# Computer Oral History Collection, 1969-1973, 1977

**Interviewee:** Rex Rice
**Interviewers:** Robina Mapstone and Henry Tropp
**Date:** October, 10 1972
**Repository:** Archives Center, National Museum of American History

**MAPSTONE:**

It's October tenth and we're talking to Rex Rice, Fairchild Camera and Instruments Company, Palo Alto, California. A good way for us to get going is for you to tell us how you got into, how you started in the business.

**RICE:**

I was in the Structures Division at Northrop Aircraft, starting in '46, and in those days the engineering computing was a room full of girls banging on Friden mechanical calculators. Fortunately, with regard to what I'll call the engineering side of computing, we had the missile emphasis there at Northrop, which was a separate division, and at the time we were using IBM equipment for engineering computing. That would be (forgive me) card punches, sorters, collators, tabulators, and the very early step-'n-fetch it multiplier--what was that, the 601? The 601. Went over and grabbed a card and flipped it around. Call it step-'n-fetch it. At the same time in the financial end at Northrop they were using Remington Rand equipment, which was card equipment, for accounting and financial analysis.

The missile guidance group--I'm sure you've been in contact with some of these people, Jerry, for one--were really the impetus to the development cycle that started at Northrop, and the role that the engineering computing group played was to get, let's say, the IBM-oriented or later the BINAC equipment that went into the operation of the calculations for the missile. However, all the special purpose equipment which the missile group developed --which later came to be the [?] and equipment of that nature--was directly designed and built by people at Northrop, went into the guidance equipment. So the area that I'm familiar with was, if you will, the engineering side, as a support to the missile and also a support to the aircraft structures calculation. Two people who are rather key in that operation are Gregory Toben, who is still with IBM, lives here in Los Alos Hills, and Bill Woodbury. It turns out they were running the... let's see, there was one other contract that was going on at the time... Lee Ollinger had come from AEC operations and they had some AEC shielding calculations going as a separate contract independent of the missile system. As it turned out, Tobin and Woodbury are really the guys that got it started at Northrop. They had the IBM equipment and the job of doing all the computing and this was while the missile people were still thinking of computers and specifying them and so forth. They had the actual job of running it on card punch

equipment and probably Woodbury is the guy that got tired of carrying cards from step-'n-fetch it over to the tabulator and back, and he got the idea of connecting the two together. So, I don't remember the exact date, but we came in one morning and they had some wires hanging between step-'n-fetch it and the 405, and, sure enough, it would feed a few cards through and bang away on the multiplier and come back... That gave them the idea that there was no reason you couldn't connect all this together. At some time along about that time period, the installation acquired a tube multiplier... I forget the number of that... 603, I guess (it's been a long time ago)... acquired a 603 multiplier, which, as I remember, had two six-digit inputs and came out with a twelve-digit product, and that's all it did. Well, Woodbury was convinced from his connection of step-'n-fetch it to the tabulator that there was no reason you couldn't connect this thing together and quit carrying cards around and get the full multiplication speeds of the 603. So, through the emphasis that the missile people had with the Defense agency, they were able to convince IBM that it was in fact a real thing to do. Something like six weeks later, they shipped the 405 and the 603 back; six weeks later they arrived back. That was the first of the Card Programmed Calculators. If you're interested in some philosophy that evolved from that I'll trace it for you a little later...

**TROPP:**

I'm very much interested because I talked to Tobin and got into some of the technical aspects, but the philosophical aspects, the impacts, the impact and the way it affected people's thinking is the area that we are most interested in exploring.

**RICE:**

Well, you now know the world of microprogramming. That was the first micro programmed--really, as far as I'm concerned--production system, and if you think of the concept of a plug board--not a bunch of wires as the programmers think of it--but as a fixed program that one has set in the controls of the system and you had the ability in those days to change it. For example... well, to back up a little bit, I think Woodbury did most of the creative thinking but I want to give Greg Toben credit for starting a lot of us in computing. Tobin had the idea that if you didn't know what was in a computer, you shouldn't use it, and, much to IBM's chagrin, he had the customer's engineering diagrams pasted all around the wall.

**MAPSTONE:**

We saw a photograph yesterday of that.

**RICE:**

Right, and when I came down from the structures group, we had some very tough structural calculations to make. We had these girls banging away and had some spare time. It was just about the time they had accomplished the connection of the 603 with the 405, and just about the time they were starting to do the calculations for the AEC

shielding, so I was fortunate to hit when the customer engineering diagrams were pasted on the wall; and what they taught you was that a computer is a pot full of parts to do what you want with. In contrast, you know, it's all locked up and just put FORTRAN in it. I think that was very beneficial. So I entered the picture at the point where Toben and Woodbury had just gotten the 603 connected with the 405, and we literally took the aircraft structure problems that were there and transferred them from manual calculations into a systematic way, and got them on the Old Betsy, I guess we called it.

**TROPP:**

These are problems primarily involving stress and strain.

**RICE:**

Aerodynamic parameters, stress, strain, flutter calculations, structural calculations. Now, at the same time, on the same machine, we were running all the guidance calculations because there was no production computer available.

**TROPP:**

The guidance calculations in terms of predicting fixed star positions...

**RICE:**

Star positions, all the astronomical calculations, the detailed engineering calculations of the equipment they are designing. So in a sense, for quite a period of time Old Betsy was doing all of Northrop's calculations, both scientific and engineering, as well as starting the shielding calculations. After much pressure by Northrop on IBM, the Card Programmed Calculator evolved and the first public showing of that, as I recall it, was 1949... is that right?...when they announced it at Endicott. They called everybody in. At that point they had gotten the 603 out and put in the 604, which was a much more sophisticated system and a little programming of its own. Now at the same time period, the concepts which led to NAC were evolving in the missile group. It was a separate effort to get a very, very high speed calculator. So in a sense, Northrop was pushing computing on several fronts, one of which went the digital differential analyzer route, another of which actually got BINAC going, which is how we happened to tie in with Eckert and Mauchly and the third front was pushing IBM into what was the Card Programmed Calculator area. Am I covering the kinds of things you want?

**MAPSTONE:**

Very well, yes.

**TROPP:**

In terms of the Card Programmed Calculator, I guess I'm interested in as many

viewpoints as possible on how much IBM had to be dragged into this. Now they obviously made that first stuff up very quickly, the six-week period that you talk about.

**RICE:**

They were dragged into it reluctantly. Being diplomatic and an ex-IBM employee, I'll say it...they did not yet believe that you should tie all this together and make a calculators of it. Now, when I say they, IBM even then was a big company. There were those in the company that did believe it and there were those that don't, but the policy-making at that time had not yet really realized what a boon that was. It was really the pressure from Toben, Woodbury, and J. K. Northrop with the missile guidance problem that pushed them into it. Now as soon as IBM did that they immediately saw the advantages and then they converted and came out with the Card Programmed Calculator.

**TROPP:**

There's another interesting aspect. You described your engineering, your structural division with a group of girls punching mechanical calculating machines, and I guess I'm assuming that every division had its own ability to do whatever computation it needed more or less depending on their level of computation. Now with the advent of something like card program calculator, even at its early... just hooking them together with wires... it seems like that... change...

**RICE:**

That exerted... that was where the change occurred, right there. In fact you could feel it.

**MAPSTONE:**

[?]

**RICE:**

When you were there you could feel it happening. First of all, the accuracy that one could get from a mechanical, or electronic combination with mechanical, in terms of not having errors, was immediately apparent. Well, as it turned out, I was complaining about the calculations in the structures area because I had done some research and had a new method of wing structures analysis that needed a lot of computation. The boss said, "OK, the girls are yours, organize them," and I got real tired of that in a hurry. Just sheer luck that Toben and Woodbury had the machine downstairs, so I went down to find out what we could use in their spare time, and going back to my point earlier, Toben said, "Before you can use the machine you've got to learn what's in it," so we took the course by walking around the wall and looking at CE diagrams, and learned what relays did and how the plug board worked, and how things worked out.

**TROPP:**

And when you had a problem...

**RICE:**

And then we in those days programmed--we actually literally plugged the program in the plug board. That was a very, very helpful thing because if you want to think of it, we took what was a general purpose machine and created a special purpose... I think of it philosophically as a special purpose language, which is what the plug board was, to do that job. So, we didn't have to take a job that... As I look at computing now with all of the operating systems... In fact I've just been back looking at a '67 installation, and instead of dropping a plug board in, it takes something like five to ten minutes, lights flashing like crazy and millions of operations going on just to get all the resources set up, and now it's ready to ask you, "What do you want to do on the terminal?" The plug board, if you want to think of it that way, was a special purpose language you dropped in to do specifically what you wanted. So, we literally then had plug boards around Northrop for each of the special purposes, and Toben saw to it that the people programming them were trained to do that. Surprisingly enough even in those days, although the rate at which we computed was archaic, the amount of programming you could do with a plug board and get it debugged was, I still think, equal to or better than some of the high level languages. It's rather amazing what you can do with one, and we proved that quite a few times. We had IBM representatives come by, learn the machine in a day or two, and set up plug boards and go. You can't really program problems that complex now in FORTRAN any faster. Well, to make a long story short, we started putting aerodynamic, structural, and missile guidance calculations on Old Betsy. A sort of a challenge grew among those using Old Betsy as to how much hair you could hang on a plug board, and we had relays and all sorts of things. The wiring was about that thick. I don't remember the exact date, but sometime shortly after the card program, calculators were in production at IBM, the other stream of computing was coming to the front, and the 701 Defense calculator was thought of by IBM. At the time that was initiated, as I recall it, it was Cuthbert Hurd was at IBM in the effort, I guess both engineering and marketing... or at least marketing was where I saw him... I'm pretty sure he was in engineering... he came out to convince everybody that the 701 was the way to go. Those of us at Northrop who had had the experience with what I would call special purpose languages--that is to say, plug boards--felt that that was not a good way to go in the sense of programming, but it was an excellent way to go in the sense of getting more computation rates. The rates were bothering us and all we could do at that point was multiply, and I guess it wasn't until we got the full card program calculator going, the 604, that we had division and other things. Well, I think the fondest recollection I have of those days on Old Betsy was... if you'll picture now, what did we have on one of those... a hundred positions of digital storage in the tabulator... as I remember it's a hundred mechanical positions, something like fifty line a minute printer...

**TROPP:**

That sounds possible. It may have been a hundred.

**MAPSTONE:**

I think it was as fast as a hundred.

**RICE:**

It was a hundred numeric and fifty alphabetic, because the type bars came up twice as far, or something. Let me give you a picture of what one of the most fascinating jobs ever done on the machine was. The atomic energy shielding calculations that they were doing for material research were actually being computed as late as, I think, '53 or '54 on Old Betsy. There was a plug board in there with about twelve inches of wire and fifteen or twenty relays hanging out, and some other stuff around it. That machine--remember, now, it had a 6x6 multiplier, 100 positions of counter, and a plug board which we will call a special purpose language, if you like. That machine was calculating one collision every time the type bars came up. Not only were they doing that, but Woodbury had set an oscilloscope up on top and you could see the collision and the direction it went. Can you picture doing that kind of a calculation even now on a machine with only a hundred positions of storage other than the card file? They were literally getting a collision fifty times a minute, which was the limit of the machine.

**TROPP:**

That's unbelievable. Toben talked about that, and he talked about this random number of (rads) that they used in order to get this variety of motion effect.

**RICE:**

They actually had that thing going.

**TROPP:**

Then they had a view, as you say, the view scope where they could look at it and watch the particles until the explosion occurred.

**RICE:**

Then you'd see the direction the next one took off. I think even nowadays if you tell people you're doing that on a tabulator, they don't even know what a tabulator is anymore. A tabulator on the 604 is rather fantastic. Well, I guess you have a fairly good picture then of the card program calculator.

**TROPP:**

Well, except as I said, from the philosophical point of view, which is... everybody has their own viewpoint and we're interested in as many different views as possible in terms

of how it affected the thinking of people, You really went from one world--a world that had been in existence for a long, long time, where a computer consisted of a person...

**RICE:**

Yes.

**TROPP:**... to suddenly this very sophisticated machine by comparison. You also went from a world where things were compartmentalized, in that everybody had their own group of little computers, to a kind of a central place where you're always talking to people.

**RICE:**

And where all the work funneled in and a computer dictated the setup of formulation of the problem.

**TROPP:**

Kind of a natural open house situation as well, where everybody learned how to do their own problem, because for many kinds of problems you really couldn't trust somebody who didn't know your problem to do it, because, as you said, you only had so many digits to work with, and you had to know [?] you had--you couldn't do the sophisticated numerical analysis, the truncation problems, where some of you had to know about them independently.

**RICE:**

I can give you several philosophical ramblings from that beginning if you like. Let me first talk about people. In a sense Toben and Woodbury created the machine that we used and, as you point out, got us from people on Fridens to automated computing. We had to first learn the machine, and I think I have never yet gotten over the fact that the machine is a pot full of parts that should be organized to do what you want them to do. Now, I don't think... I guess ninety-nine percent of the computing works the other way around--you stuff the programming through what's there. A few of us still believe it's a pot full of parts that ought to be doing your work. I'll trace that for you in a moment. The first generation of us--that would be myself and some of the people helping me on structures--had to get down and learn the machine, wire the plug boards. We knew our engineering problems because we had set them up and formulated them. We could look at results coming out on the type bars way, way ahead of the time the problem was finished, and tell somebody had blown it on the inputs. We had a system there that took us about three days and I think something like fifteen separate passes of card decks into it; and we put checks and balances at the end of each step, and although you didn't know exactly what was wrong, you could feel things were going haywire. This number should have been negative and it's positive, or a balance didn't come out. So, back in those days at that rate of computations we were able to pull a job long before it went all the way

through, and get things fixed. And, of course, computing resources were very precious, as you can imagine, when you're only getting 50kc multiplication and about fifty cards a minute through the on put. So you're really worried about these things. Well, the first generation of us knew our problems cold, knew what the computer was doing cold, and essentially were in control. Shortly after we got that set up we brought more people in and this was true of several problems, types of problems like aerodynamic structures, missile problems, we would bring in then other people who were the beginnings of what we think of as programming people, these [?]. They may or may not in those days know the formulation of the problem they were computing. Generally they did; they came out of the aerodynamics department or the structures department. Generally, they had a feel for the problem and what irs parameters were, but they took it at face value that the plug board of the machine, which I consider the special purpose language, that the plug board was fixed. They didn't quite think of the machine so much as a pot full of parts. They would accept it as being the way it was. Now the second generation, which was this group of people, still had a very good feel for the problem and what was going on in it, and they would tell when somebody put in an incorrect input by looking at the outputs, and stop it early and really know how to go back and say, "Hey, that wing analysis is wrong," or, "the flutter problem didn't have the right parameters..." Within about two years we had expanded the use of the system and by this time we had the first official card program calculators that IBM had put out, so we had much more computing power than we did with Old Betsy. We brought in what I'll call the third group of people. Now these were people who more nearly simulate what has developed in the programming profession. These were people who would take a problem formulated by somebody, convert it to machine able language, take the runs back and look at it, and give them back to the person who had sent them in. They had lost two feels: they had lost the feel of machine as [?] because they never had it; and they had not formulated the initial problem, so they were essentially what we now see as the programming profession, where they accept an application from somebody, send it down to the computer room and get it back out the computer room window and give it back with a few checks and balance.

**TROPP:**

Essentially an open shop kind of operation?

**RICE:**

Well, more than that. On two years' calendar time we had seen the people doing the computing completely lose touch with the real world, which was the application. We have never turned around from that since. The only time it happens is where one particularly motivated individual will do all this himself, like programmers... he'll program his system, he'll learn the machine, and he'll program his application. We have had several instances. Those were people who work here for me. I think they're typical at universities. Wherever somebody comes up with a new language to solve some new problem, if you look a little deeper, you'll find some one guy who bridged these three techniques, three technologies. You could actually--this was between '48 and '51--you could actually feel that happening. In retrospect, if one were smart enough you could

predict a profession breakdown from that point. Now those people wanted nothing to do with wires. It got to be s dirty word putting plug board wires in. Right at that instant in time is where we lost the ability to at will special purpose languages we've never regained it today. Now you see things coming back that are pseudo-[?] but we've never really regained it to this day. Some of us who have been around that long said it was the fun went out of computing. That's not really true--it was the fun for us. But I think the key issue is the feel of what's going on in the machine as a machine, the feel of what's going on in the software as software, and the feel of what's going on in the application , was split into three different areas.

**TROPP:**

Let me just throw a particular name at you just to get a feel of some of the people I've talked to and where they fir in this expansion level. I'm trying to figure out where Rogers Mills fits into this...

**RICE:**

He came at a much later...

**TROPP:**

He would be roughly at that third level?

**RICE:**

He would have come in at the third level, after the third level was apparent. Roger worked for me. OK, when Toben and Woodbury decided to join IBM--I guess Woodbury had gone back to study under John [?], Toben went to IBM, and then Toben and Woodbury went to IBM together and developed what was the Wooden Wheel, which was a natural outgrowth of Old Betsy. At this point they had a rather sophisticated plug board, set on the tabulator, and we came out with the system that was 797. Toben and Woodbury did that system at IBM and I was the one that came down from Structures and took over as Associate Director for the Computing Center. We picked up analog computers and at that time BINAC came in. The missile people never got it to work, so we inherited it; we never got it to work either, but it did advance computing in the sense of speeds and thinking. I was not convinced then and I'm still not convinced, having watched BINAC come in against the Card Programmed Calculator, so we picked the easiest way in the world to get problems on the computers with useful solutions. He had a very good contrast by about... when did BINAC arrive?

**TROPP:**

It would be about '49 or '50.

**RICE:**

No, it was a little later than that.

**TROPP:**

Then in 1950, possible as late as '51.

**RICE:**

'51, OK. BINAC came in. We still had the Card Programmed Calculators that IBM put out in production which by that time were doing most of the world's computing. The 701 was coming along at IBM. We all knew all about it--roughly what its specs were. I don't think any of them were delivered until what?...'52?

**TROPP:**

'52.

**RICE:**

And they weren't really in general use until '53. It was fairly obvious that with IBM's financial resources and the problems we were having with BINAC, that the other computers were going to surpass BINAC, although we struggled with it. But we did have a good contrast in the problems of getting a machine which is binarily programmed and not decimal. I would like to think of a plug board for you philosophically as having macrodes not microdes. What is a plug board? Well, you take something that already had decimal addition in it and you jack it up to do five or six things at once. Micro coding as we know it takes binary flip-flops and builds them up in a time sequence so maybe you do binary addition, so the tendency on a plug board machine is to go macro. From a level which I'll call add, subtract, multiply, divide, transfer, test, set--if you call that level one (or unity), a macrode puts those together to get operations which do then special purpose systems type operations. Micro coding takes the flip-flops and the gates and builds them up to where you get test or binary addition, so we were clear back in the age of '49 very, very macrocode conscious on the plug boards, to a degree they haven't met yet in a symbol computer we did here, in modern machines. IBM, correctly I think, decided that they had to go electronic. Who's to know--perhaps correctly or incorrectly--decided to follow the 701 route. The 795, which was the early machine that [?] and Northrop got and then later the three 797's which followed the macrocode group with a plug board looked very good, and we got, between '53 and, oh, '56 or '57, we got extremely respectable performance out of those computers. Remember they were 50kc machines, had no tape, only 100 words of storage, and did a fantastic amount of...

**TROPP:**

Eight digits per word?

**RICE:**

Pardon me?

**TROPP:**

Eight or nine?

**RICE:**

I'll have to dig back on that one.

**MAPSTONE:**

Wasn't it eight, because

**RICE:**

I think it was eight.

**MAPSTONE:**

some other people crossed up using nine digits.

**TROPP:**

Right, it was eight.

**RICE:**

Yes, it was eight. Well, in those days there was a debate, fixed versus floating decimal, and all sorts of things, how many digits one needed. I think people settled on ten because you've got ten fingers. It was an eight-digit machine with about 100 words internal storage. The reason that machine philosophically never progressed further is because IBM chose to kill it. Northrop wanted many tapes on it and they wanted more core storage and IBM's main thrust had gone off in another direction and they didn't want that program to continue, so we were never able to get bulk file on it and that financially killed it. So I think the hierarchy, then, what happened, sort of tapered off. We ended up with one fat Stanford for a while, and one at the University of Michigan.

**TROPP:**

There were only about half a dozen...

**RICE:**

There were three made.

**TROPP:**

Were there three?

**RICE:**

Three... The benefit those machines had to computing, aside from just running computations I think, was quite a few people, maybe on the order of 50 or 100, both at Northrop and the universities still believe that computers are a pot full of parts to be organized the way you would like them to be organized. If you'd like I'll trace one branch of that work that I ended up in, and later here at Fairchild, which ended up the only machine in existence that has the high level language directly implemented in the hardware, which I can trace clear back to what Toben told me--you know, organize the machine to do what you want. Well, that sort of covers just at a fast pace, Northrop. You've probably already traced the branch out of the missile group and followed the digital differential analyzer. Jerry Mendelsohn among others is a very good course of information.

**TROPP:**

We've talked with Jerry and to...

**MAPSTONE:**

I'm in the process of talking to Harold Sarkissian...

**TROPP:**

Don Eckdahl, Dick Sprague...

**RICE:**

Orville Eiseman is another one. He's at DataProducts if you need to talk to him about it.

**TROPP:**

Orville Eiseman? Data Products?

**RICE:**

Data Products, an organization here in Canoga Park.

**TROPP:**

By the way, I wanted to thank you for your efforts and your secretary's efforts in enabling us to locate Greg Toben.

**RICE:**

Oh, good.

**MAPSTONE:**

And through him, Woodbury.

**TROPP:**

And through him, Woodbury.

**RICE:**

Woodbury's a hard one to trace. I've lost track of him.

**TROPP:**

Well, there are two people--talking about tracing individuals--we have no line on Glenn Hagen.

**RICE:**

I've lost track of him too.

**TROPP:**

And only a partial line on Floyd Steele.

**RICE:**

There's a fellow named Charles Williams...

**TROPP:**

Charles Williams...

**MAPSTONE:**

I'd like to...

**TROPP:**

Charles Williams might be the key..

**MAPSTONE:**

I'd like to find Williams too.

**RICE:**

Charles Williams, the last I knew, lived on Via Larga Vista on Palos Verdes Estates, in the 4000 block.

**MAPSTONE:**

Can you give me that again just so I can write it?

**RICE:**

Via Larga: L-A-R-G-A V-I-S-T-A, Palos Verdes Estates in the 4000 block. Now he's probably moved, but you might be able to trace him from that address.

**TROPP:**

He was with Glenn Hagen through all of these...

**RICE:**

Williams would be a good way... he probably went over with Glenn. There's another one who wasn't actively directly involved in computer development but was administratively in charge: George Fen. George has had a consulting business in Palos Verdes Estates. He might be able to help you. He would know...

**MAPSTONE:**

Fenn?

**RICE:**

Fen--I think there's one N. He was administrative charge under Al Weaver.

**MAPSTONE:**

Oh, yes.

**TROPP:**

We haven't located Weaver either.

**RICE:**

Weaver you can find here in Palo Alto, California. What the dickens is Buck's first name?

**TROPP:**

He's always listed as E. L.

**RICE:**

His brother is right here in Palo Alto. I ran into him about two or three weeks ago... I always called him Buck, so I can't remember... Buck Weaver was in charge of the Structures Department, and he's the guy that said, "Those girls with Fridens are yours." This is Al's brother. I think you might try Weaver Associates Incorporated, Consulting Engineers, Palo Alto, 321-7210. They'll get him for you anyway.

**TROPP:**

One of the areas that I'd like to have you expound on philosophically and from an information point of view is this heavy West Coast environment. Northrop seems to have been somehow in a center position.

**RICE:**

It spawned an awful lot of things.

**TROPP:**

It spawned a lot of things and there was a lot of communication within the company itself. There also seems to have been a lot in terms of the whole environment of universities and the whole airplane industry. In looking at the West Coast development as a sort of isolated environment, which was to some extent...

**RICE:**

We used to think the center of gravity of computing was sort of about the east city boundary of Los Angeles. It was, for a good many years. That was the impetus of the aircraft industry, though.

**TROPP:**

Now what I'm interested in is the interrelationship. Now, for example, on this problem of shielding. They needed some kind of a random number endpoint. That came from Rand Corporation. Now the CPC originated at Northrop, and soon there were 200 of them

built and in use, and to many people the CPC was the only computational device that they were exposed to for a long, long time; and for that same number it was really their first exposure to what we now call computing.

**MAPSTONE:**

Do you mean there were 200 CPC's?

**TROPP:**

Yes.

**RICE:**

Yes. In fact, it was up until about 1955, if you took the number of computations made in the country, it was clear up about 1955 before the electronics began to exceed the number of computations per day that the CPC's were banging out.

**TROPP:**

Now this was a relatively small machine that rented for a reasonable amount.

**RICE:**

Very reasonable compared to today's rents.

**TROPP:**

And I guess, you know, there was an impact from them, and I guess what I'd like you to talk about is the kind of West Coast environment, and where the flow of ideas came from; how people affected each other in types of them...There are so many new problems can be gotten--I guess I can say this over and over again--but until you have the equipment you'll be faced with the problems that can be gotten by computing devices.

**RICE:**

Well, there was a--as you can imagine... Let me describe the aircraft industry on the West Coast for you at that time period. It was contract-contract and the engineers and scientists were very fluid. You'd go to Douglas or North American or Lockheed or down to Convair in San Diego--wherever the contracts were--so the aircraft industry in the sense that people moved to where the jobs were was a fluid industry. That meant a lot of information changed technique, so I think there was fundamental environment held over from the thirties in the aircraft industry that caused technology to move with people. So it's a part of the question you asked. Secondly, we had a pretty strong university backup on the West Coast, in particular UCLA, but Cal Tech and Stanford and [?]. The beginnings of the scientific organizations such as Rand were formed as a result of the

aerospace emphasis on the [?] literally in LA. Also, the Atomic Energy Commission needed computations of the type that we hadn't had before, and it really turned out that as a result of the Card Programmed Calculator a lot of the early expertise, what you could do with automated computing--as you put it, get you from people banging on Fridens to where it's automated computing--came as a result of them.

IBM quite naturally, due to the aerospace business, watched very closely what was going on the West Coast and they would conduct courses and seminars, bringing... We've had all sorts of visitors coming in to see what we were doing... As in any profession, each group prided themselves on doing something slightly different than the next group. I recall Jack Strong and Frank Wagner over at North American--Strong was in the machine area and Wagner was in engineering--priding themselves on inventing and doing things. So you'll get together and have bull sessions. I don't know if you've heard of the first meeting of the Digital Computers Associates [?]. Really, we'd get together in groups of two's and three's for lunch before that. That was the first formal one organized by Ralph Harris of IBM and...

**TROPP:**

Ralph Harris? That's the name that Jack couldn't remember.

**RICE:**

Ralph Harris, yes.

**MAPSTONE:**

The biggest bill that was ever picked up by IBM for boozing, right?

**TROPP:**

Booze? What was it that writer called it--Drunken Computers Association?

**RICE:**

Well, that came a little later. As a matter of fact, the first meeting was pretty serious and IBM right at the time that was formed was the way with the 701's that we knew about; they weren't delivered yet, and they were beginning to push what were the beginnings of FORTRAN. And Ev Yell...

**TROPP:**

Dr. Edwards.

**RICE:**

Dr. Ev Yowell was quite active at UCLA and the National Bureau of Standards, which got slack, entered into the mix. So I guess what I've described that as philosophically was a group of people who exchanged ideas freely but were highly competitive with each other and, you know, it was just a natural to generate this environment. You can see that several branches of computing, the things that eventually became NCR going off in that data project out at Northrop, Higgins' operation when he went to Bendix. Another guy you might talk to if you haven't talked to him already is a professor up at Utah...

**TROPP:**

That's a new one.

**RICE:**

He was at Bendix. Dave Evans, Professor David Evans. He came in a little later than this early thing but he picked up the stuff at Bendix and got the Bendix computers out.

**MAPSTONE:**

And he's at the University...

**RICE:**

He's a professor at the University of Utah.

**TROPP:**

And of course I'm sure that's where this contact with Harry Huskey development was also connected with that... chain of development.

**RICE:**

The scientific community--that is to say, the Rand Corporation, the National Bureau of Standards, IBM, and then just our natural getting together, caused information to flow throughout the community and a very healthy competition developed. And the arguments in those days were how many digits should be fixed or floating. FORTRAN came along a little later and there was still a violent argument going on whether we should have the sort of programs that Rand came up with, or other forms such as the machines began at Northrop. I'm the first to admit that in those days I espoused the special purpose approach and that got killed off mainly because, as I pointed out, IBM chose to continue the thrust in other lines, and didn't support it with mass storage which obviously...

**TROPP:**

It's also interesting that another of the important things that you spawned out of those

get-togethers was the organization of sharing. It came right out of that. You know it spawned so much and I guess another point that ties in, because you described the closeness and the inter-weaving as well as the competitiveness...

**RICE:**

People moving competitively... friendly competitiveness...

**TROPP:**

But also the isolation away from the East Coast development, so much so that architecturally, if two machines were built anybody could tell you whether it was an East Coast machine or a West Coast machine. The whole approach, the whole philosophy behind

**RICE:**

Block diagrams versus equations; I remember it all. I think in those days you will recall it was a five-day trip to get to the East Coast. If you're lucky you get on a DC-3, but there was a geographic isolation in that sense. IBM were the ones that traveled a lot; the rest of us only traveled a little bit. Well, it evolved to where the thing that you saw philosophically... the three generations I mentioned had happened in the first two or three years with the Card Programmed Calculators, so people evolved then to where one of the big philosophical arguments was, "We don't want to plug wires in, we just want to store programming;" and then the development naturally from interpreting routines on out to what we now see as the full-blown operating systems, time-sharing gradually evolved. The other interesting facet was that Remington-Rand (was Rem-Rand in those days, or Sperry-Rand) had a shot at it. At Northrop we talked to Remington-Rand pretty seriously about getting what became the 787, of IBM, and they were interested but they never really decided to go with it and IBM did. The net result was instead of Remington-Rand coming up with Card Programmed Calculators, IBM did. We were seriously, at Northrop, looking at their equipment and their ability to connect their tabulators with... From a performance viewpoint at the time the Card-Programmed Calculator started Remington-Rand equipment really had a little edge, you know, if you just looked at it and didn't care whether it was IBM or Remington-Rand. They're a little bit better. But where they lost out was in the decision to spend money and to push it. It wasn't that their equipment wasn't...

**MAPSTONE:**

It was IBM's good philosophy of taking care of your customer first and responding.

**TROPP:**

It's interesting that both of those grew out of the competitive early tab machines, IBM out of Hollerith and Rem-Rand out of Powers. Those were the two major competitors, and

the same kind of contrast existed in their early days...

**RICE:**

Well, I think you observation that the West Coast was sort of a hotbed of a lot of things coming out is correct... I attribute it to that aerospace environment.

**TROPP:**

How about people, say at Boeing in Seattle, which is in that period a long way up the coast. It's almost as bad it it was in New York.

**RICE:**

Competitively speaking they had to track what we were doing but I don't think the personnel... I don't recall at a computer society meeting, or whatever we called them im those days, seeing any Boeing people, but they used to come up to San Diego from Condor. People would drive up for lunch or dinner. Ben Ferber was one of the people in those days. He put the Card Programmed Calculators and succeeding machines into Condor. Ben Ferber.

**MAPSTONE:**

Ben Ferber. Was he...

**TROPP:**

Is he still with...?

**RICE:**

I don't know where Ben Ferber is. Frank Wagner might. People from Condor came up so we did, within driving distance, essentially did attract them, but to get people from Point Mugu to come down, for example...

**TROPP:**

What was you connection, in terms of this environment, with the work that was going on at the Bureau of Standards at UCLA, in terms of their numerical analysis?

**RICE:**

Indirect, indirect. The Bureau of Standards did two things they got slack doing and I think to a large degree, as you mentioned, some of the random number generators that came out of Rand, we used them as a consulting source for mathematical methods rather than considerably.

**TROPP:**

That is the area I was most interested in.

**RICE:**

And we'd go up... I happen to remember Ed Yell, and I'm sure there were others there, I just don't remember... we'd go up and present our problems and talk about algorithms, debate these things, and usually they ended up with a definite interchange of information. The fact that everybody was within driving distance...

**TROPP:**

Essentially what you had was a kind of invisible university going?

**RICE:**

In the forefront of a new industry where we were all on our own, all experimenting and all pushing our... to be highly competitive.

**TROPP:**

This raises the question of the kind of... what almost was but never did... and that is the push in terms of the number of people and in the Card Programmed Calculator area, the MEMDATA group, those who were involved in trying to get BINAC to run, and trying to get Northrop into its own computing business. Do you have... were you close to that?

**RICE:**

Part of it. The people themselves--that is, the Glenn Hagens, the Jerry Mendelsons, and others--always felt that Northrop should get into the computing business, and they had... the one effort which was the MEMDATA, which then was sold to Bendix (if I remember the sequence there).

**MAPSTONE:**

Yes.

**RICE:**

But I think it's fair to state that Northrop's management was always aircraft oriented, and it was only the Stark missile, and only the guidance problems that got them into the computing, and I don't really believe anybody above Al Weaver shared the feeling that (a) computing was going to grow like it did. I think if anybody had the foresight to realize what had happened, in retrospect they should have really grabbed onto it. But

they were aircraft management oriented. That was certainly true of Northrop himself, and then later he retired; and the other group, they were always aircraft oriented, so their prime business was aircraft. They tried again with their electronics division, but they never really decided that's going to be a major thrust, even to this date.

**TROPP:**

At that point in time...

**RICE:**

It was always those of us at the lower level pushing for it, but I don't recall any management emphasis to capitalize on what they really had.

**TROPP:**

Regardless, there were meetings... I'm sorry... there were meetings to attempt to convince Jack Northrop, or at least one such meeting.

**RICE:**

Oh no, there were several.

**MAPSTONE:**

Was one of the considerations the fact that Northrop was in financial problems at that time?

**RICE:**

I really don't know.

**TROPP:**

Looking back, you know strictly academically, at this, you can see the centers were [where?] the people exist who have the largest amount of experience, know-how, imagination in terms of what tomorrow's going to be; and first you think of the Moore School at Philadelphia, you think of Princeton, you think of Harvard and the Bell Labs--those are all these folks. The only other place where you find that collection is right here at Northrop.

**RICE:**

Yes, it started at Northrop.

**TROPP:**

Northrop had an incredible collection of people with interest, talent, and rapidly gaining experience.

**RICE:**

And insight and drive. I suspect part of it had to be the available capital to do it, but again you have to recall that Jack Northrop was an aircraft engineer primarily. The main thrust of the company in those days was missiles and aircraft.

**MAPSTONE:**

Did you yourself get involved with the Medeita?

**RICE:**

No.

**MAPSTONE:**

Operation. But you...

**RICE:**

Oh yes. We had continual interchange daily of information between the missile group. In fact, we had desks down... there [?] was about a mile apart geographically. I saw missile people, the guidance people doing the calculations, at least every day, and there was a continual interchange of information and discussion--in fact, friendly competitive discussion.

**MAPSTONE:**

One of the things that, you know, I talked to... talked to the people who were, you know, actually doing the gut-level building, and they were very close to it and really felt this was the way to go in computing. Did you, as a slightly more outside person...

**RICE:**

The MANDATA route? No, I never... sorry, let me see if I can characterize how I think of computers for you. They are a pot full of parts to be organized to do what you want to do with them. MANDATA was specifically set up to do differential equations which worked with time increments. There are also kinds of problems where you do all sorts of things in parallel and they hit at a time when drum technology was available. I think it was an absolutely beautiful, just fabulous, concept of putting together an application and technology to do a specific job of digital differential equations, which... So I always use MANDATA as a fantastic example of a special purpose computer to do a special purpose

job using the components, and in that vein it was a beautiful thing to see it operate, you know. We just weren't getting computer speeds... there was no way the 701 could do a digital differential equation like they can do it on the MANDATA in its little box. On the other hand, if you tried to calculate the payroll, or structural calculations, or something that wasn't strictly differential equations, it fell apart. So, we in the other end of the operations there, viewed it with a very great deal of interest for that class of problems. In fact, I had three or four people working for me, and we had a MANDATA in there and we did do digital differential equations. So, we had BINAC, the Card Programmed Calculators, a little later the 797's, and MANDATAs, all in the computing operations. We got to observe them all.

**TROPP:**

I'd like to talk about BINAC, because that's the big mystery. The sequence as I gather it is that somebody in Northrop... that someone decided they ought to have a machine built.

**RICE:**

They had the need.

**TROPP:**

They had the need and they made arrangements with Eckert and Mauchly, who were the only people capable of building the machines...

**RICE:**

That is correct.

**TROPP:**

To build the machine based on their work at the Moore School: the store program concept, in a general purpose computer called BINAC. As I pick up the train of knowledge of the people who were involved in, say, MANDATA development, they had already started developing MANDATA (MEDITA?) to solve their class of problems. When they found out about BINAC...

**RICE:**

I guess I wasn't specific.

**TROPP:**

Well, that may not have been a major issue...

**RICE:**

I think it's probably parallel...

**TROPP:**

But BINAC was along before...

**RICE:**

Yes.

**TROPP:**

This group went back to Philadelphia--I'm talking now about Floyd Steele...

**RICE:**

They were part of the environment we were all in those days. You have to put back in the environment in those days to correctly judge it. None of us had computers of any kind capable of doing the jobs that had to be done; we were searching for instruments. We also had to have something that could be airborne, as well as something that could do the preliminary calculations on the ground, so there were really two thrusts if you want to think of it that way. There was one thrust towards getting automatic computation that would set up all the variables, preprogramming the flight but everything ground-based, let's put it; and you also had to get the equipment which you could put on the guidance platform so that you could compute in the air. But I think if you look at it in that philosophical vein, both BINAC and MEDITA were inevitable.

**TROPP:**

So BINAC is contracted for. You've got a group of people who go out and learn how the machine is built, how it's put together, and, as you say, what the parts do. They had essentially the same job that you were faced with when you first saw the diagrams on the CPC, and Eckert, Mauchly, Grace Hopper, and maybe others go out to the West Coast. And then the machine is delivered. I guess my question is: during the period that BINAC was there, what did it accomplish? Now you mentioned--you touched on some from the philosophical point of view, and I guess from a problem point of view--a learning environment, an introduction to this kind of machine: I guess all of these things are worth your commenting on them.

**RICE:**

OK. In all fairness, I think Northrop was very interested in getting BINAC as fast as they could; and BINAC, for its day and the components they had, was a tremendous engineering problem. They went to four megacycles with tubes, and they had mercury delay lines... Both of those taken independently is quite an advance in the state-of-the-art.

Put them together...

**TROPP:**

One megacycle at that time...

**RICE:**

One megacycle at that time was fantastic, but they went to four. Put them together, you get a problem. Well, I don't know from the Eckert-Mauchly viewpoint what they felt, but Northrop wanted the machine so badly they really took it before it ever fully ran. They ran a few things; you know, hold your finger up to the wind and hope everything's tuned up and whistle through it--sort of demonstrate you get pulses through it--but it never really ran back there. Northrop accepted it and then set it up and undertook the job of finishing it up. I entered the picture on BINAC when I went down and took over the job of Associate Director of the Computing Center. BINAC was already installed, but never yet run, and we were faced with the job of what do you do with it? Do you continue? So you not continue? How do you go? In the meantime you could gradually feel the thrust of computing leaving you behind (speaking of BINAC people now). The 701's been announced. They had found other solutions to missile guidance on-board computer problems; and BINAC was always one of those things, "Well, if you dump in a few more hours of labor it's going to be up and running." Philosophically BINAC went to four megacycles with tubes. That's one of the things you can say which probably helped advance circuit analysis rather drastically. They went to mercury delay lines, which were in competition with Williams's tubes, neither of which survived; but I would say it was a worthwhile experiment to get in that field. Thirdly, it was the (to my knowledge at least) first dual system where you literally had two computers, each of which we could check the results and decide whether you got the right answer or not. So, philosophically, it was all of those things. Practically speaking, we got... we--the people working on BINAC--they were literally engineers trying to get it running. There was quite a lot of programming done, but that tapered off as it became apparent that BINAC wasn't going to operate. So the programming emphasis was much less at this time period. The hard wrench was to get it reliable. We put quite a few—many years of effort into trying to bring it up, but it was just too much of a backbreaker. If you tuned up the delay lines, you had trouble with the tubes; if you tuned up the tubes, by that time the delay lines had drifted. They did actually, literally, get it to where a small problem ran, with both halves running and they compared the results; but it was never in a reliable state: you know, where you could go in for eight hours of problems. So, I guess I'm one of those people who had the unfortunate decision to make of what do you do with it? I indicated, 'Let's see if we can give it to an educational institution to gain what insight they can out of students looking at the machine,' and we arranged for it to be given to UCLA. So BINAC, as such, never really got off the ground, in the sense that did computing.

**TROPP:**

How about the people that it trained?

**RICE:**

It trained an awful lot of us; Dick Baker, for example. It got an awful lot of us thinking about storing programming. It got an awful lot of us thinking about things that went fifty kilocycles. So, I think it played a very important influence in the thinking of many people. In my mind it turned me on, on stored programs as they were coming out, because it was... by comparison, I guess it's because you always feel at home with whatever system you learned. I happened to learn the diagrams on Old Betsy... I like a machine you can walk up to and understand what's going on in it, and make it do what you want it to. The BINAC people were an early-day version of what we now see in programming work. But the programming is completely and utterly functionally separate from what's going on in the hardware. It's like the hardware is something out there and there's an invisible boundary, and you program it mathematically and away she goes. I think probably BINAC had a bigger influence on Eckert-Mauchly than it did the center of gravity of computing on the West Coast, except that it made people think o high speed computing. It prepared us for the stored program systems to come.

**MAPSTONE:**

It prepared them for the UNIVAC line...

**RICE:**

Oh, by all means.

**MAPSTONE:**

They must have learned a tremendous amount.

**RICE:**

Well, that's what I was saying. I think Eckert and Mauchly, you see, went into Remington-Rand and came out with their successful UNIVAC line, really traced the genealogy of BINAC and this fallout far better than we did on the West Coast.

**TROPP:**

Well, in many ways I think that it all fits at the same time the strength and weakness of BINAC, in that apparently it was a machine that was designed from an engineering point of view far beyond the level at which people...

**RICE:**

Technology...

**TROPP:**

Understood the technology at that point in time. Had it been more conservative, it probably would have worked.

**RICE:**

If you had modern components at the frequencies we were operating and you tried, that is, cores or semiconductor memorites and integrated circuits, BINAC, for its generation, would have been a very excellent machine--extremely productive--and would have probably generated an awful lot more in the way of programming technology. As it turned out, when you don't have a running machine you don't really generate much program technology.

**TROPP:**

As you look at the ... in 1949, which was before BINAC was really delivered to you, there was this sort of program computer running with mercury delay lines, and it was operating at half a megacycle.

**RICE:**

Yes, which is reasonable.

**TROPP:**

Which is, you know, one-eighth of what BINAC attempted to do.

**RICE:**

Then you had Aiken up at Howard with his system going... Harvard... at that time. Then you had this selective sequence calculator.. Then you had the ENIAC.

**TROPP:**

And you had the Bell Lab's relay machines. Did you have the opportunity during that period--say between '46, when you joined Northrop, and '50--to go to the East Coast to visit some of the installations?

**RICE:**

Yes. I visited Aiken at Harvard, and the selective sequence calculator at IBM, but I didn't get to Bell Labs or Moore.

**TROPP:**

Did you get to MIT to see the Whirlwind group going?

**RICE:**

Yes. As a matter of fact--let's see, when was it?--well that wasn't until after '50 that I got to MIT.

**TROPP:**

By then Whirlwind was running...

**RICE:**

Whirlwind was running by the time I got there.

**TROPP:**

What was your reaction to each of the things that you saw on the East Coast, as contrasted with your own experience here?

**RICE:**

Whirlwind impressed me probably three ways. First of all, how many rooms full of stuff it was. (Laughter) Secondly, that it worked. You know, university generating a machine that's worked was impressive, and certainly Whirlwind from that viewpoint was impressive. Thirdly, I thought that the teaching of programming and the programming technology that Whirlwind spawned was very impressive. This was by '53, I think.

**TROPP:**

Could you elaborate on that, because that's an interesting... point...

**RICE:**

Well, Charles Adams and Dave Forester and others were teaching courses on... by this time the computer's a box, and you don't worry about what's in it, you're starting to do routines. And they had the... they were really teaching classes to teach people how to do programming. Now the language was unsophisticated yet, but you could feel them coming, and you could do this... well, in a short classroom example you did the bouncing ball thing. Now I had done the bouncing ball thing on Old Betsy, and had wired plug boards to do it, and knew pretty well what it took. Now realizing that Betsy's no competitor for a portable in terms of internal hardware, but I was at least aware of what it took to wire that algorithm in a plug board to do a special purpose job. We were able to go back to MIT and sit down. They gave us a couple of days of classroom lectures, and Art Moore went toward its unit and sat down for a few minutes, programmed a problem, and [?] ran it on the machine. It was fairly impressive. One could get a feeling of what

was coming in program development, so... I believe Whirlwind, as I say, I got three impressions... I gave them to you. It was the first experience that I personally had--I'm not saying there weren't others to parallel--Grace Hopper and others were doing it at Remington-Rand. Some of the people at Rand and Ed Yell were doing it on the West Coast. It was the first impression I had of an organized approach to teaching masses of people what the machine will do for them--in contrast to an organized approach within a company to get our own job done. So those were my impressions.

**TROPP:**

I think of the Whirlwind in terms of machines as a major milestone, but I'm always interested to hear other people's comments about it. Did you have the opportunity to visit Princeton and see the engineering work that Bigelow and his group had done?

**RICE:**

Unfortunately, I did not.

**TROPP:**

Because that, too, from an engineering point of view, was very impressive.

**RICE:**

That started the chain.

**TROPP:**

It spawned many machines.

**RICE:**

That was a very key effort. No, I did not go, unfortunately--go down to see it.

**TROPP:**

I think you've made a really important philosophical point in terms of the very first areas we were talking about--the impact of the CPC--because philosophically, apparently its impact on you as an individual has stayed with you to this day.

**RICE:**

That's right. It really stayed with computing to some degree, two ways. I think the CPC and the 701--or, if you prefer, Whirlwind, they're all about the same generation--and the work that Eckert-Mauchly and Hopper were doing at Remington-Rand, spawned what I'll call the programming industry, that none of us understood in those days. You know, I

don't know how much computers now... ask you a question as long as you've got your philosophy. How do you rate the productivity of a computer installation?

**TROPP:**

Not being a computer oriented individual, I couldn't answer that question.

**RICE:**

Try it anyway.

**TROPP:**

The productivity?

**RICE:**

Mm-hmm. Of a computer installation.

**TROPP:**

Well, I guess it would be the efficiency with which I as an individual could get something done on that facility, as compared with how I could do it elsewhere--say if I were sitting down with pencil and paper and trying to solve the problem...

**RICE:**

You're speaking now as a person?

**TROPP:**

I'm speaking as a mathematician who has a problem to solve.

**RICE:**

Suppose you were the company or the university running that computer center, how would you rate that efficiency?

**TROPP:**

I guess one way to rate it, and again. I'm only...

**RICE:**

I liked your first comment, but now let's talk about...

**TROPP:**

I guess I would rate it in terms of the complaints I got from people who would say to me, "When am I going to be able to find enough time to solve the problems that I need on it?"

**RICE:**

O.K. Let me give you a little more philosophy... Wouldn't you rate it on the number of jobs that went through, and perhaps the turnaround time service you gained?

**TROPP:**

Well, that's essentially the equivalent...

**RICE:**

Did you ever ask yourself the question, "How many of those jobs are useful?" For example, when I go up to computing installations and I see a job going through, it's because somebody dropped a card deck and they had to reassemble it, or, there was a hole punched in the wrong place, or now, on time check, somebody pushed the wrong key. You know, you rate a computer in a computing installation by its volume, but not its efficiency in solving, what you said, first your problem...

**TROPP:**

That's right.

**RICE:**

And we've gotten worse and worse and worse. If you rate efficiency in terms of jobs done and you also take account of the fact that we're in millions of cycles per second compared to kilocycles, we're doing a worse job today in getting jobs done than we were on Card Programmed Calculators.

**TROPP:**

Of course part of that problem is due to me as the individual. I have a problem, and if I really want to use the machine effectively--and here we're out of the subject of efficiency--I should do everything I can towards the solution of that problem that I'm capable of as an individual. I only should go to the machine when the machine can do something I cannot do. At that point, I'm using the machine effectively.

**RICE:**

That's the first item. Now the second item is that the machine and its software system is

so complex you can't understand it. So you have to depend on getting two kinds of programmers: an applications programmer who interfaces with you and uses the software system; and an invisible bunch of people who have programmed that software system which makes this very fast machine do your problem. There are umpteen layers buried between you and your problem, and you get such answers back as, "We can't go double precision because," or...

**TROPP:**

As an individual, I don't want any layers between... that's the other problem...

**RICE:**

That's what I learned from the Card Programmed Calculator, and let me follow that genealogy for you. I left Northrop to join IBM and had a very pleasant association with them. That was in '55--with the idea that maybe we could get, not a Card Programmed Calculator per se, but the idea that you had a special purpose--we now call it micro program--but a special purpose functional language which you plug in, what your plug boards were. We actually had a project going at IBM; eventually it got killed as many of them do. The 650 came out and then the 704's. Anyway, I had the philosophy in those days that the plug board would serve the purpose of allowing you to tailor the machine language to a higher macro-level to do its special purpose, but still realizing that you'd have layers of system programming, and then application programming above that, but trying to get more macro-type instructions available to the user: you. In about '58 I realized that an awful lot of the threshing that we have to do in software is just moving data from here to there. I'll call it memory management in its total context. The 704 line had taken over and started to do some work, so I convinced them to do a little research on it, and came up with the idea that you could automatically manage all your memory resources. The net result was that we ended up with a research project which we finally called ADAM, between '58 and '62. In fact, we actually got a contract from the Air Force to do work on that machine. There are documents around on it. The idea behind that machine evolved back from the old plug board days: why don't we get macro-type instructions in the system? But I define macro instructions now as useful things you want to do on data. That is to say, why put binary addition in a machine when people work in decimal? Why don't you get it fast enough and just make it straight decimal? Why should we manage memory resources in software? That's a function that the hardware ought to be able to do. So we actually had blocked out a machine which is reported in the '63 Spring Joint Computer Conference, called ADAM. But IBM at that time came out with the 360 line, or it was in the works, and decided that they didn't want to continue in the other vein. So, I had the opportunity to come here to Fairchild to work on semiconductor research, but what they told me is, "Well, if you'll do research on packaging and other things we need here, we'll let you build what you want with our components." So we started then. I laid off of logic for about a year, but I always had in the back of my mind this automatic memory management, and what I will call macro-level machine language would be a boon to computing. As a result, we were able to actually, in fact, build such a machine. It exists. It's now at Ohio State University and

supported by the National Science Foundation--where the high level language is directly implemented in the hardware. I think perhaps we overshot slightly, in the sense that we put some things in it we didn't have to, but... the time-sharing system--think of the amount of software that goes in the operating time-sharing system as essentially hard; all memory management--virtual memory normals--flatout hard. It takes something you can't even get in any software system: its completely and utterly variable field at execute time. never do you declare the size of data, the size of structure, or anything. It's automatically handled. All of this is in hardware. So the outgrowth of another train of thought from the old Card Programmed Calculators--make the machine do what you want--has ended up in the result of SYMBOL. Now SYMBOL, as such, is not going to be manufactured by major manufacturer because it's too radical and too much of a change. However, very fortunately, it's out in the educational community, and the new group of students coming up are going to now look at it and say, "Well, why don't we?" What we didn't do is implement FORTRAN directly, or COBOL directly. We went back and said, "What is a fundamental operation that everybody does?"--You know, FORTRAN does it this way and COBOL does it this way--and literally, I call it typewriter English, or mathematical stuff if you want. So we have evolved then this machine which came up from that chain of events.

**TROPP:**

Is that the right spelling: SYMBOL?

**RICE:**

SYMBOL.

**MAPSTONE:**

And you've evolved a complete programming language to go...

**RICE:**

Well, the machine language: the raw machine language and the programming language are very nearly equal at the start; but now you can build routines up from that, which will go on. But you can walk up to this machine, you and I could sit down at a console on this machine and I could teach you just a few rules--very few, like ten or fifteen--and if you had a simple problem to program (and let's start out with our old bouncing ball example, which is trivial, but it's in point), literally what went on in the machine and what you typed out on the input typewriter would be one and the same; so you would know when you goofed up, and you'd know what to do about it.

**TROPP:**

You really almost come back to programming a machine language?

**RICE:**

Without plugging wires into the micro program.

**TROPP:**

For me, as a problem solver it sounds like the kind of way I would use this equipment.

**RICE:**

I've tried to bring machine language up very nearly to what I call unity in programming language, which is where you get when you get FORTRAN.

**MAPSTONE:**

Well, for someone who has struggled with programming language, it sounds fantastic too.

**TROPP:**

Well I say, as a problem-solver...

**RICE:**

Such a machine exists now.

**TROPP:**

As a problem-solver, this means that when I am at the point where the machine can assist me, then the machine really does help me get my problem solved.

**RICE:**

You might find it interesting...

**TROPP:**

It's not just a number-cruncher.

**RICE:**

You're trying to follow the early history: you followed the MANDATA branch, you probably ought to look at what happened to BENDIX; you followed SLACK, you followed BINAC, you followed WHIRLWIND. You might even want to project a little bit of what came out of that early stuff that might influence the future. We think some things are going to come out of it in the future. Not a machine...

**TROPP:**

Well, one of the people that I haven't talked to--and all I have are second-hand comments--and that's Howard Aiken, who's a very key person...

**RICE:**

Yes.

**TROPP:**

...in the history of computers.

**RICE:**

Yes, absolutely.

**TROPP:**

Now Dick Block, who sees him relatively frequently, tells me that at times they get into philosophical discussions about the future, and I'm most anxious to talk to him--talk to Dr. Aiken-- on this point. But his background goes back to the first machine at Harvard...

**RICE:**

He built macro instructions...

**TROPP:**

... which is really a machine that took a problem as he, a scientist, would have solved it, and sequentially went through the same stages. And I think--and again...

**RICE:**

He did it with the hardware he had available in those days, and we've done it with modern hardware.

**TROPP:**

I'm only paraphrasing second-hand conversation. He, too, feels that much has been lost in translation as our technology has increased, and (Toof?) also seems to feel--and again I'm only quoting somebody else--that much of this is going to return in the future.

**RICE:**

It can; whether it will or not we don't know. It can.

**TROPP:**

I'm most interested in talking to him.

**RICE:**

Aiken and I talked to each other--or at least I talked to him, I guess, is the way to put it--away back in the early days; and I think philosophically I shared very much the same feeling he does: that as a user, a guy trying to....See, in those days I was a structures engineer trying to put problems on the machine. I didn't like all this garbage I had to learn; I still don't! Now I'm a computer designer, but I'd like to get rid of the garbage for other people.

**TROPP:**

Have you seen Aiken recently?

**RICE:**

No, not for years and years and years.

**TROPP:**

Can you think of any other people who were in your environment in that '48 or '46 to '55 era that you have contact with today, who have philosophical thoughts on this subject, either agreeing with you or disagreeing with you, but whom you've talked to on this...?

**RICE:**

I talk to Eckert quite frequently. We have a great deal of agreement. I lost track of Ev [?].

**TROPP:**

I saw Ev just a few weeks ago. He's got his own consulting service.

**RICE:**

Jerry Mendelson and I are sort of at odds in the philosophical approach. In fact, I'm the oddball these days. I still believe that we ought to make programming easier instead of making the machine... See... I don't know if it's appropriate to your deliberations or not, but philosophically, an engineer who designs a machine sees the results of his work--or the group sees the results of their work--something like three to five years after they start; so the reward is remote, you know, from designing a logic equation or putting something

in: the result is remote. The programmer, in contrast, sees the result of his effort something[?] in minutes--with time-sharing, seconds. The reward and punishment cycle, if you will, of computing as it exists today, is drastically out of balance. The people will settle--I've watched my own engineers who did SYMBOL here the same way. I like to describe them as they disappear in the typewriter and all you can see ids their heels sticking up. They get so happy with the accomplishment they made--something you can see the result come back immediately--that they get lost in software, and very few of them will stand back and ask, "Do we really have to do it this way?" You know, would the generation beyond us be doing it that way? I think the whole industry--ninety-nine percent or better--has not even addressed themselves to the question: 'Is this right or wrong?' You know, they just stick their heels in the darn typewriter, and literally you got to grab them by the heels and pull them out once in a while. So, we don't find the emphasis now in the industry on radically different machine organizations. We find the emphasis on speed, or sophistication, or amounts of hardware, if you will--amounts of bulk file--but we don't find any major emphasis on getting rid of the software... and I mean the system software.

**TROPP:**

I think Twan Chu had a telling comment that I've mentioned a number of times, in that, in terms of high speed computing--computability or ability to compute, whatever phrase you use--ENIAC made a quantum jump...

**RICE:**

Yes.

**TROPP:**

But from ENIAC to today we have not quite doubled that jump, if you will. In a sense we really haven't come that far, is what he was saying.

**RICE:**

I'm saying one other thing. The software required with pure [?]-type machine organization with goosed up performance, is what is holding us back. See, the hardware speeds are fantastic. But what do we do? We do a dribble little operation every microsecond; you know: move something from here to here.

**TROPP:**

Well, what you're saying I think is very relevant to our deliberations, because we're concerned with the flow of ideas and concepts and their environment people. So, let me throw another really broad question at you, and that is: form your viewpoint and your experience, what do you consider the major milestones?

**RICE:**

You mean from the start?

**TROPP:**

That's a ...

**RICE:**

Starting with Bamage?

**TROPP:**

No, starting with Rex Rice. Oh, start with Bamage if you will, or Pascal, or [?], as far back as you want to go.

**RICE:**

The mechanization of information-handling which occurred with punched card machines has to be a milestone. If you want to go back to that, we can.

**TROPP:**

Right, the whole Hollerith approach... Jacqard loom if you will...

**RICE:**

The Hollerith...has to be--well, for its influence on the rest of us--has to be a major milestone. I think probably between World War I and World War II the scientific community's emphasis on getting sophisticated hardware of all kinds in all sort of things--like the ballistic missiles--in getting—I'll call them digital numeric algorithms, in contrast to just...

**TROPP:**

One of the real centers for this was somebody like Connery in England who really developed sophistical ways of using machines.

**RICE:**

So I think the stage was set by what I'll call numeric analysis and the digital setups. I don't mean just numeric analysis for pure scientific reasons, but where there was an algorithm developed that gave us the ability to calculate in a digital sense. Then, of course, I think the Card Programmed Calculator was a major break with tradition. Paralleling that was the new hardware and the Von Neumann concepts. We essentially,

with all the modern machines, haven't changed very much from the Von Neumann concepts in what the level of the machine is. We've sort of abandoned where the Card Programmed Calculators were heading us with macro level languages in the machine. As a matter of fact, we're struggling now with micro level...

**TROPP:**

Well, that's essentially the natural thing... ENIAC through Von Neumann leads you right into microprogramming.

**RICE:**

Obviously, it's brought the world a tremendous amount of calculating power that they would never otherwise have. The only question is, are we going to continue that way, or not? Not: is it good, or bad? OK, so I would say card-programmed machines up to ... I think the Card Programmed Calculator was a break with tradition, in that, all of a sudden--as you pointed out earlier, and I never thought about it that way before, but--it suddenly got people using automated processes. I think that that has to be a major milestone; not in its specifics, but the fact that [it?] brought computing to lots and lots of people that had never seen it before. It forced us to go back in sequence steps, not with people on Fridens, but with machines. Then obviously the UNIVAC--ENIAC-UNIVAC group with Hopper and their work in programming--the WHIRLWIND at MIT. I'm not personally familiar with the Bell system that much so that I can't...

**TROPP:**

That was again an early development.

**RICE:**

Yes, but even so, I don't think it influenced my life at all.

**TROPP:**

It influenced a lot of people who visited it and saw the fact that you could do certain things.

**RICE:**

But I can't relate anything directly to that. You asked me how do I feel.

**TROPP:**

Right. I'm interested in your feelings.

**RICE:**

I told you my thought on WHIRLWIND. I think really the 701 was the turning point in all of the world's computing, in that we suddenly got a company that was capable of production and sales and sales support--that support was very important, with high speed electronics pushing computing.

**TROPP:**

And servicing...

**RICE:**

We had that now. Don't forget we had that with our card machines. We had it with Rem-Rand. I think it was--I'm sorry, I'm separating hardware service and contrasting software service. I think IBM had a push to put software service out there. Probably the next major break in computing came in languages, not in hardware, the things that led us to FORTRAN and later to large systems.

**TROPP:**

Like Pl/I and ...

**RICE:**

Yes. They've influenced the industry, but they influenced me to try to find different ways to do it. I'm essentially a guy who'll work like the devil to be lazy. It really bothers me to go into a software system and start using it and have the work come back: you can't do it, but I don't know why. It's because some programmer six years ago wrote some routine to use some latch somewhere, and the latch doesn't know what it's doing, but I can't do it. It bothers me, co I guess my emphasis has always been, how can we take the very sophisticated and elegant hardware technology available, and the excellent applications we're putting on, and get rid of that mass in the middle. I'm not sure we will...

**TROPP:**

I think a lot of people are disturbed for a lot of reasons on the current scene and a lot of them that I've talked to are not quite sure why they're bothered.

**RICE:**

I'm definitely sure why I'm bothered.

**TROPP:**

Yes, you've been very articulate about your unhappiness, but...

**RICE:**

I want to be able to walk up to a machine and say... for example, I can now buy a desk calculator, desk calculator, a little electronic gizmo. It's still like the Friden I used to run except a little more sophisticated. Let's take the Hewlett-Packard desk calculator. I can walk up to that thing and say I had two numbers to ass and get the answer. If you tried to go up to an IBM 360/67 and add two numbers together... (LAUGHTER) Like twenty minutes later they bring the software system up. You know this is a machine that gets million(s?) of instructions per second and a duplex gets two million, per second. It's fantastic to sit there and watch all these lights, tapes flapping , and try and add two and two together. Now got to the other extreme. A problem which is very massive like Sid Fernbach brings over to Livermore and he'll spend many years programming it and you put it in the machine and get a fantastic output that you get not other way. The end we've missed is the application work, where the application is [?] . The other end we've missed is helping [?] . Why should I have to tell a machine ahead of time how big the problem's going to be? I don't know how big It's going to be a year from now.

**TROPP:**

This ties in with problems connected with, say, information storage and retrieval. You design a system for the amount of information you have, but you want it to be expandable and changeable and you don't want to be locked in to a fixed size. Otherwise, your whole thing may be wrong from the beginning.

**RICE:**

Such a machine now exists: SYMBOL. It works and it will save roughly fifty percent of th file space. What major break did we produce in SYMBOL? We store the attributes of the data information line--that is, size, field length, structure--with the data. We made a major break from the one line tradition. We do not store data attributes in the stored program; we store them in the data strip and then we learn how to automatically handle it.

**TROPP:**

So that the system then would be expandable and changeable?

**RICE:**

At execute time. You never have to pre-declare. Up to the limits of memory. Not only is it expandable dynamically, but you literally compress memory compared to the way you normally do it. You can save somewhere from thirty to fifty percent of the file space. So there's a long way to go in computing yet. I think the major influence that my branch of Northrop experience will have, it influenced a lot of these young PH.D.'s to start asking nasty questions... universities...

**TROPP:**

You're not alone in this. I listened to Fritz Bauer give a talk on the subject he calls Software Engineering," which is philosophically in exactly the same vein, and it's a phrase that is roughly three years old.

**RICE:**

Yes. The main difference is I built a machine to do it. In fact, you can trace back--all the way back--Wilkes and others, people wanted to do something about it and doing certain things. We've known the need in the industry; we've mouthed around a lot but its always somebody else ought to for it. I've been to lots and lots of conferences, joint computer conferences, hear somebody get up and say, "Well, somebody ought to do that. Somebody ought to build a FORTRAN machine. Somebody ought to eliminate some kind of language." Yet, when you do it, they're pretty slow to accept it, because they're not used to it

**TROPP:**

I want to turn this tape over; It's about at the end.

**RICE:**

I enjoy you comments on pre-'55 and post-'55. But I think in all honesty, if we look at post-'55, and really look at it in depth, I would think we'd have to say that in '55 the programming profession had not really come into its own. It was sort of... there are hardware people, and there are programming people, and you have them in two buckets. As we look at it in '70, the hardware bucket's pretty small, although we rent machines... but if you look at all the people using the machines in programming, that bucket is extremely large. So I would state that the dominance of software decisions and influence is the side that has changed since '55. In '55 we really didn't yet appreciate how big the software area was going to get.

**TROPP:**

And, of course, '55 was a kind of a watershed in that suddenly, bang! everything exploded and we're just beginning to be able to look back and see that explosion and its impact. That's right.

**RICE:**

So I guess that the only major change has been our underestimating the software side of the picture.

**TROPP:**

This underestimation existed from the year One. Even Von Neumann in the late forties

foresaw a very limited number of machines that would handle all the computational needs of the scientific community for a long, long time to come. Nobody really foresaw what happened. Well, I won't say nobody; I'm sure some people did.

**RICE:**

Not very many.

**TROPP:**

But, for example...

**RICE:**

Even those in the middle of it, like myself, didn't really foresee the explosion.

**TROPP:**

One of Aiken's colleague's mentioned that he tried to get a higher speed printer on his machine and Howard said, "Why? Why do you want anything that prints faster than anybody can read?" And at that time he was right. I mean, nobody thought of a million people reading something simultaneously, in the sense of a bill. But let's go back to this other question in terms of the people that you were acquainted with and their role and involvement at that time.

**RICE:**

Starting with the people that I specifically knew and were meeting, certainly Greg Toben had more influence and impact on me in his philosophy of what a computer is--that you ought to bend it to suit what you need, and not try to give you a course in just how to use it, but to make you stand back and look at it.

**TROPP:**

I think he still feels that way about it.

**RICE:**

Greg Toben also had the good sense to turn Woodbury loose, who also influenced the way I think about computers. Woodbury did in fact invent and show that you could invent things, then stuff a little bit. Probably starting at that time... following the efforts going on in the missile group and tracking the work that Mendelsohn and Edel (Al?) Weaver's people, influenced an awful lot of the thought process of what is computing--what should we do with it. We've already discussed earlier the MANDITA and that made me realize that special purpose computers can go very much faster and very much easier on jobs than the general purpose one can do it. A lot of influence came

from the Rand group in the formulation of problems. I think of Keith [?] among others. Ev Yell and I used to have some philosophical discussions at the Bureau of Standards there. The competition with Frank Wagner over at North American was really healthy. The people at Douglas, Morrison and others, were active in those days. Ralph Harris at IBM, their Santa Monica office manager, probably influenced all of us. As a matter of fact, I would characterize Ralph Harris in his job as the catalyst that got us all together. He wasn't a technical man but he saw to it that his customers communicated...

**MAPSTONE:**

Was he the 701 man?

**RICE:**

No...

**TROPP:**

Cuthbert Hurd was...

**RICE:**

Cuthbert Hurd. But Ralph Harris is the guy that got Cuthbert Hurd all of us. Northrop gave Cuthbert Hurd a very bad time when he came in and presented the 701, a very bad time. He sort of took that program to Remington Rand.

**MAPSTONE:**

Earlier you said that you felt the 701 was not a good enough way of going at that time, and I thought maybe you could enlarge on that.

**RICE:**

If one had presumed magnetic tapes--and as it turned out, Williams's tubes for memory--and the fast hardware that was going in the 701--as much as I think as a megacycle of hardware--and had given us a machine that was like the 797, the one that Woodbury came up with--we called it the Wooden Wheel--I guess you've heard of that one--rather than the 701, I think the whole course of computing would have changed and I think we would have been much more conscious of the application user and macro languages than what I'll call the Von Neumann level, which is frozen in the machines now and lots of software gets you out. That was a turning point. If Woodbury's thoughts of the plug board system he started had been able to gotten [?] through, and we'd gotten at least one generation of IBM machines made that way, computing would be fifty percent different than it is now. As it turned out, we didn't win that battle--that was won by the Cuthbert Hurd's and the other people back at IBM. So the 701 literally, as you put it earlier, froze us into the Von Neumann environment. Now I'm not depreciating the

effect of the 701. I think it did a tremendous job. I'm not negative about it, but if you philosophically ask what did it do?... that's the reason you can't find any, past '55. Did I answer your question?

**MAPSTONE:**

Yes.

**RICE:**

It started a very successful commercial trend, with very great vested interests.

**TROPP:**

Was there any kind of a center of activity going on at Hughes, and if so, who were some of the people?

**RICE:**

A little later than the early Northrop work, the answer to that was yes. Walter Bauer would be the person to contact to find out about the Hughes people if you don't know them personally.

**MAPSTONE:**

Do you know where Walter Bauer is?

**RICE:**

Yes, he's in Southern California. Look him up in last year's IEEE directory. His fellowship... he was the one that... I'm sorry. He's the Chairman of the Fellowship Committee for the IEEE last year. That will be the way to look him up. Excuse me for just a second. (Do we have an address for Walt Bauer? Bill, I'm probably going to be a little while here. I'll still go with you--what time is your appointment? Yes, you might as well go ahead--I don't know. Would you all like to be my guests in the cafeteria here, frankly?)

**TROPP:**

That will be very nice.

**RICE:**

OK, when we get...

**TROPP:**

As soon as we finish up this one portion of the discussion. That's a kind of a center that we don't...

**RICE:**

Yes. Walt Bauer would be...

**TROPP:**

B-A-U-E-R?

**RICE:**

Yes, B-A-U-E-R. She may have his address. I've been in contact with him the last year or so.

**MAPSTONE:**

OK, the names that I have, and maybe you can give me something on them, are: Eldred Nelson...

**RICE:**

Yes. John Postley.

**MAPSTONE:**

Oh, yes, that's right.

**RICE:**

Came from Hughes to Northrop.

**MAPSTONE:**

And then Informatics.

**RICE:**

Informatics is Walt Bauer and Frank Wagner.

**MAPSTONE:**

Oh, that's right: Wagner. Is Walt Bauer there with Informatics?

**RICE:**

Walt Bauer would be... well, even if he isn't there, they know where he is... so you'd talk to Informatics, yes.

**TROPP:**

I guess, in germs of our own research, and this is throwing something else into the hopper without ignoring that broad question that I gave you earlier--we want to look at the total West Coast environment as a research area, and I guess you've got a feel now for some of the things we're interested in--would be some of the key people that you would identify that we might have missed. Now we've picked up, you know, the group that Dick Sprague talks about in his recent article... the CRC...

**RICE:**

Have you got all the Rand People?

**TROPP:**

Now Rand...

**RICE:**

Go see Keith Uncapher and Willis Ware...

**TROPP:**

Willis Ware, Keith Uncapher...

**MAPSTONE:**

George Brown

**TROPP:**

George Brown

**MAPSTONE:**

Bill Gunning we saw yesterday.

**RICE:**

Bill would have a good line on those.

**MAPSTONE:**

Madden.

**RICE:**

Don Madden, certainly. Have you taken care of the Bureau of Standards people at UCLA?

**TROPP:**

that's what I was going to ask you. I've talked to Ev Yowell...

**RICE:**

He's the one that sticks out in my mind... Harry Huskey...

**TROPP:**

Harry Huskey. I've talked to Maynard Reese, who was, in a sense, a supervisor or...

**RICE:**

Yes, but I think Huskey and Yell... and I've given you Dave Evans...

**TROPP:**

What about UCLA and Cal Tech?

**RICE:**

Rogers, Professor Rogers at UCLA.

**MAPSTONE:**

So you know his first name?

**RICE:**

No. He's still there.

**TROPP:**

He's left the area.

**RICE:**

In EE. He's in EE and he's the man we gave BINAC to.

**MAPSTONE:**

O.K. Irvin Reed is there so... and he's also in EE, so...

**RICE:**

Yes, you could find him there. I think it's Don, but I'm not sure of that. Who's the fellow at UCLA now that goes to all the conferences... Gerry?

**TROPP:**

Estrin.

**RICE:**

Estrin. Gerry Estrin will be able to tell you how to get hold of Rogers. Monty Phister came out of the Hughes group, I think, and he was a little later than this early group, but he influenced things.

**MAPSTONE:**

Yes. Stan Frankel...

**RICE:**

Oh, Stan Frankel...

**MAPSTONE:**

... said we definitely should talk to Phister.

**RICE:**

Stan was up at Cal Tech--yes, and he would be... he had a little machine going up there. I'd forgotten about Stan.

**TROPP:**

They had a number of machines, analog machines going there before that period. There's a name--Does [?] Feinman strike you as...

**RICE:**

No, I didn't know him.

**TROPP:**

How about from Berkeley? Paul Morton was involved and built the machine at the University of California, and was also in the... wasn't it UCLA and their analog machine? I think he was.

**RICE:**

I personally didn't run into anybody that I recall from Berkeley.

**TROPP:**

I guess the Boeing thing is another blank area in terms of any info... you mentioned...

**RICE:**

Ralph Harris, Ralph Harris to the best of my knowledge lives here in Atherton and he might be able to dig out some names at Boeing, because through his IBM contacts he knew who was selling at Boeing. I'm completely out of touch with Boeing. I didn't get much contact with Boeing until after I joined IBM, which was after '55, and then I went out and visited them, but... the Los Angeles group I don't recall--my memory's not all that good--but I don't recall Boeing people coming down to meetings or...

**TROPP:**

Because they were involved in essentially the same problem... the residency...

**RICE:**

There was a geographic separation again.

**MAPSTONE:**

Apart from Northrop and North American, were any of the other aerospace companies...

**RICE:**

Douglas: Douglas was active.

**MAPSTONE:**

In their own developments or as a user?

**RICE:**

A little of both; as a user more but they did do a little development.

**MAPSTONE:**

Anybody there that I could contact?

**RICE:**

Contact James Morrison at Douglas in Culver City. I think he's still there and he would be able to dig it out. Yes, as a matter of fact, the group at Douglas El Segundo used CPC's and did a lot of work in doing instrumentation to use them as a part of the structures testing and they did a lot of early data transmission from remote locations into the computer.

**TROPP:**

Remind me of lunch because...

**RICE:**

John... guy who had a wooden leg... Douglas... well, Morrison would know them all.

**MAPSTONE:**

OK.

**RICE:**

John...

**TROPP:**

The use of a CPC, the different uses each of you put [?] to in terms of the kinds of problems you suddenly decided you might be able to solve on them, I'm sure had an impact on the development of numerical analysis...

**RICE:**

Oh, it did.

**TROPP:**

Conversely... I'm thinking of aircraft structures and I remember studying aircraft structures from a textbook point of view, the (summation? acceleration?) of forces around points and this kind of thing. Suddenly, you were in a position to do...

**RICE:**

Very sophisticated analysis.

**TROPP:**

New kinds of analysis. I think of our wind tunnel data. It was very, very crude; you really couldn't get much from it.

**RICE:**

That's right.

**TROPP:**

And suddenly you were able to do these kinds of things.

**RICE:**

That's what the CPC did for us--changed the whole world. Well, as we mentioned, the development of digital numeric analytical methods...

**TROPP:**

This is another research area that I'm interested in, the development of numerical analysis and its impact--or the impact of machines...

**RICE:**

There was a classical textbook--I don't even remember--I've probably thrown it away now--on numeric analysis that came out...

**TROPP:**

John Alston Householder...

**RICE:**

Yes, something like that.

**TROPP:**

I'm interested in getting to him--research and development.

**RICE:**

You know it's kind of asinine when you think back about it--I bet we had more arguments about how you should take a square root than it's worth shaking a stick at. [LAUGHTER] How many megs do you need?

**TROPP:**

Well, a young man was just in my office at the Smithsonian recently who has a suggested research topic, 'The Evolution of Square Root Algorithms,' and I'm sure that would fill a book.

**RICE:**

It would. I'm not sure what good it is. And whether it should be a floating decimal or a fixed decimal is another argument we used to have.

**MAPSTONE:**

That one's still going on, isn't it.

**RICE:**

OH, yes; it'll go on forever. Whether we should use decimal or binary will continue on.

**TROPP:**

Well, your comment on the ease with which one could get all the operations on decimal is an interesting one, because one of the major breakthroughs in the early machines was the realization that a relay represented the Ones and Zeros of the binary system, and the ease with which you could do arithmetic in that system...

**RICE:**

It's funny though, because the old tabulators were all decimal going in. The real reason they went binary--I shouldn't say the real reason: A reason they went binary is it takes fewer components and there are two reasons that generates. One is it's cheaper, and the other is they thought reliability would be a problem. In the early days it was,.

**TROPP:**

Well, as somebody pointed out...

**RICE:**

But now neither of those are necessarily true.

**TROPP:**

Probably either Eckert or Von Neumann or somebody said that to do it decimally at that period of time meant that you had to have sixteen tubes or throw six away... that seems like an awful waste.

**MAPSTONE:**

It would make life an awful lot easier if they'd gone to decimal.

**TROPP:**

Well, but the user doesn't know that. He still works it in his own...

**RICE:**

He does when it blows. Your scaling is not decimal--binary. SYMBOL computer is ninety-nine digits and the mantissa, if you want it, from one to ninety-nine. The decimal point can be anywhere, and it can be either fixed or floating. If it's floating, the exponent can be either one or two digits with a plus or minus sign, and it has truncation algorithms built in that you can tell it how long you want a number to... you know, if you want twenty digits, just put in twenty...

**TROPP:**

That sounds like it would be great for doing prime number searches.

**RICE:**

It's good for research but it's more interesting when you think that we had very little field length storage in doing accounting problems and you need six digits you put in six for your limit; and you're doing mathematical analysis in the same machine and you want to investigate small differences in large numbers, you put thirty in it. I think we've creamed the whole problem of the floating decimal... I'm sorry: of double precision; ninety-nine is more than anybody's wanted so far.

**TROPP:**

How did Iowa State get interested in this?

**RICE:**

They knew of us, and they recruited a couple of Ph.D.'s. They, consulting there at Fairchild, they knew about us all along, and when Fairchild decided not to go in the computer business, which was a while back, they wanted to pick up the machines till they got the National Science Foundation to give them a grant. We, of course, working on the

machine, wanted to get it out...

**TROPP:**

The name CYCLONE I think goes back to one of the early projects when the machines were all named after winds.

**RICE:**

Yes. They built a copy of the early....

**MAPSTONE:**

Before we forget, there's another little phase of this project that I'm just trying to start today, and that is: we thought along with the tapes it would be really good to have photographs of the people we talked to. So, I just realized I walked in with a camera, which probably was illegal...

**RICE:**

Yes. As long as you don't show it...

**MAPSTONE:**

But while we're talking, one of the thoughts I had was even if you would mind playing games with the camera for a little while.

**End of Interview**