perfect programs (laugh). We wanted those also to trap to the instruct supervisor,

so a person could not create unpredictable...

NORBERG: Was it possible back then that the companies, when building such a machine -- say, designing the 704 -- were working under the assumption that they would provide all the software anyway, and that therefore they could control the perfect characteristics of the software and not worry about this problem of poor design that you just mentioned?

CORBATO: That may have been true at some level of the company. I think most of the practicing programmers and field managers and the like already recognized already that they could not control programming. That was an idea that died very fast. It had been an idea that was true in the mid-1950s, but it was amazing how fast it went. In part that was their own doing. By introducing FORTRAN they convinced a lot of people that you did not have to be a professional programmer to write programs. But what they did think was that they knew how to run machines and that they knew best the style of use. One of the reasons that we at MIT were so vociferous about that it could be otherwise was that many of us had cut our teeth on WHIRLWIND. WHIRLWIND was a machine that was like a big personal computer, in some ways, although there was a certain amount of efficiency batching and things. We had displays on them. We had typewriters, and one kind of knew what it meant to interact with a computer, and one still remembered. We could see we were losing that under the guise of efficiency of the machine. But what was obvious was that the efficiency of the people had gone down the tubes. The resistance we ran into was kind of a "Don't tell us how to view our business or how to build machines." In particular, the architects of the equipment were more and more insulated from the use of their machines. They were infatuated with making the machines faster and did not recognize that there was a need for a modal change in the way it was dealt with.

NORBERG: Yes. Were there other centers where this sort of reaction might have taken place as well, people who had access to a machine that they were programming for themselves and therefore, might have felt the same as you people did? I am thinking of the federal labs, for example, before they became federal, of course. I am thinking of Los Alamos and its early computing facilities.

CORBATO: No, they were too well off. People's interest in the problem of timesharing was almost directly correlated with how short of computing equipment they were.

NORBERG: Yes. I remember you saying that before when the ~~Annals~~ meeting took place.

CORBATO: So that it was mostly universities. There were people like Perlis with the Bendix G20 down at Carnegie, and other... Actually in one of my early papers I tried to collect all the names, and my recollection is that they are mostly university.

NORBERG: They are, with a couple of exceptions. I would like to go back to this late 1950 period, and ask you two questions. In passing, you mentioned Lincoln and MITRE as sort of spinoffs from the computing environment here at MIT. What was your association in the late 1950s with people at Lincoln specifically (not so much MITRE; that was a little bit later)? Who did you know there? Who were you interacting with, if anybody? Did this spill over into BBN as well?

CORBATO: Okay, I did not know anybody well. Well, that is not quite true. I knew John -- his last name is blocked in my memory; he was a very clever guy. I knew some of the WHIRLWIND programmers that went out to Lincoln. There were a couple of first-class systems people -- Frank Helvig, John Frankovich, and I also knew Charlie Adams who was not at Lincoln but went out and formed a company. Those people had some influence, I think, on the TX -- not the TX0 as much as the TX2 computer -- and helped carry on some of the exploration in software and hardware. But again, since they did not have a user population breathing down their neck, they were not so preoccupied with creating a timesharing environment as they were with creating in some sense a humongous personal computer (laugh). This included people like Ivan Sutherland, for example, who did the SKETCHPAD using TX2. That was a direct result of being able to have it all to himself, as though it were a big personal computer. I did not have close involvement with him; a lot of our involvement came when we got into the fallout of our decision to reject IBM and work with GE on the Multics project. Then IBM was furiously trying to maintain some identity in the MIT environment, so they managed to convince Lincoln Labs to be one of the initial subscribers to the model 67.

NORBERG: But both of those are later, as we know, and I am still fishing in that early period. The second thing that occurred to me as you were talking about the Computation Center and its growth here was that the context you described was rather isolated, from the description I heard. I am interested...

CORBATO: Isolated from what?

NORBERG: Well, I am interested to know what the interaction was with the administration of the Institute at the time when you were considering where the resources were going to come for new machinery. Or were you just dealing with IBM trying to get the machines? That is, was Morse and other people in charge of the Computation Center doing these interactions with IBM, or was there a whole series of evaluations on computing at the Institute at the time? I am thinking of the McCarthy memo; I am thinking of the Teager committee and so on.

CORBATO: Yes. To a large extent the Institute wanted to see the problem go away. First of all, a lot of young turks (not Morse; he was not a young turk, but McCarthy and others were) were pushing for expensive resources, and they did not see how they were going to pay for them easily. Obviously, as a private institution the usual pattern is to let people try to somehow solicit sponsors who might come up with the equipment that is required. So Morse's ability to keep IBM interested in upgrading the equipment was certainly something that they undoubtedly were pleased with. Every time he was successful it minimized the problem. They also gradually conceded with other groups who would go out and get a small computer for this or for that. (A small computer... Well, PDP-1s were the first of the minis.) With time they eventually began to allow other centers to get established by using other equipment. But basically, those were almost political decisions to keep people happy. The argument was, if you could show how to finance it, you could probably argue your way to getting another computer. I think the Sloan School of Business had at one time a 1620, which is a small IBM machine. The civil engineers had another 1620. There were little pockets of computing around. But that actually leads to the formation of Project MAC itself.

NORBERG: Well, let me just get a couple more things...

CORBATO: But you brought up the memos -- or rather the long-range committee.

NORBERG: Right, I did. Now, can you tell me how that committee got established, or do you not know? Were you not privy to that?

CORBATO: No. Well, I may have fuzzy recollection. Let me see if I can put it back together. It was all kind of an amalgam. McCarthy was pushing hard for MIT to do something. We in turn were pushing on IBM, and IBM was doing a little bit. I guess the Institute responded by creating a committee consisting of Morse, and Al Hill, and, I think, Bob Fano. I am not sure who else was on it. They created a subcommittee with Herb Teager as chairman, and about a dozen other people, myself and John McCarthy included. A copy of that exists, doesn't it?

NORBERG: Yes, I have a copy of that.

CORBATO: The charge was to explore what MIT should do about acquiring a large computing facility. I do not think it said, go get a timesharing machine. In fact, one of the arguments was whether we should try to buy something off the shelf, whether it should be timeshared, whether it... The committee fairly rapidly agreed that it ought to be timeshared. I do not think that there was that much dissent on that point. The second question was whether it should be off the shelf, whether it should be something we built by ourselves, or whether we would act as contractors and put out a request for proposals and see who would bite, and commission somebody to go build it. There were two poles of it. I think Teager wanted to see if we could buy something off the shelf, while McCarthy, on the other hand, was probably the strongest advocate of, "Let's be our own contractor and put it together ourselves and/or get somebody to build it for us." In either case, there was an assumption that we would write the software for it. After a lot of argument and discussion about it, Herb Teager, who was chairman, who kind of played it close to his chest, finally decided that he would go off and write a draft report. He wrote up a draft report, which was overkill from the point of view of documentation -- it was very ponderous, about an inch thick. The bottom line was that he recommended that we ought to go out and get a STRETCH computer, an IBM STRETCH. I missed the session where

this happened, because I was out of the country or something, but the rest of the committee apparently went over the committee report line by line, chapter by chapter. It was the first they had seen of it. They said, "That is all very nice, Herb, but you ought to change this and change that. And, oh, by the way, you ought to change the conclusion." (laugh) Herb apparently kept his temper in the meeting, but he came back from it very angry. The first thing I knew about it was I got a letter in my mailbox that was just steaming, "I won't do it. I give up. I resign." Herb had thought about it and he could not accept that.

NORBERG: Do you know what his reasons were for not wanting to accept the contracting status?

CORBATO: No, I do not know in detail. Herb was very dogmatic. He had strong views, but he was not good... I can imagine reasons, but I am not sure if he had them or not. I think he had some skepticism over our ability to contract something out. That was warranted, I think, because in some sense there was a gross underestimation of the software difficulties. And I do not recall where the funding was supposed to come from for this great project. But the result was that the rest of the committee went off and created the second report, which was a thin one, about a quarter inch thick, which everybody but Herb signed, which was really basically "speccing-out" the kind of computer that ought to be built, regardless of who built it or who worried about it. Remember, the working committee I described -- about a dozen people -- was a subcommittee to the Al Hill, Fano committee. There were four people; I cannot remember now exactly who was on it. They did not know what to do with that, I guess, because basically it was sort of a demand, "Let's get going." Yet, they had no idea how to do anything about that. The result was that MIT basically stood on the group, just sat on it. I do not think that they ever even formally released that proposal, even though we had let copies get around all over the place. So there was real frustration by McCarthy. I think it was one of the things that led McCarthy to decide to move to Stanford; he was pretty angry at the fact that his views were given such short shrift.

NORBERG: Given short shrift by the administration, I assume, not by the computing community.

CORBATO: Well, John was always viewed as a provocative and interesting wild man. He would state things boldly,

and of course when he said, "All you have to do is this," you knew that "All that you had to do," was a loaded statement. But he was right too, so it was a combination. So there we were. We were in that situation where we had a grand plan -- a grand vision, I guess, would be a better description, since it was not so much a plan as a vision. Part of the specs were a million words of memory. I forget how fast the computer was supposed to run, but the notion was that it would be a pretty powerful computer. It was effectively timeshared.

NORBERG: That group finished its study, as I remember, in late 1959, or mid-1959.

CORBATO: I think it has got a date on it of April 1960, but I am not certain; I would have to look that up. That would be approximately right.

NORBERG: Yes.

CORBATO: So that was where we were, and meanwhile, we kept trying to prototype timesharing systems and build some modest ones out of what we could do. Phil Morse got a contract out of the NSF... I think it was NSF; possibly ONR; or possibly both... for the computation center to do timesharing research. Herb Teager was the main leader of that effort. Herb started off trying to build a timesharing system on the 704. This is in lieu of being able to do the grand vision. This was early 1960. There were several flaws in what Herb tried to do (this is all with hindsight, of course). One, he had the vision of trying to do all the software with his little group. He basically wanted to drop all existing software packages -- like no BASIC, no FORTRAN, no nothing. He was going to design his own languages. He was forced to design his own hardware to get typewriters in and out. That was a set of aggravating problems in its own right. To some extent, he could not keep everything in perspective. He was also working on handwriting input. One of the problems was that Herb could not recruit people to work with him, because he tended to hold his plans and visions so close to his vest that he would be a little headstrong for what the resources were. Herb was one of the faculty in the Computation Center, and he was working on his contract with Morse, I think as the PI. I was also in the Computation Center and I began to put together a prototype of timesharing.

TAPE 2/SIDE 1

CORBATO: Well, what happened was that I had started up what was to be a demonstration project where I was going to write a crude version of timesharing, just to dramatize what could be done. I had done this partly out of frustration, because Herb's vision seemed too grandiose to ever come to fruition in a reasonable amount of time. In fact, that turned out to be true. We could not get anything working in a reasonable amount of time. Actually, Herb let me borrow his typewriter equipment, so that was a help, just to get in input and output to his computer. We also recruited one of the programmers that was working for me to help. In fact, there were two key programmers working with me: Marjorie, then Merwin, and later Daggett and then Bob Daley, who was another programmer, worked with me. So the three of us -- initially myself, and then myself and Marjorie, and then myself, Marjorie and Bob -- put together this very simple typewriter interrupt in the main computer. A supervisor program that was restricted to only 5000 words, and we used just magnetic tapes as a storage device for our purposes. We had enough drives in the machine that we could dedicate magnetic tapes to this crude system. We really began programming in earnest in the summer of 1961. I remember the fierce problem we had unraveling the confusion in the IBM hardware over interrupts and faults -- all of the dirty little problems of the hardware not being designed for this purpose. By early fall we were debugging this framework, which is what eventually came to be CTSS. By November 1961 we were able to put on a four terminal demo. To show how crude it was, each terminal had to have a whole dedicated tape drive as its storage. So it was clearly not extrapolatable in that form. But we knew that. Furthermore, we had to wedge out from the main memory 5K words out of 32k. That meant we were restricted to running jobs that needed 27K or less -- and that was a subset of all the jobs, because the memory had been 32K for a while, and many jobs were active by then. So that was mostly to convince the skeptics that it was not an impossible task, and also, to get people to get a feel for interactive computing. It was amazing to me, and it is still amazing, that people could not imagine what the psychological difference would be to have an interactive terminal. You can talk about it on a blackboard until you are blue in the face, and people would say, "Oh, yes, but why do you need that?" You know, we used to try to think of all these analogies, like describing it in terms of the difference between mailing a letter to your mother and getting on the telephone. To this day I can still remember people only realizing when they saw a real demo, say, "Hey, it talks back. Wow! You just type that and you got an answer." Obviously not everyone had to be hit over the head...

NORBERG: No, indeed not.

CORBATO: Right, but it did take some of that. That eventually culminated in the whole Project MAC experience, which is more of that.

NORBERG: In that period when you were working on the interrupt system, was this essentially McCarthy's design?

CORBATO: No, John had nothing to do with CTSS beyond inspiration. It was strictly myself and these two programmers, and then later three or four other programmers who helped flesh out some of the beginning commands and extend the system. We did not claim it was a work of art; we claimed it was a work of feasibility.

NORBERG: Yes. How similar was it to Teager's design?

CORBATO: To my knowledge, not at all, because he was marching to his own tune. Herb wanted other people to work with him but only on his terms, and it was so difficult that people did not try very hard. He also had this vision of too vast and grand a system, where everyone would start over programming from scratch. And I guess I was closer to the people that already were using the machine hard for FORTRAN problems and all that.

NORBERG: I assume that you did not have to have any interaction with IBM for changes in equipment to do this.

CORBATO: Well, that was an ongoing dialog with both Herb and ourselves. Everybody needed that. So that was continuing. And yes, they indeed did do RPQs on both the 709 and the 7090. It was not until we got to the 7090 that they were fairly complete. In particular, they added the timer, the bounds register. I do not remember if we got the input/output trapping cured or not. I think we did. Yes, I am pretty sure we did. This is all in advance of Project MAC's use of the machine, and by the time we got to Project MAC we had a pretty tight system that could stand up to people playing games if they wanted to.

NORBERG: Were you aware of any other groups that were trying to do timesharing at that time? This would be prior to November 1961.

CORBATO: In that first paper describing CTSS -- it was presented in the spring joint conference of 1962 -- there were a lot of ongoing activities of people using interaction in modest ways. What distinguished us from them, I think, was that most of them were not aiming for the vision of a large timeshared machine. They had various levels of special purpose applications in mind. In fact, one of the arguments that we had with IBM, I recall, was that they tried to argue that they had solved the problem of interactive terminal systems by building the SABRE system, which is an airline reservation system for American Airlines. We tried to convince them that it was not a solution because it was not general purpose. So there was a lot of confusion on that point. One of our goals in trying to do it was that we wanted to set the vision of what people were after.

NORBERG: Because as you know, a number of these appear very soon afterwards in 1962 and 1963. I am wondering if they picked up the idea from your group or whether or not there was some sort of general consciousness around that this was a thing that ought to be tried.

CORBATO: Well, I think people were tracking one another. There was a venture at BBN, which BBN arrogantly claimed was first, but it was not. It was in parallel. The commonality there was John McCarthy's vision so I do not think that they borrowed from us directly. We both borrowed from the same source. John was a consultant on that, but I think others did the programming. Jack Dennis built a machine, again inspired by the discussions that we had on the long-range planning committee. He, in turn, did that independently from us. However, he had a different vision. He was trying to envision a machine where each user thought he had the raw hardware at his disposal, and he did not appear to have an operating system at all. So that was a different vision. Out at Rand they built the JOSS machine, which again was a vision of trying to build a machine that was kind of a supercalculator, but it was not a general purpose programming machine, according to my recollection of it. Again, you were not able to inject any arbitrary computer language. You had to use the one they had. You could write programs, but not in any arbitrary

language. So, that is off the top of my head. I certainly never claimed to have invented timesharing, but I tried to claim we were pioneering, because that to me was really a more accurate descriptor. We were putting it together. We knew we were laying down a pattern, which would probably be emulated and imitated. Indeed, that is the way it worked out. You can see traces of that. There were a lot of subsequent systems to ours which began to have the same facade. A lot of this had to do with the model that the user had of what he was using. It carries on even to the personal computers of today -- the notion of sitting down at a terminal and/or keyboard and having a set of commands at your disposal. Now, some of it is semi-obvious, but you could create machines which were different. There was one fairly exotic machine -- the Culler-Fried machine, two different people, which was a very elaborate calculating machine, which they built one or two of. I do not know how many. It had a very intricate modal way of using it. You had to know what level you were at to know what a keystroke meant. I never did understand it. I just knew that that was the problem -- that was the reason why I did not understand it. But I think the technology transfer occurred by people seeing the systems in action, and the programmers moving from one project to the other. So you can trace some of the genealogy by the programmers: who went from where to where. For instance, one of the IBM Cambridge Monitor System, CMS system, which later became VMS (or evolved into it), was influenced by the fact that some of the key programmers who did it had worked with CTSS first, so they carried a lot of the CTSS flavor into what they did. Some of the DEC work, the PDP-6, PDP-10 timesharing systems done out at Digital were influenced by some of the work to be done by ourselves, as individuals, and also by McCarthy as an individual. He was kind of an advocate.

NORBERG: One last question, and then I will get off this topic. What was the reaction of the IBM people who were here in the Applied Science Laboratory?

CORBATO: Well, there was no Applied Science Laboratory when we were doing this. There was this liaison group. Oh, wait a minute; that may not be true. I take that back. No, you are right. There was an Applied Science Lab, I believe. I am thinking of people like Jean Sammet and Nat Rochester. But they were off doing their own thing, and I do not think that they were paying close attention to us. They were supportive, but the key person that we were working with was part of the University liaison group. The person that was leading that was Loren Bullock --

basically a salesman, but a very sophisticated one. He was very supportive. He really knew what we were trying to do and knocked himself out trying to convince people at IBM that we might have something to say to them. There was one other thing I meant to say when we were talking earlier about whether they thought they should do all the programming. One of the things that was going on at the same time (which we did not quite realize) was that IBM was beginning to gel their 360 design. And what IBM focused on was that they had too many machine varieties and too much effort going to too many fragmented things. They convinced themselves that they ought to build one class of machines that would be universal for all their applications. It came to be the 360. They also convinced themselves that it ought to have a 32-bit word, and everything should be a power of 2 practically, primarily with the focus that it would be easy to engineer a wide gamut of processors at different price performance points. They were right on that score, but they got fixated on building an engine to support batch processing. They did not really recognize that they were optimizing things for themselves, but not for the customers. That was really orthogonal to our whole venture, and so by the time the 360 was announced on April 7, 1964, to my recollection, we were stupefied, because it was apparent that we were watching a locomotive speeding down the tracks. Nothing we could say or do would cause them to deviate from what they were doing. They had too much at stake. It was not only that; it was also a very difficult process for them to get it. It was announced about a year earlier than a lot of the technical people thought they would need to do it. They were really in deep trouble, and they were not interested in hearing anybody else's ideas on how to do it otherwise.

NORBERG: Well, are you suggesting to me that even at that time they should have gone in the direction of something like timesharing totally for all their machinery? For all users?

CORBATO: I think they were very slow in the company to recognize the need for genuine system research. They had first class physicists working on devices in their research lab. But they had almost no system people of any note. Or at least, what people they had in their research lab worrying about software issues were worried about the fringes of the problem, not the central part of it. They had a major success with FORTRAN, and they had forgotten about that, so to speak. In some sense, all of the detailed hardware design was being done in places like Poughkeepsie, which was really the development and production center. The experimentation and exploration of

different kinds of designs was not done in research. They did not recognize that it was an open question; they thought it was a closed question of how to build machines. Now we have seen this in another industry, automobiles where people... So that was what we were fighting, and we were having a very hard time getting people's attention.

NORBERG: Okay, who's the "we"?

CORBATO: Myself and my cohorts who were kind of locked on. We had bought the idea of timesharing. We really believed that it was a better way to operate. I suppose if somebody had said, "I will give you a free machine," we might have said, "We do not need timesharing." But it wasn't credible, because machines in those days were huge things. They did not quite take up a football field, but they took up a large room and they required air conditioning. They required maintenance because there were so many parts. They were just not a casual thing. You did not normally think of having a personal machine in those days -- exclusive use maybe, but not a personal one. So we really saw a need to try to change that. We were frustrated.

NORBERG: Let me turn to another topic, which will probably recall some other things. When do you remember first meeting Licklider?

CORBATO: Let's see, I am trying to remember whether I knew him at BBN. I did become aware of him at BBN, because John pointed out to me this keen paper on man-machine symbiosis that he had written. So I was aware of him and I guess I was aware of his enthusiasm for the PDP-1. However, I did not interact much with him in those days. The place I remember him was when he went down to Washington to try what I came to realize was an attempt to help the DOD do timesharing, or to improve upon their man/machine interaction. He saw an opportunity, I guess.

NORBERG: But you do not know any of the details of that.

CORBATO: I do not know the details, no. I just knew he went down there. The next thing we knew is he came up here and asked to have a meeting with key people about whether MIT was interested in being a center of exploring

man/machine interaction. I recall a meeting where there were a bunch of us --I forget exactly who was there, but I think McCarthy was there; (no, was McCarthy still in the act? Because he announced he was going to Stanford at some point.) Teager was there; certainly I was there; Doug Ross was there; I think Fano was there. Now at this meeting we began to argue furiously in front of Licklider, arguing that, "Oh, you don't want to do it that way." We sounded like a pack of dogs going in all directions. It was really a disgraceful... My understanding was that Fano came out of that meeting really disgusted. Finally he realized that we needed some leadership, that unless we did better than that, we were not going to be able to get any support for these ideas. My recollection was that it was as a result of that meeting that Fano began to organize a group to nominally explore the feasibility of getting support from Licklider, and out of that gradually came a group. We developed the plan of what came to be called Project MAC.

NORBERG: Yes. Now, if there was this vision that had been developed through the earlier interaction with the Teager committee, why not just trot that out and try to sell it at this point?

CORBATO: Well, first of all, the groups had no organizational loyalty to one another. They were representatives from all over the place. Doug Ross worked for the Electronic Systems Lab. I worked for the Computation Center. McCarthy was in EECS (it would be EE in those days) and also the Computation Center. Minsky, in those days, was in the Math Department. Well, he was in RLE and also, somewhat, in the Computation Center. So there were a lot of diverse people who were interested parties, but we were not administratively united at all. I think the thing that made it difficult was that if we were to have suddenly tried to form together, 1) there would have had to have been a strong charter to make it worthwhile and; 2) it would have caused disruption in every one of those organizations. That in fact happened, in part, when Project MAC formed. The people that became key members of Project MAC, in effect, withdrew out of their other organizations, or split their allegiance. So that was one of the more difficult aspects of forming it. But to a first order, what happened was that the same kind of people that had contributed to the report, the leaders of computational effort, were the same ones that were recruited to be part of Project MAC. There are two reasons why the Project got off to a rolling start. It did not just suddenly start from scratch. One was that the computing platform had really been evolving for several years. CTSS of 1963 was really a considerably polished up version of the prototype we had been demonstrating. Secondly, the people had been active in computing for several

years in little subgroups all over the place. So they came together and were ready to go to work. So it was deceptive to the outside world. In fact, Carl Overhage tried to start another project on campus to do library automation work, and he misread what had happened. He thought he had to come, have a sponsor, pick an interesting problem, and recruit people to come together. However, it did not work very well. One, there was no hardware platform to work on, and two, there were no people to draw upon who were already doing that.

NORBERG: Did that take place before Project MAC?

CORBATO: No, that was after.

NORBERG: Even after?

CORBATO: Yes, that was another project. It was independent, but it did not fly very well because, in fact, it did not have the fruitful infrastructure.

NORBERG: There was a library project associated with MAC, was there not?

CORBATO: No. They used the MAC equipment, and it was not a total failure, but it was not a roaring success either. In fact, I think Frank Reintges of the Electronic System Lab was recruited to keep it from being a failure. He basically did a lot of substantial work exploring some of the hardware implications. But it was a tough uphill battle. It was not easy.

NORBERG: You remember the interaction among your colleagues rather well. What about Licklider's reaction to all of this? Do you recall that?

CORBATO: Well fortunately, Lick was not a typical bureaucrat. So I do not think he was as thrown off as we were, or I was.

TAPE 2/SIDE 2

CORBATO: Well, first of all, Licklider had been an academic before, so he knew how academics sometimes argue without worrying about form. I think he probably also was aware of the rich legacy of ideas we had. Of course, that would not have mattered unless we could coordinate and pull ourselves together. What it was is in part is that, most of the people that were active in computing in those days were relatively young. There were no elder statesmen. People like Morse were kind of locked into the Computation Center. He also did not fully appreciate timesharing. He thought it was just another thing. At least initially, he did not see it as revolutionary. The appreciation of timesharing was directly proportional to the amount one personally programmed. The more you personally programmed, the more you knew what it meant.

NORBERG: What did the onset of Project MAC mean to the computing community at MIT?

CORBATO: It was a major milestone. It served as a focal point for all the computer science activity at MIT. The initial nurturing of computer science at MIT was mostly in the EE Department, with some numerical analysis work in the Math Department. The rest of the activity had been mostly applications work, which had sprung up in the other departments as people picked up interesting applications and of necessity became knowledgeable about using computers. So that was the kind of environment that MAC operated in. MAC became a coalescence of the computer science community. Remember, computer science was just coming of age in the sense that Stanford had just begun a department. We, in fact, had been a de facto department. CS was imbedded in EE in a de facto sense. So here was the first research lab organized around computer science that was of any magnitude. In fact, it became a progression. MIT established an undergraduate curriculum in computer science in the late 1960s, and by 1975 we were ready to change the name of the department to EE and CS. So there was a gradual recognition of the maturation of each. So from the beginning, although we had this great starting theme, I think most of us felt that Project MAC was more than just a project. It was really the beginning of the computer science community, because the theme seemed such a deep one that it would last a long time.

NORBERG: How did your position change, as a result of MAC?

CORBATO: Well, a lot of things happened. Let me back up slightly and say, I was a post-doc at the Computation Center. It was on the basis of my accomplishments in getting the prototype CTSS that I felt I was warranted in applying for a faculty position. So I became faculty in 1962, primarily on the basis of that. Well, I had a history of some applied physics work, but I felt that that was my real cachet. Then I was also eventually made deputy associate, then deputy director to Morse of the Computation Center. I think that Morse was unhappy that I had this split allegiance since now I was a key player on the Project MAC enterprise and supposedly running the store for him at the Computation Center. It was actually feasible, because there were some pretty good staff people there that I could depend on. But he was two layers removed. That came unglued when we started looking for the next equipment to build the next timesharing system. To make a long story short, it eventually came to be called Multics and we eventually picked General Electric Corporation as a vendor to supply the equipment. This was after a procurement process which went on for about six to nine months. In the midst of that IBM began to recognize they might not be the obvious choice. Before that they had assumed they had us in the bag. They went to great effort to try and persuade us to continue to use their equipment, but all their efforts were based on coming up with very special purpose ad hoc solutions, which were either incredibly expensive or incredibly inelegant, so we rejected them. None of them were based on standing on their heads and changing the 360 line. (laugh) So when we finally broke the news that we had picked GE, IBM in retrospect overreacted. But react they did. The consequence was that I dropped out of being deputy director. I forget whether I was actually let go or whether... I think what formally happened was that the Computation Center was repackaged into a new organization Phil Morse lost out too, because Gordon Brown, who was then the dean of engineering, took over. I think it was called the Information Processing Center. He used that as an excuse to get IBM to give MIT more equipment, an IBM 360 model 65, and also the promise of a model 67. He also used that as an excuse to start introducing, with my encouragement actually, the notion of charging for equipment, so that we could get on our own feet, and contracts would have charges for computing costs, and MIT would eventually work its way out of the hole of always being on the dole. Oh, some good things came out of it. But to some extent it was set up that IBM was helping because they wanted to see a

countercourse to Project MAC on campus. I became known as one of the people that had let IBM down -- it made it so individuals were still friendly but wary. The upper management at IBM was still angry for many years. They were also vindictive with their own people. They are very punitive when somebody does not come through for them. The liaison guy, Loren Bullock, who worked so hard to make things work out, was basically given a crummy assignment somewhere. The upshot is that I think he caught it fiscally too. They were very harsh, because the liaison group worked out of marketing and marketing played hardball. So it was tough, but we felt we had to do what we did. I said earlier that IBM overreacted because they assumed that just because GE had a lot of money they could do as much as IBM could. Where they misjudged is that GE was really a financial confederacy rather than a big corporation. They did not have the same technical dedication to computers.

NORBERG: Let's come back to Multics at another time. I want to still stick to this period of the origins of Project MAC. You began discussing, in answer to my question about what your changing role was as a result of MAC, your removal out of the Computation Center and becoming a faculty member and so on. Now, let's get inside Project MAC. How did your attitudes, your approach, your activities change as a result of MAC? Then I want to ask you what your interaction with the ARPA people was.

CORBATO: Well, to a first order, some of what I was doing in the Computation Center, where I had been responsible for keeping a first class service going, mapped over. I mean, here I was responsible for keeping the flakey prototype a sufficient platform that people could use. That was really difficult at first, because not only was it a pretty green system, but people were also pushing it hard and there was a lot of frustration -- typewriters were slow, the manuals were not terrific. I was used to it though. Some of the more mature faculty recognized that, in fact, the fact that they got frustrated was telling them something. People like Weizenbaum, for example, spotted that we obviously did not have everything just right, if we still got frustrated. One thing it was telling him was that it was very important to him that he have a computer, that if he cared that much, it must be important. So there was a lot of wisdom around too. I think the key thing it did for me is it pulled me together so I interacted a lot more with people whom I had formerly known less intimately. But to a first approximation I had already been in such a focal point and position at the Computation Center. I was well known around the Institute by virtue of my administrative role in the Computation

Center. Memos and notices would come out with my name, so people knew me that I did not know very well. So I still was in that kind of a role. The key change that it made for me was that I got on this kind of fast-moving treadmill of going to the next timesharing system, the Multics thing, and that turned out to be a major commitment, more than we realized at first. But I was the one that was primed and ready to play the key role, because I had been a builder of CTSS. I had the staff, people, and everything was all set to go.

NORBERG: What was the effect of the summer study?

CORBATO: That was a very exciting period. Everything came together just about right. I think just before the turn of the year in 1962, the order was placed to get a copy of the Computation Center machine, so we could run software instantly. But it was not going to be delivered until about November. Meanwhile, in the spring we were furiously getting the system ready for the summer. We planned to borrow time off of the Computation Center machine for that summer study.

NORBERG: Without charge?

CORBATO: Without charge. (laugh) I guess we looked the other way. There were a lot of ambiguities in this situation. For example, one ambiguity in the use of that machine was that, if you think about it, there is the nominal first, second and third shifts, and then there is really a fourth shift; namely, the weekend. So there was some confusion about who got to use that. So between the November 1961 demo and the summer study we had to make several key upgrades to make the equipment viable. One of them was that we attached a 1302 disk file. It was a large, random access disk file -- large enough, large by its day -- where we could keep all the programs and data in storage. The second thing we had to do was we had to get typewriters basically connected in quantity. They turned out to be Teletypes or modified IBM 2741s. Actually, they were IBM 1050s initially. It seemed like nothing was right in those days. But we ended up getting a huge kludge of a controller called a 7750, which attached to the IBM 7090, and which basically allowed one to get up to 32 devices attached at once. So we could run with a nominal load of, I think, 16 active terminals at one time. We also got a telephone switch installed so that people could dial from many sites,

with the use of modems, so that in fact eventually there got to be several hundred terminals around campus. So I think initially we were hardwired but... So all of that was ready and there must have been one or two hundred people that came by. Bob Fano and Oliver Selfridge, who was an assistant director (maybe he was associate; I am not sure. Fano can tell you), and Dick Mills, who was brought in as an administrator and was an assistant director, were the ones that managed Project MAC, and they helped plan that summer. They drew up a list of invitees -- probably one or two hundred long. People came for as little as a week, sometimes longer. Some of those people were recruited to come to MIT and join the project. But we had two motives: 1) to attract high quality staff and; 2) to disseminate the idea that we were trying to propagate about trying the change in the computer industry -- to influence the vendors and manufacturers to do things differently.

NORBERG: Now, this was a conscious objective at the time?

CORBATO: I thought so. We really were frustrated. We could not get IBM's attention. We wanted to convince them that there was a different mode to run machines with, and that this kind of batch processing optimization was all wrong for people. Yes, we were trying very hard to change things, and it came out in lots of ways. For instance, we wanted a typewriter input/output but not line printing. We wanted upper and lower case, so that you could get a document out that looked like a document, not a telegram. You know, just... all the nuances that you can imagine. We had discussions and arguments and discussion sessions. I remember leading one or two about how things ought to go, or should go. We were kind of ahead of the game, because we thought about it longer. We also were the leaders. But some of the people that came from afar, I think, had things to add too. I mean, it was not entirely a one-way street.

NORBERG: Does that suggest that the summer study only enhanced the view here rather than changed it?

CORBATO: Well, I would say it fleshed it out -- enhances is a reasonable word, but I do not want to sound arrogant, because I do not think that we felt that we had all the answers. We had just been thinking about it earlier and faster. Some people came with new ideas. I mean, Engelbart already had the genesis of some of the things that eventually

led to his work out at SRI. But I think we got a chance to see how people would react to a change of image. Some people were still preoccupied with assembly language programming, and we thought that was a little too detailed. Some people were still locked into worrying about issues of editors and things. So there were arguments about which was the right kind of editor. The technology was not advanced enough to have display terminals. That would have opened up the door somewhat. I think that by and large they augmented our views. I do not think that in any instances we were dramatically shifted from what we doing.

NORBERG: That is my feeling from looking at the memos and looking at what happened after the summer study. I looked at your 8 July 1963 memo on problem areas of timeshared systems, in which you listed 13 different problem areas, which in a sense are highly detailed, for example when you start talking about problems with scheduling algorithms. This is highly detailed, and for people to come in and be faced with this sort of a detailed list and want to change your vision is rather difficult.

CORBATO: Sure.

NORBERG: One last question. This can go on for a long time, and I want to be parsimonious with your time. One gets very tired under these circumstances. Do you recall Licklider's presence at the summer study?

CORBATO: I cannot nail down a precise episode, but I think that from the beginning he viewed us as one of his major grants that he had been able to make, and he gave us the impression that he thought very highly of what we were doing. He was always a great supporter. So we felt he was really a true patron in the sense that he really made us feel good and seemed very pleased that we were doing what we doing.

NORBERG: Do you remember a memo that he wrote in April of 1963 in which he outlined what he would like to see from Project MAC?

CORBATO: No. I do not remember.

NORBERG: It is one of the MAC memos. I should have copied it today but I did not. It is a very interesting vision of what a timeshared environment ought to be like. What I am going to try to do is compare that with documents from MIT in the same period with his to see whether or not he drew it from the MIT experience, or whether there was some sort of double track here to be aware of. But he was definitely guiding things in one sense, and I am hoping to ask Fano how that guidance took place.

CORBATO: At the time I felt that Lick was totally supportive. I was amazed at how wonderful it was to have a sponsor who believed in what you were doing. Subsequently, I think I have discovered that Lick did not always understand what we were doing, and probably would not have agreed with every little detail, but he was enough of a visionary that he recognized that we were moving in the right direction. It was actually a very benevolent way of managing things. I also felt that Lick had large confidence in Fano and vice versa. I mean, the two had a relatively good relationship. That was rather critical to making it work too. They trusted one another.

NORBERG: Do you know whether that relationship preceded the original negotiations for Project MAC?

CORBATO: Well, they knew each other, but I do not know how closely they knew each other. In fact, I do not think they were that close. In fact, things got rather strained in later years, because Lick has some wonderful strengths, but he also has some weaknesses.

NORBERG: Well, we all do. One last question: do you remember anytime being asked to be a consultant to review other projects that were being sponsored by ARPA?

CORBATO: The closest situations I can recall are two episodes: 1) I think in reaction to criticism from somebody there was a comment -- and I don't recall when this happened; this was probably in mid-1967 maybe -- that we did not know what other people were doing. So I was asked to make the grand tour and visit half a dozen places and go and look and see what they were doing -- go find out; just show that we cared. I did that, and I learned from it. It was a

little bit of a chore, but it was also useful. I remember going to the 940 system at Berkeley. I do not remember if I went to Stanford, but I certainly hit SDC, and Rand, and so forth. The other thing was that Lick called a meeting at SDC to get them going, and I was part of the group that went out there to basically catapult them into developing a timesharing system. I got the impression that they developed it because Lick almost ordered them to. But once they decided to do it, they went at it and did a decent job. Jules Schwartz was the leader of that, and he was quite credible. I do not think they originated it, but once they looked around and saw it they did that fairly fast. They had in-house programmers and all that. So I think I was part of the inspirational group that Lick called together. I do not think I ever was asked to criticize anything.

DATE: 14 November 1990

TAPE 3/SIDE 1

NORBERG: Today is November 14, 1990. I am in the offices of Fernando Corbato for session #2 of our interview on the development of time-sharing - this time especially on MULTICS. I would like to begin by referring to the opening of a publication by Glaser, Couleur and Oliver in 1965 which appeared in the fall Joint Computer Conference of 1965. The article starts out "In the late spring and early summer of 1964 it became obvious that greater facility in the computing system was required if time-sharing techniques were moved from the state of an interesting pilot experiment into that of a useful prototype for remote access computer systems." What considerations led to the design of the MULTICS system? Do they emerge out of this kind of statement that is made here?

CORBATO: Well, that's a little pompous, but... [laugh]

NORBERG: [laugh] The statement, or my question?

CORBATO: No, the statement; not your question. No, it's a good question. It really goes back to the genesis of Project MAC. I think Fano, in his own vision of the project, saw it as getting off the ground with some sort of a

demonstration, but he saw it as a kind of a keystone of the evolution to the development of a commercial quality product with the obvious intent not of going into business, but of influencing the vendors to do this, to make that the norm rather than the exception. So right from the beginning there was the vision that we would do it right. It would become a major effort within the project. Fano's view, from a managerial point of view, was that this would serve as a unifying venture, too, because there were many roles where all the different players could contribute. Now, because I had been the leader of the development of CTSS, that was first a natural prototype, which was the demonstration. It obviously was not very extendable in its initial form. It had both strengths and weaknesses and it dramatized what ought to be done to make it more useful. So MULTICS started out as kind of a wish list of what we would like to see in a big computer system that might be made as a commercial model. That was the vision. It started out by our trying to encourage vendors to join with us. So we spent about three or four months - maybe it was six - touring around, talking to major vendors to see if they had any enthusiasm for this vision. The responses varied. Some companies viewed it as an opportunity to bid in a military defense contract - a way to make money on the margin rather than affecting any product plans. That was my view of Burroughs. CDC was disinterested. That is my recollection, although I would have to check notes to confirm that. Again, Seymour Cray didn't believe in time-sharing. He was preoccupied with his own directions, and he dominated the company at that time. Some of the junior CDC people, who were chafing under the dominance of Cray were a little more interested. DEC didn't think they were big enough to take on something. They saw themselves as still too small a company, which in that day may have been true. IBM treated us as... patronized us, I guess, by fiddling with products but not seriously, not giving us any say in their major product plans, which at that time were unbeknownst to us, namely the 360. I should say initially unbeknownst to us. We were eventually allowed to see the machine with confidentiality, non-disclosure agreements. We were somewhat horrified but they were too set in this juggernaut they had launched. They were not able to change it, or unwilling to change it, and to some extent didn't believe it mattered that much anyway - what we were trying to do. Then finally we stumbled into the General Electric Company, who were just launching a new computing system called the 635. They were hungry to try to conquer new domains and willing to be flexible. They were adventurous and naive, and we were naive and so forth. We all thought we could do more than we could. But in any case, that allowed us to proceed. They were willing to talk about deep hardware changes, which were initially thought of as kind of a modification of the 635 processor, but which subsequently turned out to be so massive that it

was a different computer, called the 645. John Couleur and Ted Glaser both were instrumental in doing the major changes of the processor units. The GE design was very attractive because they had thought about it from an asynchronous module point of view. So it had a lot of the same generality that we were looking for. We were highly influenced by some of the ideas of Jack Dennis about how to organize address spaces and, in particular, segmentation. So that turned out to be a very successful honeymoon.

NORBERG: Can I ask you to stop there and ask you a few questions about it.

CORBATO: Sure.

NORBERG: When did these series of visits occur?

CORBATO: I would have to check notes...

NORBERG: The year is fine...

CORBATO: Even to get the year right. [laugh] Let me pinpoint it by saying what was before and what was after. This was after our... There was a summer study, which I am always getting the year wrong on.

NORBERG: 1963.

CORBATO: 1963? Okay. It was after 1963, and we didn't make a decision until after the announcement of the IBM 360, which was April 7, 1964, because, well, I have to come back to that. It's getting a little tangled up. And it was before the Spring Joint '65, when we wrote up these six papers as a declaration of intent.

NORBERG: Spring Joint or Fall Joint?

CORBATO: I could have that one wrong.

NORBERG: Fall Joint Computer Conference '65.

CORBATO: Sorry, Fall Joint. Yes, that makes sense. I didn't prep myself on all of this.

NORBERG: It's all right.

CORBATO: So it's in that interim period - fairly soon after the summer study we began to explore with the vendors and it took at least six months traveling around.

NORBERG: So somewhere, say, between September of 1963 and March of 1964? That was before the announcement by IBM.

CORBATO: Yes. When IBM announced their machine I don't believe we had made up our mind yet, but we had prior knowledge before the announcement. IBM hadn't taken us very seriously until they got wind of the fact that we were working with GE. They had a violent reaction, because from the high levels of IBM it looked like an awesome competitor. Here was a company with deep financial pockets, we were suddenly dealing with somebody who could bankroll us enough to do something. What IBM failed to understand was that GE was organized as a consortium of shopkeepers. Each company was a separate cost center, and the financial deep pockets weren't that deep at all. They required immediate success or people cut them off. So the ability of GE to act with the determination of a large company wasn't there at all. It was a bunch of bean counters. And the GE computer division was a relatively young and... Well, it wasn't so young, but it was...

NORBERG: Still a fledgling at that time, I believe.

CORBATO: Fledgling from being in the computer business, and the management was not that wise in the ways of

computers. The upper management didn't understand software very well, and they knew how to get hardware built. But to some extent the software and hardware people were curious blends of great expertise and vision and naivete. So it was funny mix of people. One of the reasons they went with this was that the few key people were prepared to gamble with us. On the other hand, that was one of their very flaws; they didn't have the checks and balances inside to keep them from doing strange things.

NORBERG: Okay, we know who you dealt with at GE because of names on publications, although I didn't see Weizenbaum's name on there. I am wondering if he had anything to do with it.

CORBATO: Weizenbaum was kind of a finder in the sense that he had worked for the bank... I forget whether he worked in the Bank of America or GE itself directly, but he had been working with GE and knew a lot of the players and was aware of their engineering potential. So he's the one that, I think, told Fano, "You really ought to look at GE; they seem to have hardware that matches the goals of what you have set out better than some of the other companies." So that's how we got started. He led us to look up GE. Once we did he had no further involvement.

NORBERG: All right. But turning to the other companies that you visited, do you remember, other than Seymour Cray at CDC, who you talked to at CDC, who you talked to at Burroughs, and so on. I want to come back to Burroughs for a moment.

CORBATO: At CDC we went out and gave a site visit at least on one occasion. McCarthy gave a pitch for time-sharing, trying to get people to recognize the vision of what it could mean to have a machine organized in this particular way.

NORBERG: Is this after he went to Stanford or before?

CORBATO: He was still with us, so it was before. Seymour dropped into the back of the room. First of all, he was too busy to see us (so he said). That was his standard ploy. And he dropped into the back of the room and heard a

little bit of John's wilder statements, which were meant to have shock appeal, to get the audience's attention. He heard a few of those and walked out. I had to assume that he had made up his mind, as I reconstructed after the fact rather than at the time. So he had... we had similar... not quite so extreme, but we had similar attitudes of most hardware designers, who were all interested in doing their own thing. They really didn't want to be bothered with somebody introducing new ideas, and they all had their own vision of what a computer ought to be. And it didn't include the social environment of people trying to use the computer all at once.

The situation of Burroughs was that the people we talked to... we weren't talking to the commercial side of Burroughs; we were talking to the military side. So we got what we asked for to some extent. This was down at Paoli, Pennsylvania. One of the people we talked to was Ted Glaser, I think we talked to Ted at that time. In any case, Ted came up to the summer study, and he personally was sufficiently turned on by the ideas, that he saw as an opportunity to contribute, and he joined Project MAC. So he left Burroughs and joined us. But the company itself was in kind of a military cash and carry business. They would build computers if the military thought they needed them. There wasn't any great... Well, there were some ideas, but it was complicated, and disjoint from the commercial side of Burroughs, although there were a few people, like Bob Barton, who would have an influence on both sides, inspirational influence, I don't know how much technical influence they had.

NORBERG: What I am looking for is the possible transition from this interaction between the MIT people and Burroughs people and the later involvement of Burroughs personnel in the ILLIAC IV project, which was funded by IPTO, whether there was any crossover there, because that was also done at Paoli and began around the same time - in this case, 1965, so it's a little later.

CORBATO: I don't know if there is any crossover. I hadn't thought of that. In many ways the pattern was similar. Burroughs hired in as a vendor rather than as a person who had their reputation on the line in case it failed. So we obviously never consummated it with Burroughs, so it never... perhaps I am projecting... but our perception was that if we had gone ahead with Burroughs they weren't a good partner. The ILLIAC IV or, excuse me, was that ILLIAC IV?

NORBERG: Yes.

CORBATO: There were earlier ILLIACs. The ILLIAC IV was flawed from the beginning because they didn't have a very good plan of how they would make it be useful, even if it had worked from an electrical point of view. And then they had a lot of problems, including a fire and so forth. But to my knowledge there was very little interplay between those different projects, although there might have been within Burroughs.

NORBERG: All right. Can I try to repeat what I think you just told me in 20 minutes. There are two things I would say to characterize the description you just gave. One of them is that the development of a concept called MULTICS and the drive to produce a new system of this kind... a pilot system of this kind started essentially with the beginning of Project MAC - that is, you didn't spend a period of time bringing a CTSS system into operation for Project MAC and then go on to MULTICS. There was a sort of a simultaneity there, at least somewhat.

CORBATO: Oh, we did both. Clearly, because CTSS was an immediate object of use, we spent a certain amount of effort for the first year getting it to be more effective and more useful to the participants in the project. But from the beginning we had the vision that we were not going to stop on that system, that we were going to go on to something else. So it was really both. And one of the management problems we had was forcing ourselves off of CTSS and kind of freezing its development, because there was a temptation to keep going, to keep fiddling with it, and changing and improving it.

NORBERG: Who participated in the evaluation of CTSS to define the objectives of MULTICS besides yourself here at MIT?

CORBATO: Well, I think it was people like Glaser, myself, Jack Dennis. From my recollection it included Bob Graham and it probably included Jerry Saltzer, although it may not have; he was a graduate student at the time. It included Fano. I would have to look up notes. We took input from lots of people, but the people that... Glaser and I were the

ones defining the objectives from the MIT side. I think to properly answer that I have to go back and set the stage a little bit.

NORBERG: All right, please do.

CORBATO: Well, first of all, there was a loose end on the hardware before we made our decision to go with GE. The loose end was what IBM's reaction was. We made our decision. We tried to tell the MIT administration that IBM was not going to be pleased. They did not understand that very well, so we had a lot of internal problems. People didn't handle IBM very well. IBM, in turn, overreacted. Their immediate reaction was, "We have got to save this account." That was one of the problems; they kept thinking of us an account rather than as a co-participant in a new product. That reflected the marketing viewpoint of the company. As a result they ordered all the engineers to do something to make MIT happy. What they did was build a Rube Goldberg-like offering, which they then priced accordingly - namely, pretty high. What they were shoving at us had no future as a commercial product. It was done in deep (?). It was just patch quilt work using the existing 360 design. We rejected it, because we saw it hopelessly flawed, starting with some deep design decisions that had been made inside the 360, which, from their point of view, would have been very, very hard to change. The only way to satisfy us would have been to have embarked on a new product design which was different from the 360. At that time they were obsessed with making all of their computing customers fit in the 360 mode. They had been misled by their senior designers to believe that the 360 was good for everything. That just wasn't true, but the company wanted to believe it. So we got into this impasse, but we rejected their kluge. It was a period of some hard feelings. But subsequently, after we had begun to work with GE, they reacted violently, because they started a whole new product line based on the 360 with some deep, deep engineering changes (this became the model 67). They rushed to create a competitive product, because they thought we would create a product. This is covered in at least some of the publications I have. I forget exactly which ones, but basically they underestimated our ability to produce. They got it out the door faster by pouring ten times the resources, but it was very, very lame in performance. And they basically, after a period of a year or two, withdrew it as a product. By this time it was obvious that we were not a competitor. So a saga in their...

NORBERG: This is the model 67 that you are talking about.

CORBATO: Yes, with their own time-sharing system. They ultimately salvaged their commercial interests by using the system that later came to be called VM - virtual memory. But I am digressing a bit here. So that's how IBM reacted. The way we picked up partners was also important because it set the tone of how MULTICS developed. We started by our deciding to work with GE. Simultaneously, Bell Labs had been looking for a new computer acquisition for their laboratories, and they had been scouting out GE. There was some synergism: because they knew we were interested they got interested. I think they first began to look independently of us. But they saw the possibility of our developing a new operating system together. So GE was looking for new markets. We were looking to create a new vision of the future. And Bell Labs had their own form of naivete; we had ours. Where they thought that you just got a bunch of good system programmers together, and they build an operating system on top of the hardware, and pretty soon it was useful for everybody - good things just kept happening. My own view of this was that this was based on management's experience, the way they dealt with the 7090s, where system programmers had done some innovative changes and modifications and improvements to the existing operating systems on the 7090s. What people failed to understand, I think, was that nobody at Bell Labs had built a system from scratch. They had always been modifying and improving things that were already working. And that misled the management into thinking they had deep system expertise in the manufacturing of systems, and they kind of assumed that it would happen again. And so they joined together with us thinking that somehow out of this new hardware and software would come great, wonderful systems, which in turn could replace all the 7090s in the Bell Laboratory system. And these multiple objectives of the three different parties eventually became part of the problem, because they were all based on slightly naive views of what could happen. One of the most pivotal weaknesses, though, in this framework was that nobody was in charge. There was an agreement to cooperate. Although there was a contract for hardware and probably for the software activities between GE and MIT, and GE and Bell. There was no contract between MIT and Bell. So everything was based on cooperation and goodwill.

TAPE 3/SIDE 2

NORBERG: Let me ask you about my second point, because I think it's easy to answer with a yes or no. I hear you telling me that the view at MIT was that you people were involved in it in experimentation, not in the development of a system that would be of sort of a finished product, let us say; rather you were relying on the manufacturers to create this finished product, which would then become available to whoever else wished to buy it. Is that a fair statement?

CORBATO: Almost. We weren't that experimental. We wanted something that was usable, but we didn't expect to put the commercial levels of polish on the system. By that I mean we didn't expect to have to do the manual writing; we didn't expect to have to do the marketing; we didn't expect to have to develop the maintenance groups, or whatever. We were doing the prototyping, a very strong prototyping, that was our own view. We didn't see we had a choice. What caused things to go sour from the point of view of initial expectations was that in our tinkering with the hardware we had made such deep seated changes that none of the software that had existed in prior form was usable.

NORBERG: This is software from CTSS, say?

CORBATO: No, certainly not from CTSS; that was a different machine. But none of the software from the 635 was usable, so we didn't even have an assembler; we didn't have a FORTRAN compiler; we didn't have anything that worked directly.

NORBERG: Why did you think it was going to work in the first place?

CORBATO: No, no, no. Let's back off, okay. [laughter] Suppose some company decided that they were going to put their operating system in French or something, and somebody on high said, "Well, let's just do that little change, and change from English to French." You would really have to start over, trying to find all the places you were dependent on English in the operating system. The changes are very deep, and they require... even if you maintain the... Now, we made some deeper changes than that. We didn't just change the character sets, which would be

comparable to changing the language. But we also changed the addressing structure, so that we were talking about segmentation. This introduced a new dimension and a new concept into the deepest part of the software. Now, that was part of our kind of ambitious technical goals. That meant that, instead of having a modification type effort going on, we were starting over, like building a house without any... not even foundations. We couldn't adapt to previous design. We had to start over and redesign everything.

NORBERG: Okay, and you knew you had to do that.

CORBATO: Well, we backed into it. We knew we wanted a machine built with a fresh slate, but I don't think we initially realized that the depth of our ideas was scuttling all of the auxiliary software that might have helped. Now, I guess we also thought, well, it isn't that hard to build software. The trouble was that if you have everything to do at once, you are caught, because you only have so much time in the day to think about it. And although we tried to partition the problem by saying, "Well, we will get a vendor to build a PL-1 compiler," we also agreed to use a brand new language which had never been road-tested before. It turned out the vendors were flaky, and messed up badly, and the language was a bear to try to harness, so that failed too. Our problem was the aggregate number of problems we ran into bogged us down and put a crucial several-year delay into the project.

NORBERG: Is this what was referred to in the 1972 paper that you did, which said that the software was an iterative process, that it was necessary to keep developing as you went along, both in terms of developing new lines of code, but also in getting out the things that you didn't need, because they introduced complexities which then caused the machine to fail on occasion.

CORBATO: Well, in trying to orchestrate all these ideas together we underestimated how cleanly they would go together. I said that wrong. We failed to recognize that they would not go together, and so we were top-heavy with too many ideas. Sometimes it was a case where people didn't see an elegant way to do it at first, but more often it was a case where you had to shed ideas that weren't so critical or important and shouldn't have been there in the first place, if you had the wisdom to realize that they were going to bog things down. So there was a pruning down to the

more bare essentials and trying to find a more elegant way of doing things. That was a lot of the rewrite process. So, where does that leave me? I forget your question at this point.

NORBERG: Yes, I asked you if that was what you were trying to say in the 1972 article by talking about iteration - that is, trying to build new software as you went along, realizing that changes needed to be made, because it either wasn't working quite right or because of what you were just describing in terms of having to start all over again, and, therefore, finding that there were problems along the way. That's not said very well, but let's leave that aside, because I think we can get into this in a slightly different way. And that is, going back to the earlier period again, where you are just deciding how you want to do this, how you want to build a new system called MULTICS, what was the involvement of people in IPTO in discussions about a new system? This should be right back into the Licklider years now.

CORBATO: I think, again, the naivete was shared by all. Everyone thought it was easier to build systems than it turned out to be. And I guess a lot of it, I think, is based on a failure - and I am not trying to pin the blame on anybody, but in reflection I think a lot of the problem is that people failed to distinguish carefully the difference between modifying a system that was already working and starting over and building one from scratch to some new design ideas. And the latter is research, where you don't yet know the right way of doing it. An analogy I often used was, imagine you had never built a house before, and you had to try and put all the bricks in place. The obvious way to build it right is to build it brick by brick and see what you have done to date, and then decide how to go on from there. That takes a lot of time. It's a very serial process, and it also presupposes that you have a well-ordered view of what comes first. Here we were sort of rushing a system together as though one had a vision of how everything would fit together, and you would build all the pieces and then you would try to assemble them. And as you come together you find things don't always work together well. And it is that trial and error aspect of system synthesis, which I think is the major factor that people underestimate and a huge amount of... the difference between engineering projects to me is the degree that the participants have done it before. If they have done it before it's a piece of cake sometimes; if they have never done it before it can be a hellish adventure. Now, IPTO started out with Lick as a visionary. He was not an implementor. Oh, he loved to hack a little bit but he was not a detail man himself.

So he thought it was wonderful we were going off on new adventures. The plan sounded great to him. I think, in general, when IPTO put its money into different places around the country they hoped that good thing would come out of it, and they were not micro management and that was all to the good. I don't think they believed it was totally necessary to be as ambitious as we were. But they were willing to tolerate it as long as we wanted to make it happen and wait and see what happened. When the delay began to pop in is when they began to be unhappy, because there were a lot of less ambitious systems which seemed to be very handy and useful and more conventional. The TENEX system is one. So they became less happy with us. But to answer your first question, what was my involvement? It was very little, because I was to a first-order sheltered by people like Fano and later Lick acting as a buffer between myself and the IPTO management. It was only when MULTICS began to hit the hard times about two or three years in, that people began to breathe hard down our necks as to, "Will it ever work? Is it worth doing?" You know, the hard questions, "Should we kill this project?" And there was a very major review made during that period where I guess the optimists... the result of the review was close to turning it off, but they said, "Well, let's wait and see a little longer."

NORBERG: Okay, I have two questions that I want to follow up there. One of them was, this description that you just gave of the necessity to essentially start from the ground up in developing this new system, was this typical of programming at the time in general for these systems? Take the IBM case, for example, developing a new programming system for the 360, would they have had to start from the ground up once they realized what their objectives were?

CORBATO: Yes, correct, except that they were not making as big a transformation from what had been done previously. So the system in many ways mimicked previous systems. However, they stumbled badly in some of the most elementary areas too, and they had a terrible time getting that software up to speed in terms of performance and maintainability. People have forgotten, maybe never knew, but it was hard. It was a close call for IBM. They almost floundered, because one of the problems was that the management rushed the announcement of the machine by at least a year. And the software people, who thought they had another year to sort things out, were suddenly hastily trying to get it out the door. They had also probably underestimated the problems, as most people did. But things

like the assembler on the 360 were monstrous kluges and had something like 17 passes, which is just not well-engineered at all, but in their haste to meet specs...

NORBERG: I asked that question in order to set the context for the next question, which has to do with the IPTO people's response - people like Sutherland, Taylor and Roberts, because in talking to them over the last year or so I have come up with a sort of a sense of how they tended to operate. And there were times when they actually did micro-manage projects, as reported by various people. The whole process of bidding for the ILLIAC IV, for example, was micro-managed, at least as reported by Roberts and others. The development of the network was micro-managed. If there was a view of programming and its difficulties at the time, which was widely held among the community, then there wouldn't be any objections to the sort of thing that you just described; it would be expected. I sense though that it wasn't expected; at least it wasn't expected on the part of somebody like Roberts, who later reported that he came here and said, "Only ten percent improvement on the compilers that you are doing." He didn't mention MULTICS by name, so I don't want to tend to misquote him. But he said he came here and said he didn't want to see any more compiler work, because only a ten percent improvement was being generated, and therefore he would rather put his money elsewhere. Now, that sort of micro-management, it seems to me, was going on in the late 1960s. Did you ever experience it in the MULTICS project, either directly with the people in IPTO or through Fano?

CORBATO: Well, we certainly were feeling the skepticism that MULTICS would ever amount to anything, but it was still bundled as a package. And people weren't saying, "Well, we like this, but we don't like that." So it was an all or nothing type situation. I guess the argument we were making was that we would like to get it up and running sufficiently to demonstrate the ideas, to argue that GE (somewhere around 1969 Honeywell bought up the GE computer division - possibly 1970) be allowed to take over the machine and make a product out of it. People were reluctant to kill it off. There was enough vitality going and enough worthiness to seeing the experiment completed that I guess people were tolerating it being kept going a while. But it was a close squeeze, and we almost didn't succeed. And a few key people were instrumental in convincing IPTO that it should keep going. Ed Fredkin, for one, helped a lot.

NORBERG: Was there a written report developed by this review group?

CORBATO: I don't know; I never saw it if there was.

NORBERG: I have never seen one either. I was hoping you were going to hand me one, actually.

CORBATO: No.

NORBERG: Who else was on the review group besides Fredkin? I am assuming he was.

CORBATO: Fredkin was on it; Larry Roberts was on it.

NORBERG: Just the two of them?

CORBATO: No, there was about a half dozen. Dave Evans of Evans and Sutherland, as I recall. I have lost track... It must be in my notes.

NORBERG: Sure. All right. Let me ask an incidental question here, because I think it does help me to understand some of the characteristics of MULTICS. Let me give you another quotation from one of your papers. My recollection is this is 1972 also. You said that "Experience indicates that the availability of on-line terminals drastically changes user habits, and these changes in turn suggest changes and additions to the system itself." This comment was made in the context of this description of iteration that you were providing the reader as a way into discussing the accomplishments of the MULTICS project. How does it change user habits? What do users do differently with all the on-line terminals that they didn't do before?

CORBATO: Well, the style of the interaction changes. You have to remember that... Once you get a pretty reasonable response you become very impatient for any delay. The analogy I always use is the difference between

carrying on an oral conversation and sending a letter. If a person rereads a letter before responding you don't even notice, because it's several days delay before you get a response. If a person hesitates an extra ten seconds before responding to you in person, it's very frustrating. So once you have gotten yourself geared up to being in an interactive dialogue delay is very distracting and bothersome. So the timing of events becomes significant. Furthermore, dialogues are not casual. I mean, the last thing you want is the system belaboring the obvious and printing out five lines of explanation about something which could have been said in three words. So trimming the verbiage to the correct amount for the dialogue is important. Using language which is indicative of the context of the dialogue is important. For example, if you are operating a program that tries to take a square root of a negative number, the program sends out an error signal. Now, it can say that, but it may be deep in the bowels of the computation and that's the least important. That's not the issue that you as a user would care about. In fact, you may not even realize you are using the square root, because it was used... And that can translate back into the fact that the roots of a cubic equation program is using square root, and what could that mean at that level? Well, it means probably that you have two imaginary roots and one real root of cubic. So the negative square root is trying to tell you that you have imaginary roots. And what could that possibly mean? It might mean that the data you had fed into your calculation arose because someone got a bad data point in, and you were going to get three modes of vibration of a wing or something. And physically, they were all real, but because you had a bad data point two of them were showing up as imaginary. Now, how you reflect that back out to the user is something that one has great difficulty in planning out in software in trying to work through, and people still haven't come to grips with that very well. But it's part of the evolution of software, and we have seen a continued... Time-sharing only revealed the problem. We have seen a massive continuation of the problem in the personal computer market. And the best example I can give is, compare, contrast the personal computer software of 1978 with now. There has been a mind-boggling shift in the metaphors, in the ways people interact. In 1978 people were still using the screen as kind of a teletype. By now, people use the whole screen; they use colors; they use dynamics; they have blinking going on. They have learned to make the obvious thing be the default. It is just a tremendous tuning up towards the way you interact with programs and the ideas in the programs.

NORBERG: And you would see this reflected back into the development of MULTICS, as well, that you were trying

to achieve some of that.

CORBATO: Well, I am digressing a lot in the sense that that is only an example of what can happen when you change. You are forced to rethink how you organize software to support the different modes of behavior. And we are far from the end. The reason I know we are far from the end is that we haven't begun to incorporate video into programs. And I sense that the performance demands of that will be so overwhelming we will have to rethink a lot of how things are organized.

NORBERG: Let me quote for you something that Jack Dennis said, because it follows from this, and I want an elaboration of it actually, because when Judy O'Neill interviewed him she didn't follow up this point. In talking about MULTICS and some of the reasons why it didn't get the kind of attention that time-sharing generally received he said that "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 that MULTICS had to offer in terms of being able to build subsystems and all the protection capabilities in it and so forth, the things that I was emphasizing at that time did not have sufficient value to the user. Now my view is that what has happened is that this just has injected a ten year or twenty year pause in the evolution of computer systems. Now that we are 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. And solutions are going to be along the lines that we were thinking about back in the early 1960s." Can you interpret this statement for me?

CORBATO: I think so. By and large, that's a correct statement. I am not totally convinced that the past is prologue for the present in the sense that I think the solution may be slightly different. But I think the questions raised and the solutions that were developed inside MULTICS would certainly be germane to anybody that is trying to think of future solutions. What happened was that, you know, the economic priorities became paramount, and the microprocessor and distributed systems were things that were basically so important to give people the autonomy they wanted. They were prepared to go that route rather than to try to wait for time-sharing systems to get cheaper and simpler. Let me back off. It seems to me there is a very deep and powerful reason why the computing field didn't

go the route of every organization having a MULTICS. It's the same reason why the public transportation systems have been, in many instances, replaced by the private individual automobiles. The thing that people treasure in the private individual automobiles is their degree of autonomy and control. Although we still have bus systems and we still have airplane systems because they meet other needs. The automobile is the unit of discussion in most transportation decisions [?]. The same thing has happened in the computing industry, and the enabling thing was the microprocessor. That allowed people to have private, personal computers, and that was worth... You give up a lot of things with that, but you gain the ability to control your own destiny. You gain the ability to evolve your computing needs on your own timetable and at your own budget. Now, as people get beyond having gotten their own thing, now they want to get to having their cake and eating it too. Now they want to have [communications]. So they're getting interconnected with networks. They're beginning to interact and just develop privacy and security concerns. So Jack is absolutely right. All the problems that we tried to lick from a central point of view are now coming back to us. The solutions will probably have to be somewhat different in detail because of the distributed nature of the problem, and because of the distributed responsibility, which is the one thing that we assumed when we were building time-sharing systems, a very deep-seated assumption, is that there was a central group responsible for the integrity of the system. And the one thing you have to assume when you have distributed private systems is that there is no central group. There are traffic laws when people know how to behave to each other, but they are based in part on good will. It is very complicated, and so it is a different game. But nevertheless, Jack's basic theme is correct, and we will have to reexamine these things as we get more and more complicated.

NORBERG: When did security become the issue for the MULTICS project?

CORBATO: From the beginning. I guess there was a belief we had to do it right. Well, a belief that it is something you don't easily add in later.

TAPE 4/SIDE 1

NORBERG: I was just rephrasing my question about security and asking you what security meant in the context of

MULTICS in the middle 1960s. And what I am contrasting in my own mind is security in terms of protecting the files of a given user from damage in some sort of accidental way, let's say, not a vicious attack on them by someone else versus the military need for security where things have to be private.

CORBATO: Well, military exaggerate a lot. [laugh] They like to think they are different. There are some differences in degree, and there are certainly differences in the way they approach solving the problem, but we did not have in mind the military needs first. I honestly think that a lot of the military needs are met by systems that we did have in mind, but that remains for... that is another argument. The vision we had was that of the individual person in an organization, like a person in an office or a researcher in a university where there is a combination of a need to have the benefit of privacy, just like when I lock my desk at night. I can do a first approximation and assume that no one is looking at what is in the desk, including the cleaning ladies, including a lot of different things. That does not mean that I think that someone couldn't get into it with a crowbar and, or picking the lock on my office door and so forth. It means that I don't think it's... If someone wants to go to that much effort, so be it. I have a safe for a few things that are a little more sensitive than that. That can be gotten into too, although I would probably know if it had, and so forth. So we have metaphors throughout life on how to deal with things which are sensitive. It comes back to some deep questions like why do you want privacy, because some people have argued there is no need for privacy at all. Well, most individuals are shy about exposing things prematurely, before they have got them to a point where they are prepared to defend them. They are still imperfect and flawed and the last thing they want is an attack. So that they are in the act of creating an idea, they don't want premature release. In some cases there's maliciousness; people borrow ideas from one another and there are a lot of arguments. There is no need to go into that dialogue. So we had in mind just creating a system where the level of privacy was manageable. It didn't mean you had to be private, but it meant if you wanted privacy you could have it. Conversely, if you wanted things to be exchangeable you could also get that. So we were looking after a structure where one could organize a system as one wanted it rather than as you were forced to deal with it because the vendor left it that way. As an example where you are forced to deal with it because the vendor left it that way is the current disaster of cellular telephones, where, because they are not encrypted or encoded in any form, random people can listen in on cellular telephone conversations. There is no way the ordinary user can do anything about it except not use the phone. [laugh] He is forced to use it as it is. We were

trying to give a system where there is a choice. Now, this reflects back on that earlier discussion of the idea of autonomy. The key thing that you have to recognize in the MULTICS system is that we had this vision that there was a central authority managing the system and then we wanted to distribute the authority. The questions of privacy we were going to distribute. And our model was, because we didn't have a better one, was to make it hierarchical. Now, in fact, not all activities in life are hierarchical. But nevertheless, that was the model we used, so that we have the administrator followed by project leaders, or group leaders, or department heads (whatever name you want to use) followed by individuals within the groups. We had it nestable, so that was our vision of how to deal with it. It did not (?) make a useful framework.

NORBERG: Would you use the same approach now as you used then?

[INTERRUPTION]

NORBERG: Okay, I had asked you the question about security, whether you would approach the problem the same way today as you did back in the MULTICS days - the late 1960s.

CORBATO: Well, broadly, yes, but I think the detailed mechanisms of implementation one would have to rethink in terms of the different framework that one is working with. In other words, I think the details of how you solve it may change. But it is a very complicated set of problems, and challenging.

NORBERG: Would you put a different emphasis on security issues today than then?

CORBATO: No, I don't think it is a different emphasis. I think the one thing that hasn't changed at all is that the approach we used in MULTICS, and which I would still use today, is that you try to imagine the real live situations you can get into and try to imagine how you would like the machine to behave. What are the circumstances now? The thing to recognize is security and privacy matters are rarely absolute. You always have to keep in mind what the price of... How bad is it if you fail, and what is it worth to do it? Those are working questions you have to ask at all

times. People rarely do. They often talk about them as absolutes.

NORBERG: Okay. I would like to ask you a series of questions of technical import for a few minutes and then come back to the Bell Laboratories situation. When you realized that there was going to be a schedule slippage in Multics what was your response? Did you add more personnel? Did you extend deadlines? Did you do less work in certain areas, for example?

CORBATO: Well, it was a creeping thing. We began to get more organized. We never stopped working at it. We tried to work, perhaps, harder, but you can't force the pace of some things. At some point we decided to add more personnel, but that can be self-defeating if one is not careful. And so we built up to a... I guess between the three organizations at one point we had about 100 people of which I would estimate 50 were effective. That was our high water mark. And that was pretty complicated, from a management point of view.

NORBERG: Why?

CORBATO: Because there was no obvious hierarchy, no obvious rejection framework for ideas, and we had to develop, in effect, a framework for who was in charge and who decided what would be on the system and what wouldn't. Part of it was based on asking people who had ideas to articulate them in writing. And to the extent that they were able to be persuasive in writing we allowed them to become designers. Then if their design was accepted we asked them to be responsible for its implementation also. It was an attempt to force people to follow through on ideas rather than just write great thought pieces.

NORBERG: Were there differing responses to the slippage from the various partners?

CORBATO: Yes, there were different responses. GE, I think, went from thinking of us as a new product line to being more of a cash and carry customer where they kept working with us as long as there were government contracts buying machines and they saw us as a potential product for the government market. What was going on there was

that the traditional operating system faction within GE kept winning all the administrative wars, because that was where the cash flow was coming from. They kept convincing management that they were the ones that should be favored. So they got the lion's share of attention from the GE management.

NORBERG: Why was such attention necessary if the subcontract was providing the resources for the GE people involved with MULTICS to do their work?

CORBATO: Well, it is a question of which people, a question of the quality of the people. In other words, the company always has a choice as to whether they put their first team on project A or project B. The one thing that was different about MULTICS is it had such an inspirational goal that we got a lot of people to work on the project who probably were of a higher quality than the one we had expected. Frequently young people saw it as something interesting to work on rather than furthering their careers. So we had... I would say, even though this is kind of loose-jointed management framework, we had some very strong people. But, nevertheless, from the point of view of managing corporate resources, when a project falls into kind of a second tier of concerns for the management they don't get first tier attention.

NORBERG: What was Bell's response to the slippage?

CORBATO: Bell's response was to get more and more frustrated, because it had been sold internally to Bell as the next computing environment for the laboratories. And so initially, I guess they decided to go to the 635 to retreat to the conventional operating system, and they developed a certain amount of fondness for that. But more and more, it began to isolate the person in Bell Labs who had been the leader of the pro-MULTICS faction - namely, Ed David. So Ed David was more and more isolated, and to some extent, discredited for not having delivered. And I think it was probably instrumental in Ed David eventually deciding that he should leave Bell, because people no longer viewed him as being a great manager. You know how people oversimplify, make judgments. Within his team they also felt that he ? for not having delivered and it eventually culminated with the see-saw tipping, and the laboratories turning out MULTICS, Bell Labs pulling away from the cooperative venture, and ordering the people on the project to not

continue to work with us (those that had tried to the bitter end).

NORBERG: But yet, didn't you say to me earlier that there was a contract between Bell and GE about it?

CORBATO: Yes...

NORBERG: So did they abrogate the contract, essentially?

CORBATO: Well, the contracts were annual-type things - rentals and so forth. There was nothing dishonest or illegal they just chose to terminate their relationship in an orderly and proper way.

NORBERG: Yes.

CORBATO: It was all correct on paper. It was the end of it, like getting a divorce.

NORBERG: Okay, I want to pursue something else rather than that at the moment. I am interested in the choice of PL-1 as the language on which you would base this new system.

CORBATO: Good question. I wrote a paper on that where I concluded that... Well, I have mixed feelings about it. In retrospect, it was a monstrosity to be trying to use to design a system. But our first conviction was we absolutely had to have a compiler language. And I think that one was still dead right. The next problem was that we wanted one that was relatively advanced and flexible and there weren't many choices in those days. And the third key ingredient was that Doug McIlroy and Bob Morris at Bell Labs, who had considerable language expertise and who wanted to take responsibility for the language effort, were quite persuaded that PL-1 was a good choice. So we tended to let them run where they wanted. And we didn't see any argument why that in principle wasn't a reasonable decision. What we failed to foresee was that the language had gotten terribly complicated in its specification and was even more complicated because people didn't know good ways to implement it yet. So the first implementations that came

out were pretty cumbersome and almost unworkable. Actually, I have to back up. First of all, Doug McIlroy and Bob [Morris], instead of building a PL-1 compiler themselves, which was the style we would have kind of expected, they argued that the best thing to do was to go to a compiler company and buy one. Well, they picked a company which was successful in building a FORTRAN compiler, but they had done that incrementally, learning how to build a FORTRAN compiler. They didn't have much experience in building something brand-new that had never been done before.

NORBERG: I notice you haven't mentioned the company yet.

CORBATO: I am having a mental block. I could look it up. It was a little company, and they failed because of... They eventually failed.

NORBERG: That's all right. I thought you might be deliberately blocking. That's okay. [laugh]

CORBATO: [laugh] So they failed to make the PL-1 compiler, having strung us along for 12 or 18 months. We were, meanwhile, waiting anxiously for it. The more we waited the more frantic we were getting. So once they failed we were suddenly in deep, deep trouble, because we were totally dependent on it. So we hastily rammed some things together (actually Bob Morris pitched in) and built an early PL-1 - EPL, which was a very crude PL-1 compiler, which we began to use to get off the ground. So there was a long saga of struggling to cope with this language. And then, GE eventually, thinking that it had strong commercial promise, took a couple runs at it. The second run was led by Bob Freiburghouse, who eventually did create probably as good a PL-1 compiler as one could build. He became a world expert in that language and helped set standards in the language and so forth, and has since gone on to be part of the start-up company called Stratus here in Massachusetts. And, you know, he was a very powerful guy. But we didn't get a really solid PL-1 compiler for three or four years after the project started.

NORBERG: It's a long time.

CORBATO: It's a long time, and so, meanwhile, we were struggling with all these interim stages. And so, in retrospect, that is far from ideal. I think... But we didn't have a C language; it hadn't been invented yet. So the question of what should it have been is still a wide-open question. And I am never... You know, you don't get to play in again, but I don't know what the alternative would have been. And I still find that a hard question to answer.

NORBERG: That is a very good comment to make, though. Did PL-1 maximize clearness and maintainability, as you suggested in 1965 - ultimately, when it was available?

CORBATO: Two things were required. One, we had to have decent implementations, and two, we had to learn how to use it in ways which avoided all of the complexities. So one had to develop a style of using it which produced lean code and which didn't create... It was very easy to accidentally create a clumsy piece of code, because it would look relatively brief at the source language but it would generate a lot of garbage at the bottom. So, learning what constructs in PL-1 did not lead to complicated code was part of the lore. We didn't capture that in writing, unfortunately - the folklore among the programmers, or style. And we had to learn that, and part of the success was based on that.

NORBERG: Just as an aside, how did this affect developments after that - this knowledge about PL-1 and how it acted? What was learned in a generic sense that affected computer science here, let's say? Maybe you will have to rephrase the question for me, but...

CORBATO: Only slightly. I think what was learned is that you can do big projects using a compiler, and that in fact it is a big contributor to keeping down the amount of manpower you must have on a project. And that is important almost in a quadratic sense. The fewer people you have on a project, the fewer people you have to talk to each other. So the more you can make any individual more powerful, the more he can do per day and the more he can grapple with and the less communication confusion that you have on your hands. So it's... I still think that was a very important lesson.

NORBERG: Can you cite a future project (future = 1970) in which these lessons proved to be effective?

CORBATO: Well, I detached from working with the MULTICS system for quite a while. But, yes, I would say the evolvement of UNIX was probably a major example. I mean, Ken Thompson and Dennis Ritchie, who worked with MULTICS themselves, when they started over and began to do it in a leaner way, first of all, they developed the C language. Now, the C language was based on a language they had learned while working on... it was based on BCPL, which was a language they had first become acquainted with on CTSS while working on the MULTICS project. They were impressed with its elegance and simplicity. And so they carried a lot of those ideas over in creating a relatively elegant and simple language. And they developed all of UNIX using... well, most of UNIX, using C. And so they perhaps went the other direction again where they were so turned off by the cumbersomeness developed within MULTICS, they wanted to get to something small and lean. So they left out lots and lots of ideas because they didn't want to get bogged down, and they were thinking of a system initially for themselves - not the world, but just themselves. And so you keep going from one extreme to the other when you are building systems.

NORBERG: And out of this came UNIX?

CORBATO: Yes, yes.

NORBERG: Okay, because you started out talking about C and I want to make sure I got the point.

CORBATO: Well, in their need to build UNIX, I don't know if it came exactly in this order, but it was essentially... They decided they needed a language to build something with and so they created C.

NORBERG: What was your reaction to UNIX?

CORBATO: Well, it was a great clubhouse system initially. It was designed for themselves. They didn't worry about security and privacy. They left out ideas when they got in the way. And it was capable of evolving. But it still had

some of the problems, because of the times, it is still a system of its era. Now, a lot of evolution is occurring. I mean, when you contrast a UNIX system to, say, the personal computer, say, a Macintosh, you recognize there's a big gulf between them. The UNIX system still reflects, to a first order, the teletype interface we had to timesharing systems - a linear string of characters. Now, that is gradually changing as people bring in ideas like X-Windows, or now a graphical interface is beginning to be interposed on the UNIX system, you still have this phenomena where, in order to discuss some point in a hierarchy of data, you have to type out a long path name, or you have to do things by typing. And you still have this need to being able to point at a long and elaborate name and say, "I need that," rather than having to type it and say it again. There are major differences in the way you deal with systems. So there's a lot of plastic... well, a lot of need to evolve in this system.

TAPE 4/SIDE 2

NORBERG: What I was, at least, trying to lead you to, if it's an appropriate question, is to see as I see it at the moment that UNIX is a response to MULTICS. And what I would like to be able to say ultimately is, "This is MULTICS, and this part of UNIX is a response to MULTICS, and this part is UNIX new." Is there any way to make such distinctions between these two systems?

CORBATO: I don't think... Well, you would have to ask Thompson. My own appraisal is that they were told to stop working on MULTICS. They were frustrated at the experience anyway, and they decided, "Well, it didn't have to be that complicated," and "Let's do it right; let's build ourselves a modest, useful system. There's nothing wrong with the goals, but let's build a system which didn't bother trying to worry about the world, but worrying about ourselves first."

NORBERG: How does that constitute the value-added word of "right?" You said, "Build it right."

CORBATO: Correct. Well, I am sure they were turned off by the "kitchen sink" aspects of the big systems. I mean, that would have been my impression. Too many ideas got thrown together, and trying to capture them all became

tough to code, tough to build, and tough to get performance out of. So they were going to try to strip it down to its more bare... But they genuinely started over. They were influenced by the background, I am sure, but it is a new system. I think to their credit, whenever they saw a design strategy or idea, they didn't reinvent it just to be contrary. They took the spirit of the MULTICS ideas even if they didn't take the exact letter, which I think was very constructive. But they didn't feel they had to include everything that was in MULTICS and all the ideas. And it started out being a much leaner, simpler system, which initially ran on PDP-11s. It didn't have virtual memory initially, as I recall. And then it kept evolving and really was pushed hard out of Berkeley in its evolution.

NORBERG: Let me go back to Bell, for a minute, as an organization. Back in the late 1960s now, this is not a retrospective question; I'm trying to get you to tell me some details about the period. What skills or background did you think the Bell people brought to the project?

CORBATO: Well, first of all, they had some mature users - sophisticated people, like Vyssotsky and McIlroy and Morris. They also had pretty strong knowledge of the communications business - people like the late Joe Ossanna. And they had some sharp people - Peter Neumann, who has gone on to be very prominent with his risk electronic newspaper. They had a good stable of sharp people. In retrospect, they didn't normally operate as a team, nor did they give orders to one another of who would work on what. So everything had to be by inspiration and wanting to do something. And so it was at the managerial level that they didn't help us a lot. They were used to each person doing his own thing and working on some idea that he thought was important or worthwhile. So the people that worked most closely with us were Peter Neumann and Ossanna. Vyssotsky was the team leader, but he tended to be more of the nurturer and the encourager rather than the... He didn't see himself able to order people to do things.

NORBERG: How were decisions made then between the two groups? You described how their group operated, but how were decisions made then at the somewhat higher level?

CORBATO: Well, within MIT it was Glaser, myself, and to some extent, Bob Graham. And then within Bell it was Vyssotsky and, to some extent, McIlroy and maybe Morris. And within GE it was Ed Vance, initially at the software

level, but it went through a succession of people, and John Couleur at the hardware level. And to some extent it was by persuasion and consensus. And that was the strongest mechanism we had. In principle, we could appeal to the higher level of managers - Fano, Ed Davis and their counterpart Walker Dix at GE. But they met once every two or three months and didn't have day-to-day concern with the system. So at the technical level we did it by consensus and persuasion and force of personnel.

NORBERG: Who was doing most of the persuading most of the time? You?

CORBATO: Well, I was part of it. Glaser was another major part of it. The key system programmers were also on our team. We were used to operating as a team, so to some extent, we, in a subtle way kind of took over the management of the project because we already had a team in place working on CTSS. And we were used to thinking that way, and we didn't hesitate to ask people to do things. And we hired them not as researchers but as staff members working on a research project, which is a difference.

NORBERG: Sure. Now, would your people be prepared for these meetings before you walked in with, let's say, an agenda. I don't mean that too strongly, but...

CORBATO: Well, we did one thing, which was conscious on my part, which was we controlled the documentation. So we were able to guarantee that design documents got put out and that we created various streams. We had a design notebook, which we managed the editorial aspects of. And when we had a kind of wild idea, which didn't seem to fit in, we had an open-ended memo stream where we could publish these... People could distribute them via this open-ended set of memorandums, and they could either rise or fall at their own weight. But that was a way of... kind of an escape valve for people who had a bright idea but it didn't fit in, we could put it in that stream of publications. And that made the person feel good, that their ideas were getting out, but if we didn't see how to orchestrate it into the system itself we didn't try to put it in. The key thing that we were always driving for was people creating components of the design notebook. The design notebook became the way that something was accepted. So there was a certain amount of organizational give and take. If one of the people in GE had a guy who

was ready to work on the assembler and he said he was really the right person to do that we would say, "All right, let him do it." And later, if it turned out that person was slipping or failing people would eventually have to say, "Hey, he's not doing it; we need to get somebody else." The framework was such that it delayed decisions... hard decisions. Now, it was probably no worse than a lot of other projects, but, you know, you always had this feeling if you had acted sooner you could have saved a lot of bother.

NORBERG: Yes. Were you involved in any discussions with the Bell people when they were considering pulling out?

CORBATO: No.

NORBERG: Did you have any warning?

CORBATO: Yes, we certainly knew they were unhappy. We knew they were under bad stress. It wasn't a total surprise is the quick way to say it. Disappointment, sure, but not of surprise.

NORBERG: Was there anything vital about the relationship that required you people then to pick up certain kinds of tasks?

CORBATO: Well, by then their participation had atrophied pretty badly, so we didn't lose much except a few people who had worked very, very hard with us - Ossanna and Neumann were deeply disappointed. And I forget whether Neumann had permission to work with us or not, but he did, he hung in there, kept an intellectual interest in the outcome. So did Ossanna. But to some extent they no longer had the corporate blessing, so they were almost doing it on their own rather than as a part of Bell. But we didn't lose a lot. By then the die had been cast and all of the active development work was going on. Well, let me back up. Very early in the game, within a year or so, the momentum of development had shifted to the Cambridge area, most of the major work was being done in this building with a combination of GE people and MIT people. And Bell Labs people came as visitors and to some ext ent worked

at a distance, but they were hampered by that.

NORBERG: Did this sort of interaction continue after Honeywell purchased the GE computer?

CORBATO: Yes. That was just a name change.

NORBERG: Oh, it was?

CORBATO: Well, it clearly affected the GE lifestyle. You know, people had to take on the Honeywell lifestyle. So the people that worked for GE and worked for Honeywell clearly had a different set of issues imposed on them. But to a first order, it was the same team and the same place.

NORBERG: So you didn't see any difference as a result of that between the way the project was approached before and after Honeywell.

CORBATO: No, there are two papers that are retrospective on MULTICS. One is a seven-year program paper in 1972. The second is the one that Charlie Clingen, who by then was the manager of the Honeywell counterpart, and I wrote together on the management of the MULTICS software development. And that was published around 1979 or 1980.

NORBERG: I didn't look at that.

CORBATO: Yes, that tried to look at it from much of the perspective you are talking about now. And what reminded me of that is (I believe it was in that paper) we commented on the fact that one of the crucial problems of the project was that about every... Each organization was always in flux. As with most companies, GE was constantly reorganizing, changing names. It's a way of doing a promotional game in a company. [laugh] But the same thing in Bell. There would be periodic shifts in management. And what you discovered is that you had to resell people on

the validity of the idea, and also in IPTO. So periodically we would have to resell people as to why it was an important project, why it was worth the struggle. So that was one of the consequences of the switch from GE to Honeywell. Honeywell came in and said, "Hey, what's this we've got here? We bought the company, but did we want to buy this and do we want to continue?" So the first problem was to convince them we really wanted to continue.

NORBERG: I find that a little puzzling with respect to IPTO, though, considering that Roberts came from this community, was there at the beginning of 1967, continued through until Licklider showed up for his second tour of duty (of course, he knew what was going on here), and stayed until 1975. Why would it be necessary to continue to keep them apprised of the goals?

CORBATO: Well, put yourself in their shoes. They were like gardeners. They had this big garden with all these vegetables planted. And some were growing well and producing oohs and ahs, and others were still sort of looking scraggly and not blossoming at all. And when you are in that frame of mind you tend to follow your successes, and when thing are faltering you wonder if you need to prune them or cut back. You know, Larry has always been kind of a loner when it comes to doing technical work. He wasn't very sympathetic to the team aspects that we required to build it. He always like things that, say, one person could program, which, he is a brilliant guy and he would do that himself, but he wasn't used to working with big systems and wasn't sure it was the right way to do things anyway. So I think there was a certain amount of healthy skepticism, whether it was necessary to do this and whether the outcome would be worth the trouble.

NORBERG: Do you remember specific discussions with him on those points?

CORBATO: No, not directly. We may have had some, but...

[INTERRUPTION]

NORBERG: Switching away from the companies and back to some of the technical details again, when did you realize that MULTICS was not going to be implemented on multiple platforms?

CORBATO: Oh, well, let me back up. Why did you think we thought it was going to be on multiple platforms?

NORBERG: My colleague thinks this, not me.

CORBATO: Okay. I guess by the time we had tinkered with the 645 architecture as deeply as we did we knew that you couldn't just put this on anybody's machine. It had to be special hardware. So I would have to translate that question... When did we realize that no one was going to imitate us? And I guess our recognition of that was when neither we nor the model 67 of IBM were immediate successes. By both those systems having a lot of teething problems (the IBM 67 eventually being withdrawn), it said that there had to be a very strong reason for producing the kind of architectural mayhem that we had. Now, the first reason would be if we had turned out to have a very successful system, then someone would try to imitate it. But once we realized we had gone on a different path, I think we knew we couldn't casually take a MULTICS system and build a competitor to it.

NORBERG: So that sort of puts in the decision of all or nothing for a manufacturer who might want to incorporate this, doesn't it?

CORBATO: Yes, it does. It makes it all or nothing. I guess our original vision, in the hopes that it would been... a lot of versions of MULTICS would have been replicated, and that by virtue of its presence it would influence people and possibly generate competitors. But the pivotal thing that... our introducing the ideas of segmentation and also introducing the ideas of hardware protection and security in the way we did, at least, guaranteed we would be on a rather lonely path. And we didn't think of it as sealing off the rest of the world, but we thought of it as a way to make something better happen. In retrospect, we did.

NORBERG: We did what?

CORBATO: We did prevent other people from taking over our software and, or imitating it closely.

NORBERG: And when you say that, is it because they had to make the all or nothing choice, or is there some other prevention mechanism in operation?

CORBATO: Well, it's because they had to make the all or nothing choice. To imitate us in a partial way would be suicidal. They would have had to go almost the whole route, because the ideas didn't decouple completely. Now, a few people did try to imitate some of the ideas. I think Bill Poduska borrowed ideas out of MULTICS. He did some of the PRIME computers, and also the Apollo computers. He had been an early participant in some of these MULTICS efforts, and so he was sympathetic to and understood some of the reasons. But in most cases, I would argue, they were kind of inspired by us rather than imitation. Nobody tried to pick up the whole ball of wax and do some kind of large venture.

NORBERG: Oh, let me ask you a different question then, than the one I had intended. And that is, given this development process that ended up in an all or nothing decision necessary on the part of anyone who was going to pick it up, what do you think was the significance then of the MULTICS project?

CORBATO: Well, I think some of the major significance was showing that you could take a lot of these ideas and coherently make them work together. It was an existence proof. But in order for the system to have survived people would have had to continually revamp it and renovate it. And that would have required both money and high quality people. They had the high quality people but not enough - a few of them, at least. But Honeywell, by virtue of its very haste and measured support of the system, guaranteed eventually that the top people, or many of the top people, would despair and move on to other activities. So as they lost their technical brilliance in the team they were unable to continue to make the deep innovations and they didn't have the money to make them and so forth. So it's a sickly degeneration, and so the system did not evolve into the 1980s and 1990s. It did. It stayed alive to the 1980s. It still had fundamentally a teletype-like interface. You know, they've got CRTs as output devices. It never quite got

to the point where today's personal computers are - where you have an all-screen approach and windows and things of that sort. So that dooms it.

NORBERG: Do you think that was even possible, given the structure of MULTICS?

CORBATO: Well, if you view it as a great big piece of plastic, sure, you can always by just continually... in the same sense that one novel can be evolved into another, you know, if you change all the words. [laughter]

NORBERG: But that's not a cost-effective way to do it, certainly. [laugh]

CORBATO: Maybe not, but on the other hand starting over is not always an option, and that is part of the problem that you deal with with all large systems. How do they evolve and who pays the price of evolution? Is it the individual or is it the companies that are supporting it? Anyway, that is a complicated set of issues.

NORBERG: You mentioned a little earlier one of the effects that the project generated in terms of what was learned by the people on the project here at MIT. Were there other contributions, do you think, to understanding on the part of the researchers here that then had some sort of an impact on subsequent developments, subsequent research projects, the generation of some research projects, perhaps. I am looking for impacts, clearly, to see if we can make any positive statements about the impact of MULTICS. I want to say they supported this and the end result was nothing really of value except this one thing you mentioned. There must be others.

CORBATO: Oh, no, there is a lot of impact - a lot of it subtle.

NORBERG: That's fine.

CORBATO: I don't know to what extent it was directly impacted, but I think IBM learned that their tactics in trying to develop the 67 were wrong. We had 1/10th the manpower; we did a system that in many ways outperformed IBM.

We didn't quite make the same timetable they did, but we had a better system that was more viable. And I think part of it was that we had techniques for building the system using the system itself which were very, very effective. I think that we probably influenced other vendors who recognized it, that by creating a hardware, software environment like the one we had, they too could build systems for themselves effectively using this methodology. So each person has to do it in their own framework. But I think the notion that there could be a sort of a software workshop, or factory, where you could build things effectively probably wasn't lost on key people. I can't prove that very well, but we certainly proved it ourselves. That's how we were able to pull off what we did. Another place, of course, was a whole generation of top flight software people moved on to other places. There is a whole litany of, you know, Data General, Prime, Tandem, Apollo. There were people that... and they've gone on to play important roles... some of the people have gone on to play important roles.

NORBERG: Can you give a few examples? I know you feel you are slighting some people, but that's all right.

TAPE 5/SIDE 1

CORBATO: Well, Bob Freiburghouse went to Stratus as did Steve Weber, who was another topflight programmer-designer. Bob Daley went on to work with DEC, I think, on office automation systems. Tom Van Vleck went to work for Tandem, and he had been involved with administration and... that is, the softwares for administering the systems and with questions of reliability and backup. So that was very consistent with his own interests that he'd had inside MULTICS. Peter Neumann has gone on to become an expert in building reliable software, and has taken on a professional interest in the problems of the hazards of large computing systems - runs as a risk column. Let's see, I am slighting people right and left. Bill Poduska, of course, has gone on to become one of the founders of Prime and then went on to form a second company, Apollo, where a lot of the file system ideas were stimulated, for example, by MULTICS. He has since gone on to a third company, started out being called Stellar and is now called what? They joined up with another company.

NORBERG: Stardent?

CORBATO: Stardent. It's now Stardent, right, because of the merger. Let's see, who else? Noel Morris went to Prime; he, unfortunately, died. Max Smith went to Data General, I believe. I may have some of these company names mixed up.

NORBERG: So what you are saying is the activities that these people were involved in in some way affected what they did after they left here and that that is an important contribution spread over all these companies, and if they hadn't had that training God knows what they would have done.

CORBATO: I think there is another large group of people who I obviously don't know by name, but [resulting from our publications]. Probably one thing that MULTICS was different from most is that we published a lot about what we were trying to do, and then we tried to publish a little bit about what we had done. And so, more so than most systems, there was something for people to study and read. And a lot of students in various institutions and also just professionals, I think, have read those articles. And to some extent it's been a consciousness raising experience for them to realize what are the problems in building systems and what are the kind of hazards you can get into. I don't think that we taught someone, you know, here's all you have to do: a, b, c, to accomplish x, y, z. We basically got people to think at a more sophisticated level about systems design. And I think we probably had some influence on the need to articulate the design process. And I don't mean just blindly writing down a bunch of detailed specs. The design process could be perverted by people who fundamentally treat it as a bookkeeping exercise. What I mean is that if you haven't got some sort of grand vision of what you are trying to do and some sense of structure to it, then it gets to be a real mess when you through the system. So we probably were better than most projects in trying to spell things out. It turned out to be crucial for our own survival, because, as we had to keep reindoctrinating people as there was a lot of personnel turnover, it was crucial that we have a way to bring them up to speed without spending a lot of project resources. But, you know, the quickest way to kill a big system is for people to not feel they know what they are doing. [laugh]

NORBERG: Can I ask you to make one more comparison?

CORBATO: Yes.

NORBERG: Since we are now obviously in the early 1970s, if not the middle 1970s, can you compare the sort of ambience in Project MAC as it was beginning to transform into the LCS here in the middle 1970s as compared to what it was like when you people first started in 1963?

CORBATO: Well, there was a tremendous change in climate. In 1963 one of the reasons... There was some sense of technical optimism. Things hadn't settled down completely and one of the reasons that Lick was able to go down to Washington and nurture timesharing in general was there was a recognition within the DOD that they should pay more attention to R&D and hence DARPA and giving it significant budget. A lot of that recognition came up, as I understood it (this may be apocryphal) was that one of the consequences of the Cuban missile crisis was that they realized that most of their command and control systems were crap. And one of the stories was that some general had asked that the plug be pulled because it was confusing matters rather than helping. And it was out of that kind of recognition, that they really didn't have things in order, that they began to pay more attention to R&D. And Lick, I think, was able to use those concerns to help get him good budgets. So there's a terrific sense of a reason for doing something followed by a lot of technical optimism. And then at the latter part of the 1960s a rather crucial change happened. We stumbled into the Vietnam war, or backed into it. And the deeper we got into the war, the more and more frantic people came for short term results. Furthermore, there was a lot of sliding around the budgets to try and cover the expenses of the war. And so the research community saw either the dollar shrinking or more accountability for what was being done with the dollars. What were you really accomplishing? Why wasn't it already working? I once remember looking at a newspaper, and I wish I had clipped it out. There was a chart (published around 1970) of the U.S. troop deployment in Vietnam. It started at almost zero in 1965 and built up to a peak around 1969 or so. And I thought, "Oh, my God, that's almost exactly the same chart as our manpower applied to MULTICS," [laughter] and I refused to carry the analogy any further.

NORBERG: Yes, well, indeed. But so what? Are you suggesting that there is some sort of relationship between

massive projects in this case where manpower, regardless of the absolute numbers, essentially builds up according to the same equation?

CORBATO: It really is the latter, just what you suggested. There is a relationship, you build up project effort. The thing is that our problems in developing MULTICS coincided with the problem of the Vietnam War almost perfectly. I don't think there's a cause and effect relation. Good point; I am glad you clarified [?]. But it certainly aggravated the concern that people had for our success or failure, because people at that point were beginning to question all large ventures. DOD was under pressure just to deliver. About that time, remember the Mansfield Amendment, which was another attempt to get the DOD out of doing other things and tending to its [basic mission]. So there was a lot of heat being thrown at us.

NORBERG: So how did that change the atmosphere here at Project MAC?

CORBATO: It put a subtle but relentless pressure on some sense of deliverable. Before that, everyone had been kind of, you know, with a more benign... If you spent a lot of money on research some of it will come back in a nice way. But here we were suddenly feeling, "Well, is this money being well-spent?" It forced people in... And so, gradually, from 1970 to 1990 I would say, there's been a gradual... perhaps not linear, but long-range trend where people are more and more micro-managing, more and more expecting things to happen for dollars spent and less and less willing to take the long view. What they don't do, as they often do in science - in pure science, ... is they don't find a good man who is doing brilliant work or something and say "We will back him no matter what he is trying to do," practically. They don't back the man; they back the project.

NORBERG: This is true even in science now, wouldn't you say? When you think about all the debate in the National Science Foundation about what they ought to support and what not support.

CORBATO: It's becoming more and more so.

NORBERG: Becoming the same thing.

CORBATO: Yes. But... And of course, you know, we're digressing from MULTICS by a long ways, but one of the things that is saddening is that the U.S. in proportion to its GNP is not spending nearly the amount of money on R&D that other countries are. We just don't recognize the long range implications.

NORBERG: How did these changes that you just described that were going on in the late 1960s in the world around us... how did these affect discussion within the laboratory, or within Project MAC? Did you approach proposals in a different way, and if so, how consciously did you make this change in approach?

CORBATO: Well, there you would have to talk to people like Mike Dertouzos.

NORBERG: I have. But that's the post-1974 period. I am looking for the transition.

CORBATO: I think there was... It was a general escalation of trying... It put more and more heat on... Well, I wasn't in on the proposal writing directly. Fano and Licklider and Fredkin...

NORBERG: In the early years.

CORBATO: In the early years, right, and then Dertouzos have been the ones that have run the show there. And my impression is that they felt more and more required to argue more closely rather than broadly about what would be accomplished in research. Periodically research managers would be more and more directed in saying, "Well, I don't think that work is very interesting," which was a way of politely saying, "We don't want to support that." [laugh]

NORBERG: Well, how does that affect a faculty member's position here at MIT?

CORBATO: Well, there's been a very... And I don't fully understand this, but there's been a very sophisticated

impedance matching that goes on between DARPA and MIT. And I think it's replicated at other places. They in turn direct and they give guidance and say, "We don't like this..."

NORBERG: This is DARPA.

CORBATO: IPTO.

NORBERG: Right.

CORBATO: But they also have left enough latitude in the hands of the directors so that they can mercifully redirect people and either find new ways for them to participate, which are more acceptable, or carry them for a while on some general funds as they look for different opportunities. And that latitude... in principle, the university is stuck. If an individual faculty member can't get research funding the university picks up the difference, and in some instances that has happened. But to a first order, the laboratories have been successful in carrying people along too on projects which are in some way acceptable to DARPA. But you can't keep it up. If someone isn't doing things that DARPA likes, eventually they have to move on to something else.

NORBERG: Do you remember discussions, not of the kind of who moves on, but do you remember discussions in the early 1970s about this sort of changing nature of our work, let's say (ours, being Project MAC), in that we now have to be more concerned about deliverables - not necessarily DARPA's goals, but about deliverables now. I mean, things do have to come on-line with some regularity. Was there discussion of this kind?

CORBATO: Well, it has always been discussed sort of abstractly.

NORBERG: Yes, but usually it's publications, and we don't worry too much about that in the academic world.

CORBATO: Well, "deliverables;" I use it kind of casually. In fact, we never thought we were producing

"deliverables" in the factory sense. What I mean is some evidence of success, and that could be a publication, or it could be... I think in the case of DARPA they don't count publications as much as they count impact. And we care about that too, so in that sense we have been together. But publications are a way of achieving impact. So, demonstrations of systems, prototypes, things that might make people sit up and take notice. That's more important. In that sense we mean deliverables. But it means that there are some judgable milestones, or judgable points in the project. And, yes, there has been some pressure, not all of it bad, but... And I don't think it has gotten worse. But the kind of signals... It's in the last few years, for example, that IPTO has started talking about less umbrella funding and more sub-contracting.

NORBERG: Sub-contracts of what kind? Sub-contracts like project tasks?

CORBATO: More like NSF where you say such and such... "We like this investigator and what his team is doing, but we are not talking about the whole laboratory, in essence." You see, the pattern before that was the whole laboratory got a DARPA grant, where everybody was written into it, and then the director had some discretion about moving money back and forth to manage it successfully. As one thing faltered he could pick up the slack somewhere else.

NORBERG: Well, I recall reading the 1986 proposal for two year funding from LCS, to IPTO (ISTO at the time). And it seemed much more project-oriented. Things were confined to specific areas of programming language development, and system development of communications, and much less the sort of let me call it "visionary aspect" that we find in the proposals of the 1960s?

CORBATO: Yes.

NORBERG: So that confirms what you said.

CORBATO: That is the sort of thing I am talking about. And my understanding is that people like Dertouzos, in an attempt to be responsive to the perceived needs or wants of IPTO, were compartmentalizing it that way on purpose.

People could say, "Well, I like this a little more than this," or "We think this," - to some extent giving them the option of showing a preference list. I think by and large there has been a good amount of rapport between the directors and the IPTO managers. And I suspect there's a little bit of horse trading there, where, you know, if two years in a row this project doesn't seem to be going anywhere they begin to tailor it down, even though the director argues that that's an important project. They don't just chop things off.

NORBERG: Yes. Do you think the attempt to find other outside funding helps to mitigate some of these negative results that could come from a reduction in DARPA funding?

CORBATO: Sure. It has, and it's been pursued. Dertouzos has been the primary pursuer, because he has been in office the longest. But it helps one not feel beholden entirely to another group. The price you pay is that you have to worry about more people to please, so it makes people scramble more. But I am sure it's helped give a little more sense of balance and of talking to a larger audience than just IPTO. But despite Dertouzos' best efforts I don't think he's ever gotten beyond about a third of outside...

NORBERG: That's right; that's what he told me. He was unwilling to say too much about the finances because he wanted to keep it muddied and therefore didn't want to make any public statements about it. But he indicated somewhere between a quarter and a third is coming from outside. So what you are telling me indeed does confirm that. I want to thank you very much. This is extremely helpful for me.

[INTERRUPTION]

CORBATO: Fruits of ONR. It was done by...

NORBERG: Dennicoff?

CORBATO: No, it wasn't Dennicoff. It was someone... where he studied the influence of ONR, the things they've

gotten. One of the things they got was WHIRLWIND and what WHIRLWIND had done for ONR and the nation. But it is again a complicated mapping of how an activity like that eventually traces to a lot of roots.

NORBERG: Is this a recent work you are talking about, or is it earlier?

CORBATO: About four or five years ago.

NORBERG: Oh, because there's a recent book on ONR by Sepulski which came out last year.

CORBATO: Oh, that's not the one.

NORBERG: Okay, if you remember it I would be interested to know it, because I am not sure I have seen it.

CORBATO: I don't think it was a book; it might have been a report.

NORBERG: And then there are other things like HINDSIGHT and TRACES and so on where there were specific things that we can look at in connection with the computing field, which are certainly helpful in judging the impacts. But it's very difficult, I think, to make the argument that one of the biggest impacts of the funding that came out of IPTO to the various elegant, excellent computer science programs around the country over the years in terms of manpower, because we can generate lots of manpower. If you have lots of tennis players, does that mean tennis is important? You see, so... [laugh]

CORBATO: I forgot to mention one of the great coups that came out of MULTICS. It's something that you don't plan on, but yet it happens. Two of the people that had worked on MULTICS and cut their eye teeth on the system were Dan Bricklin and Bob Frankston. And their claim to fame is they went on to build Visi-Calc, which has produced a revolution in the personal computer market now. Now, their whole frame of thinking and education... Basically, they got educated on MULTICS. We didn't design spread sheets for them; they did that on their own. But they had a

high level of sophistication about systems and what they could and couldn't do and what the issues were, and that's the kind of breeding ground we were.

NORBERG: Yes. It's another good example, like the Apollo one.

CORBATO: Yes. It doesn't mean it couldn't have been done with other systems, but we raised more questions and more sophisticated issues with people. That's why it [MULTICS] remained, even though it wasn't a commercial success, why, the system stayed interesting.

NORBERG: You know, that raises another interesting question that I will have to explore, and that is, can we use other systems that were developed, let's say UNIX or maybe something else out on the West Coast at Rand or SDC to demonstrate that the impact of those were somewhat smaller? I don't know how you quantify this, but it was somewhat smaller than the impact of something like MULTICS. Is that a reasonable question to even exp lore, do you think?

CORBATO: Well, if you were able to show it (I am not sure how you would show it) you would probably trace the cause back to the documentation issues. We had books written on our design. We had papers written.

NORBERG: Isn't that true of UNIX as well?

CORBATO: Oh, UNIX is an exception. UNIX by now has been beat to death. And UNIX obviously has had a lot of impact. I suppose you [would] in some tenuous way say we set up UNIX.

NORBERG: [laugh] I might.

CORBATO: Well, in fact, Thompson has been vociferous in saying he didn't imitate MULTICS. And I will give... He had the right to say that, but I can't believe he wasn't the beneficiary of the background of MULTICS. So that it was

in his... That was the foundation from which he started a new venture and he benefitted from all of the foundation...

NORBERG: It would be interesting to see what the interviews that are being done by the Bell people of the UNIX group reveal in terms of what they say about where they were influenced and so on, because one of my colleagues - one of the best historians of science around - who is sort of the overseer of the project (he's not actually doing the interviews himself but overseeing the project) will make sure questions like that are asked. And they have promised to give us that material as well so that we can make the comparison at whatever levels that are possible. But, back to the question again of whether or not one can quantify these effects in terms of manpower, I think it's a critical one for us. We tried to do some analysis of where faculty are, where tenured members of faculty in the major Ph.D. programs are, and where they came from originally, and so on. And the numbers are intriguing, but they are only statistically relevant. And so we can't make a case on the basis of them. But it would be nice if we could do so.

CORBATO: Well, see, at the very least UNIX was a reaction to MULTICS, "We don't want to do it that way so we are going to do it differently." And the first proof of that was the name. The name was a joke. It was pun based on MULTICS. UNIX is not a MULTICS.

NORBERG: I see. I didn't know that, actually. [laugh]

CORBATO: [laugh] So I think Thompson, to some extent, doesn't like to feel that he borrowed or stole the ideas from MULTICS, so he would argue for his own originality. And he certainly deserves credit for having been the mastermind to know how to reject that... That's non-trivial. He was a very clever guy even when he worked on MULTICS, but it takes extra good taste to keep things out of the system.

NORBERG: Yes. Well, very good.

END OF INTERVIEW