FERNANDO CORBATO: The history of time sharing is the subject of the lecture. And I'm not going to try to address that. Instead, I want to address the context of our speaker who has played a key role. And I want to go back to his roots a bit, because I want to explain why he's in a position to give this distinguished lecture.

I won't discuss his books. He has several. I won't discuss his long list of papers. I won't discuss is patents, the awards, or the many academies he belongs to-- has been elected to. He has a truly distinguish record. As the program notes briefly outline, Bob Fano was born in Torino, Italy. He came to this country when Europe came under the totalitarian grip. He came to the US and MIT in particular.

He got his bachelor's in 1941. And he went on from there as a student engineer at General Motors, whose specialty was power engineering. And he tells me that he was in charge of maintaining the welding machines. And he lasted all of six months. And he decided that was not for him and came back to MIT to enter graduate school, clearly a great decision.

As a graduate student, he had a nice career of being a TA and then an instructor when that was a meaningful rank in those days. It still is. He became a staff member of the radiation laboratory during wartime. He moved into working with microwaves, antennas, basically traditional electromagnetic or modern electromagnetic applications.

He got his doctorate in 1947 just after the war. And he joined the faculty as an assistant professor. At that point, he moves into his second career, because one of the amazing things about Bob Fano is he's had about at least four careers depending on how you count. The first was this power machinery. The second was communication theory and information theory. And that was a very fruitful period. It lasted for approximately a decade.

And he also at that time became very active as an educator. And he was part of the curriculum revolution that MIT led, which led to our current notions of a core, and in particular, led to a couple books which he and Dick Adler and Lan Chu authored. By the early 1960s, the time was ripe for time sharing. But it hadn't yet happened.

I'm not going to discuss that because I know our speaker is going to talk about that. Suffice it to say, it was a watershed period-- very important period. And Bob was right in the thick of that. But he also played an even more important role. And that was he played the role of establishing computer science at MIT. And this is not as well known. So let me spend a moment to outline it.

The situation in the earliest 1960s was we had all the ingredients of a computer science department. But we didn't have the focus. And we certainly didn't have the leadership. Most of the participants were relatively young. And we were missing the leadership. Bob Fano was the person that recognized that and stepped in and supplied it. And the major outgrowth of that was Project MAC, which has in turn evolved and now comes in two flavors-- the Laboratory for Computer Science and the AI Laboratory.

But out of that same framework which the time sharing served as kind of a nucleating base, came the first beginnings of a computer science major and what is eventually evolved to be a de facto department within ECS. The computer science undergraduate curriculum was basically started and established under Bob Fano's urging. And it now accounts for approximately 10% of our majors, undergraduate majors, of course.

A few years later, Bob Fano became the first associate head for Computer Science. And that, in turn, was a recognition that, in fact, the subject had become sufficiently important and required enough attention that, in fact, it needed further management. And finally, let me before introducing Bob, let me just leave you with one brief anecdote about names.

We all know names are terribly important as symbols. And I hope Bob will go into the name of how Project MAC got to be what it is. But he probably won't mention that when Bob first came to this country, he was named Roberto Fano. And when he finally came to be naturalized, he was debating whether to change it or not to Robert Fano, to be a little more American.

And he got to talking with his friend Professor Al Hill. And apparently All Hill came up with a brilliant thought. But since he was in the Boston area, that the way could really get ahead was to make a change, but not to change the letters, but to change the punctuation. And that all he really ought to do is change his name to Robert O'Fano.

[LAUGHTER]

Well, the loss to Boston politics has been our gain. Let me introduce-- let me introduce Robert Fano.

[APPLAUSE]

ROBERT FANO: Thank you very much Corby. Al Hill still calls me O'Fano. That's a true story. I will be talking tonight about events that took place in the decade of the '60s, 20 years plus or minus 5. I would say that was a golden year for computers, for computation at MIT. A lot of things happened besides time sharing. And more importantly, as Corby mentioned, the foundation of the educational and research program in the department were established at that time. There is no question that time sharing was at the center of the scene, both as a research topic and also as a tool for a lot of other work.

But it was by no means alone on the scene. A lot of other computer science research was going on in that period, particularly the Artificial Intelligence Group, which was originally part of Project MAC, was very active in that period. And I hope that some of the other topics in computer science will be discussed at further meeting. I barely have enough time to talk about time sharing.

Now, I'm not going to make this a historical talk. I won't talk about facts and dates. This is not my style. And it would be very boring. What I'm going to focus on is the ideas, the interaction of ideas that took place at that time, and the people in whose mind those ideas were born and evolved.

So that's going to be the subject. I wish I could do something more. I wish I could bring to you the excitement of the period. But that's very hard to recreate. Now, to start with, I would like to describe to you what computation was around 1960 so that you can see how time sharing came about.

While there was a computation center at MIT, it had been established in 1956 by Philip Morris, Morris was, I should say, is a very distinguished physicist is also well-known because his work during World War II lay the foundation for a new discipline that got to be called operations research. And as a matter of fact, he established the Operation Research Center at MIT.

Now, in 1956, he persuaded IBM to establish at MIT a IBM 704 installation to serve one shift MIT research and one shift research in other New England colleges. This was established, of course, charge-free. But MIT promised in return to put up a building. And Building 26 was then being built.

So they decided in a hurry to add a wing to it. And that wing is where the Physics Library is now. That was the computation center established in 1956. Now, originally, the machine was a 704. It became a 709 in 1960, a 7090, which was the first IBM transistorized machine, that is the 709 transistorized, came in, I believe, in 1962. And the 794, which would adjust the addition of a floating point, arithmetic unit, came in shortly thereafter.

Now, want to show you what-- first of all, let me say a computation center was something special in those days. It was both a showpiece in the sense that every group touring MIT will stop in front of the Computation Center.

There were big windows. And you could still see them over in Building 26. And you saw all the time crowds gaping at this marvel of modern engineering, the computer. But they couldn't get in, just look through the window. It was also a temple with its own high priest.

Now, I want to show you what it looked like then. Well, let me see if it works. I'd like to get a picture if I could.

AUDIENCE: We're being observed [INAUDIBLE].

ROBERT FANO: Huh?

AUDIENCE: We're being watched.

ROBERT FANO: OK, it will appear-- well, that is the Computation Center. And right in the middle of it there is the Associate Director Dr. Fernando Jose Corabato. Of course, he didn't have a beard in those days. And you look a little bit younger, I must say.

Now, he was associate director. And I must mention that one of the major contributions that Philip Morris made to time sharing was that somewhere in the early '50s he gave an assistantship in computation to Corby-- Fernando Jose Corbato. He is generally known as Corby.

And that's how he cut his teeth on software by working on Whirlwind. Now, when the Computation Center was established in 1956, Morris had the further good judgment of offering a research associateship to Corby, who had just completed his doctorate in theoretical physics. So without knowing at that time, he made sure that the future Mr. Time Sharing would stay in computation. So that's a significant contribution to time sharing. Because of course, you all know there really is Mr. Time Sharing. I'm just talking about what he did.

It's a long-established tradition, as you will here, that he does a thing, and I talk about it. It's kind of an act we go about. I'm the front man. Well, now the computation center was operating what was then called a closed shop, which meant that people who had a program, and incidentally, the program were all punch-- so-called IBM cards.

Let me say, this box here contains a thousand of these devices, the IBM cards. And there are four more down there. Now, altogether, they have a storage capacity essentially equal to a garden-variety, double-sided diskette, roughly. Now, that gives you some proportion, you see. Now, all programs had to be punched on these cards. So people would arrive to the Computation Center. There was counter. And you gave them a batch of cards.

They would take them. Very efficiently an operator would go to a machine that would translate, read the cards, and put them on tape. Then the tape was run on the computer. Then the output, again, went into another similar machine of the same, which would generate some printout or some other cards.

All right, now, the whole process turnaround time was pretty long, something like one or two days often. Am I correct? It was about that. Now, and people printed inches of printout for two reasons. One was that to debug your program, you had to get what was called a memory dump. All right? That's one reason.

The second reason that you could get at a computer so seldom that when you got do it, you better compute anything that you could possibly think and print out anything that you could possibly think to make sure. Because otherwise, it would take a while before you got another shot. So that was the type of operation.

Now, to give you another order of magnitude, the 7094 that existed in that period-- that's a 7094. I read it carefully. You'll see 7094 written on the console over there. But you are too far away-- 7090, I'm sorry-- a core memory of 32,036 bit words. Now, that memory list rented for $17,000 a month. All right?

The instruction processing unit rented for about that much too, $17,000. The arithmetic unit, which was a separate unit, was half that much. That gives you some idea of the cost in those days. And it's important to keep in mind that. Now, the speed, I believe that the memory cycle was something like 2 microseconds, something like that.

So you realize that it wasn't quite the horsepower of a PC. Now, keep that fact in mind because it's important. Now, there was a lot of frustration, as you can well expect, because if you misplaced a comma in any program, well, it was two days that have gone. You corrected the comma quickly. But then you had to wait two days to have another shot. So that was very frustrating.

Now, this is the typical mode of operation of a university computation center the industry was the same thing. It was a typical operation. But at MIT, however, there was another style of computing as well. And that is very important. It has a lot to do with time sharing. The alternate style of computing that I will call online computing really dated back to the Whirlwind computer.

Since it was a homegrown computer, people were not so worried about how much it costs if you said that and fiddle around for awhile. So people started using it online. Now, the Whirlwind computer was ready to be demolished around 1951. And suddenly, a committee discovered that you could build an air defense system around the computer. That's how Lincoln Laboratory was born.

So the Whirlwind computer became the primary tool in the original experimentation on the air defense system. But another thing happened. You see, the whole group under Forrester-- I'm sure many of you heard his talk-- it was the first talk in this series-- moved to Lincoln Laboratory with them. So a tradition of online computation and development, Lincoln Laboratory, got established.

Now, there was a person that was very critical in that respect. His name was Wes Clark. He was a top computer designer. And he designed the TX-0 computer, which was shortly thereafter loaned, permanently loaned, to the Electrical Engineering Department and started students here working online, very important, and then developed the TX-2 computer, a much more powerful computer, which because Clark believed in that, was used online.

That is, you signed up on the board. You reserved for half an hour or for an hour. And as long as there were slots available, you would have the computer totally at your disposal. That was a computer with 64,000 words of memory. It was the first big core memory ever built.

He also build another little computer this time called LINC. And that was a computer that was specifically designed for real-time laboratory experiments. And that had tremendous influence too. So there was this whole tradition of online computation and computers in Lincoln Laboratory. Now, I may mention also that one of the people that were in the team that built the TX-2 computer happened to be Ken Olsen who left Lincoln Laboratory shortly thereafter and started DEC. You See? And he is still chief executive officer of DEC.

Now, that the addition went into DEC. That's why they came out with PDP-1 computer around 1960. Was it '60? Something like that. Yeah. For online use-- so our whole tradition of online use of small computers, minicomputers, grew out really of Whirlwind.

Now, there was another path, however, well, let me show. One of the very important pieces of work done on TX-2 was what called a program called Sketchpad by, that was a doctorate thesis by Ivan Sutherland. Now, Ivan Sutherland did this work. His thesis was completed in 1962. And you'll see him there at a console of the TX-2.

Now, a scope was the routine thing for those computers. I won't call that a mini computer. It was a big one in those days. And it's there. He developed the first that I know honest to goodness computer aided design graphical system. And that's it.

But there was another ramification. You see Whirlwind grew up in one of the department laboratories, which at that time was called the Servomechanisms Laboratory. It changed its name when Frank Reintjes over there became director it became the Electronic System Laboratory. And now it's called LEES. So it's changed names. But it's still the same laboratory.

You see, since the Whirlwind was born there, several people who stayed, who didn't go to Lincoln, were permeated with the same ideas. And there were some very significant computer projects that took place there. First of all, there was a numerically controlled machine tool very early. And then the APT language for programming-- for generating the tapes that control the machine.

And then finally the laboratory got together with the Mechanical Engineering Department. And by 1959 or '60, they had a project on computer-aided design. Now, one of the people, who cut his teeth in APT, is over there. His name is Doug Ross. And I think I've seen him over there. Stand up, Doug, so that people can compare.

Now, Doug, continuing his work on the language as it was called-- Automated Engineering Design-- and then not long thereafter left MIT was a founder of SoftTech and is still chairman of the board, but now is part-time lecturer in the department again. Now, at about that time, I'm talking now '62, no, 1960, there was a very important paper that was written. And it created a great deal of interest in the community. The author was JCR Licklider who wrote a paper called "Man-Computer Symbiosis."

Now, you see it from Lincoln Laboratory, these ideas permeated the firm of Bolt, Beranek, and Newman where Licklider was vice president. Now, I must also say that Licklider was a distinguished experimental psychologist and was on our faculty just around 1950 for a short period before going to BBN. So you know, he was a member of the family. And he got enamored of computers and wrote this very influential paper called "Man-Computer Symbiosis" where he was emphasizing the advantage that one could draw psychologically from man-machine interactions. That was a very influential paper.

Now, this is the environment that existed in 1960. All right, who invented time share? Well, obviously, it was a very favorable environment. As far as I know, the first written mention something that I would call a time sharing system was a memorandum dated January 1, 1959 handwritten by John McCarthy, who was a member of the faculty of the department at that time, three years later.

He left for Stanford where he has been ever since. Now, let me say a few words about John McCarthy. John McCarthy is well-known for a number of things. And about the same time, as a matter of fact, a year later, he invented LISP, just plain that. And wrote a classic paper were the theoretical foundation for LISP. At the same time he was a member of the International Committee that defined ALGOL 60.

So he was a major figure in the computer field. So he wrote this memorandum. This was a memorandum to the director of the Computation Center, which was professor Philip Morris. Shortly thereafter in June of the same year, 1959, a Britisher by the name of Christopher Strachey presented a paper at an international meeting essentially making the same suggestion.

Now, a couple of words about Christopher Strachey. Christopher Strachey, I think was an EE by trade, I believe, or a physicist. I'm not sure-- one or the other. But he became an exponent, a very important figure in computer science in England. He worked on theoretical linguistics.

And as a matter of fact, he was, in the '60s, a visiting professor here for a year. And we gained a great deal in the development of our curriculum from his teaching. Unfortunately, Christopher died relatively young. I think he was still in his 50s in 1975. I had the privilege of spending an evening with him I believe something like a couple of months before he died in Oxford. He had a chair in Oxford.

Now, what were the goals of time sharing? Obviously, one strong motivation, certainly on the part of McCarthy is to eliminate the frustration that resulted from misplaced commas that he was after that shortened the turnaround time to practically nothing. That was his goal. He felt also that big computers were not efficiently in a real sense utilized. He wanted to have an efficient utilization.

You've got to be very careful about efficiency. When the community of users puts out a large printout and computes everything that you can possibly compute to save themselves time because of the long turnaround, it is not an efficient operation. So if you look at the overall efficiency, there was a great deal that could be gained by what got to be known as the time sharing system.

Now, he also had another agenda that had to do with AI. John McCarthy was an AI guy. He had invented LISP Of course, he used a lot of memory. So he wanted big memory. Big memory, particularly then, was very, very expensive. So he was looking for a way of building economic justification for a large memory. When he didn't use it, somebody else could use it.

So that was clear then. I remember very clearly that was part of his agenda. And he was right. Generally speaking, what he had in mind was to combine the power in those days was great of the 7090s. Even if I denigrate it now by comparing two if you see there was a very powerful machine for those days.

It combined that with the easy access that had become a characteristic of the TX-0 computer that was more or less next door. It was one floor up, which was an 18-bit machine. It had the power of an 18-bit machine and a small core memory had. All right? Although it was far from trivial. So he wanted to combine the two.

Now, John McCarthy also gave a lecture. As part of this centennial celebration of MIT, there was a series of lectures in 1961. And he came out in than that lecture with the idea of a computer utility. Now, this is a very important notion. You see, it's one thing to think of time sharing a computer. Another to think about a computer being a public utility to supply computer power to all people.

And as a matter of fact, it's this notion that really got me interested, and in those days I used an awful lot the analogy in talking and writing of computer power and electric power. The real big difference that a distribution system as you could do with electric power made, was that power tools were available to individuals as individuals, amplifying the capability of the hands.

Without an electric distribution system, you couldn't have power tools only in a factory with a steam engine running the whole thing. And my vision of what a computer utility meant in John McCarthy's vision was that you could distribute computer power to amplify the intellectual capability of people just like power tool amplified the physical and manual capability all individuals. That was the kind of the image that arose from his talk.

Furthermore, MIT was very worried about this computational facility because computation was just booming around 1960, '61. So a committee was appointed to study the computer needs of MIT. And that committee, which in the end was chaired by John McCarthy, wrote a report which strongly recommended the establishment over a time sharing computer system.

By that time, work in the development of a time sharing system was already going on, of course. But that was a very strong recommendation. Now, before I get to the time sharing system itself, let me point out that while the notion of time sharing gained quickly a lot of support, that was by no means unanimous, by no means. And some very outstanding people in the computer field were dead set against it.

I am mentioning this because it clarifies some of the problem. Your position largely, but not exclusively, was based on efficient utilization of this very pressure resource called computer time. It was a very precious resource. And many people were so worried about it that it didn't even occur to them there was also something else called precious human time. And one had to make a reasonable compromise between using the two efficiently. It was not all one way.

Let me mention some examples to be very clear. I remember visiting Dick Hamming of Hamming Code fame at the Bell Telephone Laboratory. And he took the attitude that it was essentially morally wrong to allow somebody to type in an instruction to a computer. He was serious. He was serious.

While a little bit at a less scale, there was Gene Amdahl, who gained a lot of fame as the designer of the 360 system for IBM, who just found it was a bad idea to interrupt a program while it was running. He just thought it was bad, again, inefficiency reason and for reasons the designer of the 360 showed very clear these very strong feelings that he had about that issue.

I remember sitting next to Wes Clark at the Lincoln Laboratory at a committee meeting in the department. And somebody mentioned time sharing. And he turned to me and said, what is there to share? His view was the power of the computer was so small that there was not enough to share. It didn't make any sense to talk about sharing.

Now, another view was that he was somehow intellectually bad. Jay Forrester, for instance, who was of course, an expert, I remember him saying, well, when somebody makes a computer learn, he ought to sit down and think about the result before he makes another learn. He shouldn't be in a position. It's a bad habit to keep trying things. He should sit down and think.

Well, he changed his mind awfully quickly once he tried. I'll tell you later this story. So it was by no means. These are not inconsequential people. I want to mention the objective because it gives you some idea of the thinking of the time. Well, time sharing-- the first time sharing system was developed by none less than Corby with his associate in the MIT Computation Center.

And the first version was still on the 709 and was demonstrated in November 1961. Now, a much improved version improvement continued. This was for the 794 was in regular operation by the summer '63. But a lot of problem had to be solved. I believe that he could have handled something like 10 or 15 users at the time. There were a lot of problems to be solved.

Now, people have often asked what made time sharing possible at that time. And my answer is two things. First of all, you needed a transistorized computer. A vacuum tube computer-- they were just not up long enough to make time sharing worthwhile. It was simple as that. You have to have a transistorized computer.

The other thing was the fact that IBM came out with a very nice device called the disk file in about 1962, '63. And now let me show you what it was. That huge box was the 1301 disk file. It was a marvelous device. It had a stack of disks and arms that moved independent in and out to get on the tracks on the desk. I think the 1301 capacity was 9 million or 18 million words, something of that order of magnitude.

But that made time sharing worthwhile. Before that, mass storage was tapes. And we had a beautiful demonstration of the difference that a disk file meant. That means relatively quick access to programs and data in a mass storage device. That made a tremendous difference.

Now, another thing that was developed at that time was a fast ROM. That also was a big box. But it was essential for swapping programs in and out of memory. Now you know what time sharing is. So I won't try to describe the time sharing technique.

During that period, a lot of other time sharing projects got started. At MIT, there was a time sharing system on a PDP-1 that Jack Dennis-- where is Jack? Some place there-- I'll come back to that later-- developed. At Bolt, Beranek, and Newman there was another time sharing system on the PDP-1 There was a very serious time sharing effort at System Development Corporation in Santa Monica using the surplus computer of the state system. I don't remember what the number was.

AUDIENCE: Q32.

ROBERT FANO: Q32. I thought it was something like that. Then very interesting and there was a lot of consequences. That system did not have a disk file. And you could tell the difference between CTSS, the Compatible Time Sharing System that Corby developed and the system of SDS. Because of that, everything was stored on drums or tapes. And it takes forever to get the data [INAUDIBLE].

Now, the other very important time sharing actually was at Dartmouth College really spurred by Professor Kemeny who eventually became president of Dartmouth College and the creator of BASIC. That was a little different time sharing system in the sense that he was a one-language time sharing system. You could only program and use BASIC. It was built essentially around a basic interpreter while the system that I would talk about were multi-language-- were really truly general purpose time sharing system. And a lot of orders I want to get into that.

Now, at that time, that's the time when Project MAC started. And I want to tell you how it got started. Licklider, about whom I spoke before, in 1962 became director of the Information Processes Branch of the Advanced Research Project Agency or the Department of Defense. Now, he decided that he was going to start a program. He used the term centers of excellence in computing in several universities to push man-machine interaction through time sharing. That was the goal.

While he started running around the country time to gin up and buildup interest in doing that. And, of course, he came to MIT. And he wanted very much to have MIT start a big research effort on that basis. The trouble was that the people that knew anything about computers were kind of young. He saw the picture of Corby.

Professor Morris, who was director of the Computation Center, had already too many fingers in too many pies. And he didn't want to stick his finger in another pie. So nothing was happening. That's where I got in. I felt very strongly that MIT had to have a big research effort in the computer field. And I couldn't see anybody ready to do it. And I hated the idea of managing anything in those days and still do.

But I kind of got caught by Licklider's what I call contagious enthusiasm. So what happened is that there was a meeting in Virginia in which I spent quite a bit of time talking with Licklider. As a matter of fact, there was a long, long train ride back to Washington. That's when I talked with him was the day before Thanksgiving. I want to tell you the story because it shows you something about MIT. Things can go very fast at MIT sometimes.

The next day was Thanksgiving. And I kept thinking about this. And I said, oh hell, I'll dive in and close my eyes. So I decided to start what became Project MAC on Thanksgiving Day. The next day, I had a date with the provost, who at that time was Charlie Townes, the Nobel Prize-- the laser guy.

So after talking about the other business, I said, hey, I got this crazy idea. What do you think about it? Does it make sense to you? He said, oh, go right ahead. It makes sense. So over the weekend, I wrote a two-page memorandum. The next Tuesday, I had a date with the president, who was at that time Julius Stratton.

I told him what I had in mind. And he said, where are you going to do it? Well, as Professor Morris said at that time, MIT had been caught with its building down. Quote-- end quote-- there was no place. But by that time, I heard a story-- Technology Square was just going up at that time. And the top two floors of the building had been leased by CEIR, which was a computer service company.

They intended to put a Stretch computer on the ninth floor. That was the dream in those days-- the most powerful machine in the world. But by then, business was not going too well. And they were interested in getting a read of their lease commitment. That was the eighth and ninth floor. So I jumped in. I knew already about it. So I told Stratton, yeah, the eighth and ninth floor looked like they may be available. Fine, I said.

Well, on Thursday, Licklider happened to be at MIT. So we all gathered in Stratton's offices-- shook hands. And that was it. Then, of course, you had to write a proposal. But everything was decided within a week, really. There was a commitment of all those people within a week. That's kind of unusual, but it does happen.

Well, we wrote a proposal by January 14, I believe. If you want to see it, here it is, was presented and approved in principle and were given the authorization to start spending money by March 1st. Well, I'd been planning. So I had to hire somebody was going to be assistant director for administration. And they showed up on March 1st.

So I told him, you go around the Institute. Get your keys. Get a parking permit. Get everything you need. So he showed back in my office shortly thereafter, he said, they won't give me a parking permit. They asked me what I was going to work for. And there is no name.

So I realized that either I had to create a name or, otherwise, have the project go by the name that has been used. When I wrote that memorandum as a matter of courtesy, of course, I gave that memorandum to the dean of engineering, who was Gordon Brown at that time, and not much later, I went to see him about this. And he told his secretary to get the memo out. You know what he told her to do to file under FF-- Fano's Folly.

So it either was going to be called Fano's Folly or I had to think of another name. That's how the project went. We did in a day-- Project MAC-- the name was coined. Now, MAC stood originally for two had two meanings, one the goal, Machine-Aided Cognition and the other the tool to achieve the goal-- Multiple Axis Computer. Now, of course, a lot of meanings were coined. One that was coined pretty shortly was and came from the West Coast, obviously, was More Assets to Cambridge.

Now, by October 1, 1963, a copy of CTSS as it existed in the Computation Center was installed in operation under Project MAC. And that becomes the basis on what was later called the MAC system. It was originally just plain a copy of CTSS and then evolved with time. Now, I won't go through all the evolution. By the end, it could support something like 30 users.

It rented for about $62,000 a month with a 60% educational discount, which meant it rented for something $155,000 a month. Now, let me show you what it looked right. I haven't done things right. I should have switched these papers. Oh, up instead of down.

Now, an interesting thing-- it has two banks of memory. Each one had 32,000 36-bit words. One was used for the system supervisor. And the other was available to the user. Now, you see there is a lot of data channeled. There was disk file there. There were drums to speed up the swapping between core memory in and out of core memory. There was also a punch card reader to accommodate the past and magnetic tapes.

There were two graphical displays. And as a matter of fact, there was more than that. And I'll come to that in a moment. And then there was a transmission control unit called 7750. It was a huge thing. There was one person that knew how to program it. And I still remember his bald head. Terribly difficult to program, but it did the job beautifully when it worked.

And that connected to the MIT, a private branch exchange, so that we could dial in from all over the place into the computer at 30 lines eventually going through the private branch exchange. We also connected to the Telex international network operated by Western Union and TWX national network operated by AT&T. Now I've talked about the computer and the operating system. But remember, to have good time share, you need terminals and you need communications.

And let me talk a little bit about it because it was quite interesting. Originally, we used something that was called the Model 35 teletype, which was used by the TWX AT&T system for telegraphy. Now, what you see there sitting-- and that's the Model 35 teletype. And you see Dick Mills, who was assistant director. He was a graduate of this department that got a master's in management.

After being in Project MAC, he became head of Information Processing Services at MIT. And then he left for a big bank job in New York. And he is now a private consultant. Next to him is Oliver Selfridge, who was for a couple of years a social director of Project MAC. He was interested in artificial intelligence. He was at Lincoln Laboratory. And it was loaned to Project MAC to serve as a social director.

Now, those are the Model 35 type. There were other people. Some of you see there is a strange guy over there sitting on a Model 35. And that happened to be Moses without a beard. I don't know whether you were still a graduate student or just an assistant professor-- graduate student still.

I stole it from your book. So you ought to know. So there were all sorts of people in those terminals. [INAUDIBLE] all right. Now, I said that we were connected to the Telex that [INAUDIBLE]. And in April '63, shortly after Project MAC was started, we had the first, I believe, operation of a time sharing system across the Atlantic. And this is a historical document. So let me show it.

Sitting at the other end was Professor Maurice Wilkes. Now, Wilkes is the man who built at Cambridge the first stored program computer. He didn't invent the idea. But he built it first. He retired from Cambridge when he reached the critical age of 65, which was a few years ago and is now a part-time professor in the department. I hope that some of you had met him. He is called an adjunct professor. The adjunct simply means part-time.

Now, this is what it looked like. This is actual from the print out that came. I don't know whether this was the printout here or the printout in England. I am not sure. But that's what it did. It gave a demonstration. And then at the end, this was at the meeting of the British Computer Society and was an enthusiastic reception. The president ran the program himself and was all excited, all psyched up, particularly because there was an electrical engineer in the program designing filters.

And this is the end. And you see the message. Oh, this is absolutely magnificent-- Edinburgh Computer Conference. So it was a great show.

Now, you remember those displays. Well, there was various things connected there. One was the PDP-1. And look who is sitting in at the PDP-1, nobody less than Marvin Minsky. That PDP-1 was connected to the computer. Accidentally, I'm on the side there. But that doesn't matter.

There was another terminal. Let me explain a little bit here. You see me there without glasses because the TV didn't like glasses. But what you have on the light is a very, very powerful display terminal for computer-aided design, very powerful in those days, which was affectionately known as the kluge. John Watt, sitting right over there, was in charge of that whole development. Now, the kluge was a really wonderful machine.

You see that globe on the side there? It's hard to see. You could turn it just like a joystick. And what it did if you defined in the computer a three-dimensional object would make that object rotate. That is it had hardware to make the transformational coordinates so that you saw the object rotate in front of you. And this was 1963.

Now, one of the people that used quite effectively this device was Professor Cy Leventhal in the Biology Department who just one day came over to see me and wanted to have some advice. And I said, well, we'll talk about it later. I had something in mind. Let me show you around. So I took him here. And as I expected, he caught fire-- real fire. I was going skiing the next day.

And I had to make all the arrangements so that it could start the next day. This was just before Christmas. What he saw was the opportunity to study complex molecules in three dimensions. And he did a lot of work. He left for Columbia, unfortunately, because he's New Yorker. And he had enough. And for Columbia, had to go there. But he kept doing this sort of work.

Now, let me get back a bit to the communication problem. See the Model 35 teletype had only capitals. Everything in the computer field was only capitals. The big printer were all capitals while we wanted lowercase. Well, fortunately, there was a little group of people in the Cambridge area that had formed something that was called the reactive typewriter circle.

Now, I don't know why they call it was a man by the name of Calvin Moore who was a computer type. And then there was another man by the name Stewart Ferguson, who was a computer type. And they were beating the drums for lowercase and for other things. So among other things, they arranged for a visit to the Teletype Corporation in Skokie, Illinois.

And I asked the chief engineer about getting lowercase in it. He said, you know, we made recently a survey of people whether they wanted lower case. It was negative. They didn't want lowercase. Of course, if you ask anybody what he would do with something he never had before, he can't think about it. There are very few people can envision what they can do with a new tool. So eventually Teletype produced the Model 37 that had also lower case.

But in the meantime, the model IBM came to help. IBM had gotten not long before the Selectric typewriter, you know, the golf ball machine? So it was not very hard to put some electronics and making it into a computer terminal. And that's what they did. It was called at 1050. And that was really a major tool. But let me get back to communication. When we first approached New England Telephone about connecting a computer to the switch network, you can imagine what they said. It's impossible. It's never been done.

In fact, that's what they said. Fortunately, we got to some very helpful people in AT&T. And they really were very helpful in organizing things. One of the problems was that the telephone business, particularly in those days, was highly regulated. And you couldn't do just what you wanted. So they found a way of getting around. And we started using the WX network. I gave a demonstration from all over the place. And they always worked except at Brandeis where they had the wrong character set. I don't know why.

Now, I have to tell you a story. And part of me, if it's just slightly sexy-- sexist, I should say, not sexy. Well, a person from AT&T that was helping us a great deal was a fellow by the name of Earl Vaughn, who was pretty high in AT&T-- long line. And I remember-- I spoke at one of the meeting of the active typewriter circle. And he and my assistant director, Dick Mill, were sitting in the front row.

In those days, I made speeches are about the marriage of computers and communication because that's what we were trying to do-- marry computer and communication. And Earl Vaughn I understand turned to Dick Mills who was sitting next to him and said, yeah, sure, a marriage of computer and communication. But who is going to be the groom and who is going to be the bride? Of course, that question has never been settled.

It will never be settled because the rules of the game have been changed now. All right, now, let me get to something in a sense, more serious. Now, a goal of Project MAC was to create an online community and somewhat simulate a computing utility. That was really the goal. Well, within six months, we had achieved that goal. That was incredible.

By Spring '64 there was something like 100 terminals around MIT, some 200 users from 10 different departments. All sorts of things were happening. One thing that I am very sorry that I never did was to hire real smart psychologists, social psychologists, to really see what was happening, because the phenomenon that were happening were very, very interesting from a sociological and psychological stand.

Everything happened from people making friendship to the computer to keyboards smashed by fist. All sorts of things happened. It was extremely, extremely interesting. Now, you also had regularly users from other universities coming in from the teletype network. Now, let me show you a paragraph.

Oh, there's a picture that I didn't show that is going to mess up the secrets. But I'll show it in a moment. This is a paragraph from a report that I wrote, a paper that I wrote in the Spring of '64, just about '64. "It has becoming increasingly clear that the system's ability to provide the equivalent accessibility of a private computer is a secondary, although necessary characteristic. What users find most helpful is the fact that the system places literally at their fingertips a great variety of services. That was the important thing that became clear immediately. The system user themselves are beginning to contribute to the system in a substantial way by publishing their work in the form of new commands"

Now, this was an amazing thing. Before that time, nobody ever wrote a program and expected anybody else to use it. It was too awkward. Suddenly, it became easy. And people started looking at a program as a publication and make it specifically try to design it so that they could it be used by other people. It was a just a phenomenon that occurred very quickly and was very important.

Now, "As a matter of fact, an editorial board is being established to review such work and formally approve its inclusion in the system." We established an editorial board and worked very hard in it. "Thus the system is beginning to become the repository of the procedural and data knowledge of the community that it serves."

This last sentence is indeed the purpose of time sharing. That is it. It's a communication system between users. That's the primary function. The process is almost secondary. That's the point I wanted to make very, very strong. Now, lots of things were developed. I want to mention the particular one. And let me show you a picture. I will have to skip a couple of them. What is this?

Oh, yeah, all right. The fellow on the side is Professor Jerry Saltzer. He was a graduate student. And when he was a graduate student, he wrote two very important programs called Typeset and RUNOFF. Typeset was the first context editor ever written. It was fantastic. And RUNOFF was a primordial scribe that was quite capable already. In fact, we used that system to write all our reports for a lot of other purposes. I even used Typeset for accounting purposes. I wouldn't want to tell you how. It was very powerful in many ways.

Now, it's interesting how he wrote it. All of you who have done any writing or any work that really requires a lot of concentration know that when at some point you say, I got to do it. I got to do it. I got to do it. Then you pick out papers, a pencil, and you go and sharpen one by one, trying to delay the time when you really have to think. I've done that. I know. Jerry Saltzer wrote Typeset and RUNOFF in that spirit. He had to write his thesis proposal.

[LAUGHTER]

Now, this is the real history. It's a fact. Now, let me show you something else. Time is flying. But let me show you something else. Oh, no, I want this.

Now, there was this user community around the computer. And in a paper that Corby and I wrote for "Scientific American," I tried to make a sketch of the user's view of this marvellous time sharing system. Do you see what I see? That's a network of a personal computers.

They are called pseudo-processor and psuedo-memory. Take the pseudos out. They have a file server, of course. They have a message center. They have also some services. What Demon was like an automatic user. What it did, when anybody logged out, they went to look at the file. If anything has changed, it was copying it in the background. It was copying it and backed up automatically.

[INAUDIBLE] was Foreground Initiated Background. What it did from a typewriter-- you could run a program with the computer at time, and put it in my file, and I'll look at it tomorrow. I combined typed, online use with batch plus. This was more traditional form. No, Demon was that? What do I have background?

AUDIENCE: Background is a traditional background.

ROBERT FANO: Was a traditional background, yeah, where you just ran the program when the computer had time. Now, I was surprised when I saw that. It is as the geometry of a network of personal computers, as far as I know, none are with those features in operation today. I'll come back to that in a minute, although I understand from Professor Saltzer that indeed that sort of a goal he has for [INAUDIBLE].

Now, let me move on. Very early in Project MAC we started planning for the next system. It was quite clear it was unbelievable what you could do with your 7094. But that machine was certainly never designed for time share. So why not get a new one? So we started talking with manufacturers and so forth to find the right machine.

Well, during one of the visits at UNIVAC, you see some strange character over there, Professor Dennis, stand up so they can see you in the corner. There's Corby, whom you know, and that is Arnold Cohen from UNIVAC. We invested quite a bit of time in looking around. We also did a lot of thinking.

And from our experience with CTSS, it became quite clear what the general organization of a computer utility ought to be. First of all, let me give you a sketch. This is what is in the program that was given to you. There were several ideas first of all is a memory centralized system. This, incidentally, was also in the 1965 "Scientific American" article.

That's where it comes from. It's a memory center system, all the communications was through the memory. That was a very important view that had come out of experience. That's how the user viewed the system as the memory was the process poking into it. That was important. Also as you see, there are pools, more than one, of every single box. There are two processors, four banks of memory, two drum controllers, two disk controllers, two general controls. Then it was a pool.

And the idea was that if anyone went back, well, you can run it. This was, again, the idea of a public utility. You don't stop to fix things. You keep running. Now, as a matter of fact, we established certain goals. And I want to put them on the slide so you can see what they are. Continuous operations-- that's public utility-- pools of identical units. We wanted that reconfigure the system online without stopping.

Communication to memory-- this was a technical point. You want a clean communication system. Now, the hierarchy or memories with automatic migration of programs. These are all things that you see now but didn't exist in those days. Now, to save time, I want to go to this point-- execution of unbound modules. This is what is often called segmentation-- or the two-dimensional addressing.

Now, that idea if you really look back came up in conjunction with the design of the B-5000 machine way, way back. But as far as I know, the person who really understood the implication of this was Jack Dennis, as I mentioned before. As a matter in fact, I remember vividly a scene that took place toward late spring or early Summer '63. Jack was giving a talk on program segmentation for an [INAUDIBLE] program. And I was sitting next to Corby in the front row. I was the chairman of the affair.

And at a certain point Corby started getting all excited-- yes, yes, that's right. What are you doing? Yes, yes, yes! Really got excited-- that's how segmentation got into the Multics system. Sometime communication takes place in strange occurrences.

I think some of you may not know what a segment of virtual memory is. It's a great idea. I believe so. Segments are distinct procedural data modules. They are the basic unit of computation. Segments are files. There is no distinction between segment and file. A file is a segment. A segment is a file. It stores individually.

You are using essentially two-dimensional addresses. That is, one dimension is the name of the segment. The second dimension is the location, just like a book. If you want to make a reference to a book, you give the name of the book and the page in the book, location in the book. It seems very straightforward. Then segments are divided into pages, just like a book. That's the paging mechanism that you are familiar with.

Now, paging is not seen by the user. It just goes on automatically in the machine. But the users are very conscious of segment. They are a name that they know and they use. The very important thing is that segments are bound to one another in a computation and execution time. You don't have to pre-prepare anything. You don't have to load programs together. It's a separate thing.

Now, the analogies that I often use is if you read a book, it makes reference toward books to paper. What is the traditional way a loading program would correspond that you give a book, if you have a book, you give it to someone to Xerox all the paper and all the pages and everything that the book refers to. Then they are all assembled together. And the references are changed to make reference to the various things. And then you read it. That's the normal loading. This is what we actually do with books and papers.

Now, there are lots of other things that I don't want to get into. But that putting paging segmentation together was a real difficult goal. Well, the Multics Project-- we talked with all sorts of manufacturers. Eventually, three companies-- IBM, DEC, and General Electric made a proposal. It turned out that the architecture of new 360 system and the architecture of the PDP-6 just plain were not suitable for our goal. We wanted a memory-centered system. And those were processor-centered system.

So we went with General Electric. It was a memory-centered system. But a lot of hardware still had to be designed to get what we wanted. And I wanted to mention another name, the name of Ted Glaser, who was a member of our faculty at that time. He was a superb designer. Incidentally, he is blind. But he had overcompensated, I tell you, really overcompensated for being blind.

He had a memory that was fantastic and an ability to find-- I went with and visit the company. And he was, the way said, he was looking at a mechanical device-- somebody open it and just went out where they said, ah hah, ah hah, oh, that's what you do. He had understood the operation of that machine-- just incredible. Well, Ted Glaser was a major contributor to what turned out to be the Multics system.

Unfortunately, he was not here to the end. And he got offered a chairmanship of a computer science department. So he left. Well, the Multics project, which had those ambitious goals, I don't know when they started. I would say '64 we decided to get that equipment. In 1965, we had the four joint computer conferences series of paper on the design of the system represented.

The same for the hardware was frozen to make a longer story short, we finally got into operation in November '69, two years later. But you have no idea what a project it was. Everything had to be done from scratch. There were no software tools. When the machine arrived, it had to be debugged. A compiler had to be written for PL1 because it had to be written all in PL1. It was a major undertaking.

It was completed I say, quite honestly, because of Corby. He is the only leader in the team involved with three organizations-- Project MAC, the Bell Telephone Laboratories, General Electric, and later Honeywell-- was the only leader of that effort that was there at the beginning. And he was there at the end.

I want to tell you something else about Corby because it's important. When people design complicated things, they say now, they scratch their head and say, now, suppose if this happens, what should I do? Corby never does that. He is a firm believer in Murphy's law. Everything that could go wrong will go wrong. So he always says when this happens, what will I do? And that is the secret of the time sharing system that he has developed. He always knows what to do.

Now, let me make a couple of concluding remarks because time is really flying. We are going through a microprocessor revolution. Now, essentially microprocessors have invalidated what used to be called Grosch's law. Now, Grosch's law way back said that you always get a bigger bang for the buck with a bigger machine. That's what he said.

And that was to a good part, the rationale for time sharing a large machine rather than having separate computers. Right? That was the rationale. Now, clearly Grosch's law is no longer valid. So this is just changing the entire technological environment in which time sharing grew up. But the technology of implementation may be different, but the goal is still valid-- communication between people, sharing, online computation, that is still valid. But you will probably want to implement in the future. In a different way, you may want to have the computational power at the terminal [INAUDIBLE]. But you still need mass memory. You still need communication.

Finally, I think I ought to give you my view on the successes and failures of the time sharing movement and our effort at MIT. I think from a research point of view, I think the time sharing work at MIT was a major, major contribution in many ways. The variety of techniques that were developed just went out in the field. And you find them today all over the place. Virtual memory in some sense was started in those time sharing systems. Now everybody uses virtual memory. A lot of technique went out in the arena to use.

It certainly online research in a university sense, and it did at MIT. There was really a sudden growth of computer-based research at MIT in the '60s. There is no question about it. It did provide people with physical access to a computer from wherever they might be. Certainly, it did that. It failed to provide the corresponding intellectual access from wherever people might be.

The reason is that when you work on this system, you always are in a learning mode. And it makes a great deal of difference whether sitting in this next office is somebody that has used the program that you're going to use next or you will isolate then miles away from anybody that knows what you're trying to do.

While at one point, we tried to do something about that, but it never came out. So the intellectual axis is still a problem today. While it was a successful research community at MIT online community, I think that the movement failed to create online communities, certainly outside the university and even within universities. That is, for instance, we didn't have an online student community at MIT. We are just beginning now to have some. It was just [INAUDIBLE]. The major reason is cost has been too expensive until now. We're just getting now to the time in which it's cheap enough to do it.

But there is another reason if you go outside the university, you see the kind of time-sharing system that I've been talking about are really designed with one objective in mind, to create decentralized communication and control. That is, it's a decentralizing system. It's inherently so. That was the intent. And then, of course, that runs against the grain of traditional management.

That also is quite clear. And I can give you an example. Now, the situation I believe is changing. Let me give you an example. Do you remember there was the celebration of the telephone a few years ago? Among other things, there was a series of seminars with speakers coming from all over the place organized by the late [INAUDIBLE] on the sociology of telephony.

Now, you are aware that in this country, telephony has been for the public from the beginning because that was Bell's idea. He had a choice. But that's what he decided. It was very interesting that a French sociologist came over to speak. And I never realized before that the situation in France was and is still very different.

Telephony was used as a tool of government and of the military. The reason was, and it remained that way, in fact, still while 10 years ago, I can't say today. I tried many times before Paris from Switzerland. And the communication was lousy. The telephone system was behind other European countries and certainly the United States because it was government policy not to push it. France is run in a very centralized way from Paris. And the government never wanted to provide the tools to bypass Paris.

That comes from a French social psychologist. I'm not inventing this. You see similar things relative to time sharing systems in organizations. I think the situation is changing because there is a tremendous pressure now on individual productivity, improving individual productivity. And this what online user computer does for you.

The second is that the cost of computers is going way down. So I think that we are going to see a change. The technology will lead to different types of systems. But the goal will be the same. Well, I talked too long. Thank you.

[APPLAUSE]