Oral History of Joel Birnbaum
Part One

Interviewed by:
Chuck House

Recorded: March 14, 2014
Mountain View, California

CHM Reference number: X7119.2014

Chuch House: OK, Joel, I'm Chuck House. I'm with the Computer History Museum. It's a pleasure to have you here. And let me just preface it for the group to say, I think you're the only person I know that ran IBM Research and HP research. In over a 40 year career, those are your two jobs, right?

Joel Birnbaum: That's it.

House: That's it. So I think you're a perfect candidate for the museum to have recorded, and in our archives. We'd like to find out a little bit about you first, and then a fair amount about what you've done over this career.

So let's start with you. Where were you born, raised? Where did you choose for school?

Birnbaum: I was born in Bronx, New York, a suburb of New York City, 1937. And I spent my first 12 years there going to New York public schools. I had just been admitted to a special junior high school called Herman Ridder Junior High School, which was by exam. I guess today we'd call it a charter school for science.

House: OK.

Birnbaum: And which was the principal Bronx feeder school for the famous Bronx High School of Science, where I had hoped to go. But my father decided that we would move to Long Island. So instead I went to-- I grew up in a little town, about 20 miles from the city, called Roslyn.

And I went to the Roslyn Public school high school. And from there I went to Cornell University. And then I went to graduate school at Yale after that.

House: So you had early history interest--

Videographer: I just want to adjust his microphone. Because it's rubbing against his tie a little bit.

House: Do we need to re-record all that?

Videographer: No, I think it's fine. Continue.

House: So you clearly had an interest in science by junior high school.

Birnbaum: I had an interest in science almost from the earliest times I can remember, mostly due to my father, who was not a scientist, but who wanted to be one. And was a very intelligent man who read very widely in science. He had been accepted on a full scholarship to go to MIT. When, unfortunately, his father died the summer that he was supposed to go away to college.

And he was the oldest son of a five son, very matriarchical immigrant family. And his mother decided that they couldn't afford for him to go away to school, that he needed to work. So he wound up, at her request, becoming an accountant.

And he made enough money to send one of his brothers to medical school, and another one to a college to become a teacher. But he himself became an accountant. But he never lost his enormous interest in science.

And I think when he saw that I liked it as well, and had some ability in it, he lived vicariously. And we spent an immense amount of time talking about science, going to science museums. He even took me on various business trips.

I remember particularly one to Chicago when I was very young, to see the Museum of Science and Industry, specifically for that, and to go to Boston and the museums. And of course we went many, many times to the Museum of Natural History. And he took me on a special trip to Washington, to the Smithsonian, and so forth.

House: How fortunate!

Birnbaum: And all through my life, until he passed away-- unfortunately quite a long time ago, but after I had finished graduate school-- he was always tremendously interested. He was a lifetime subscriber to Scientific American, and things of that sort.

So I came by that interest perhaps somewhat naturally. But certainly it was--

House: Certainly nurtured.

Birnbaum: Sponsored by him.

House: How wonderful.

Birnbaum: And he had a very clear life plan laid out for me. And because I liked him, loved him, and respected him, I didn't question it very much. And the plan was that there was no higher calling on earth than being a college professor, and particularly of science, and particularly at a great university where you could also do research.

So his dream for me was that I would become a professor of whatever science I chose. And when I chose physics, he was particularly happy about that. And so when I went to graduate school, the only thing that was in my mind was to get my degree, and then get the best teaching job that I could, and eventually work my way over to some great research university.

House: So you described going to Cornell and then to Yale. But were they all in physics?

Birnbaum: Yes. It happened that I was a good high school student. And I got about the same grades in everything.

And I guess I wanted to be an engineer slash scientist. I didn't know the distinction very well at the time. And I had my father constantly egging me on.

But I also had a guidance counselor who was neither. And he thought I would make a good lawyer, or maybe a good doctor, or maybe a good businessman. And he recommended that I take the set of aptitude exams at a place in New Jersey called the Stevens Institute, which in those days was known for providing this service that you paid for. And so I went for two days of College Board like exams, and then a meeting with the person who was going to tell me what the exam said about where my best career might lie.

And as I had done in high school, I did about the same on all the tests. And he said, well, he said, I don't know. He said, this country needs engineers and scientists a lot more than it needs more lawyers and more businessmen. So I think you should go into it. But as to whether you should be an engineer or a scientist, that's a much harder question.

And he said, are you aware of a program that is offered at several schools that's called engineering physics? And I wasn't. So he provided-- that was a very good service. And I said, well, what is that?

And he said, well, it takes different forms. At MIT it looks like this. At Cornell it looks like this. Caltech was a school that offered it.

So he said, I would apply to those schools. You'll probably get in them-- good grades, and so on. And he said, the benefit of those is that you get close to an undergraduate education in engineering, but you then get to specialize, your the last year or two, in either a particular type of engineering, or you can go into science.

He said, most of them are five year programs. And many of the engineering schools in those days were five year programs, anyway. So I said, well, which one do you think is the best? And he said, oh probably Caltech.

And so I applied to Caltech, and MIT, and Cornell, specifically to get into the engineering physics. And I also applied to Harvard and Princeton. Turned out I got into all of them.

So I went around with my dad to see the different schools. And in those days I had three interests-- one was science, one was girls, and the other was playing sports. I played a number of varsity sports in high school and college. And so Caltech failed on two of the three.

And I said, science ain't that great. And I'm going to go to a place where I have to be on a Frisbee club or something. I want to play ball.

And I had a-- Harvard and Yale. I also applied to Yale. So I went to all of those. And I really liked Cornell.

Because, first of all, it had a really beautiful campus. It had lots of girls that were right there. You didn't have to troop over to Radcliffe, or anywhere else. That seemed like a good idea. And I don't know.

I had been offered what was basically a baseball scholarship. They weren't allowed to call it that, because it was the Ivy League. And I really liked the baseball coach. And I liked the stadium, and all that.

So for absolutely crazy a set of reasons, I chose to go to Cornell. I also had a New York State Regents scholarship, and a Cornell scholarship, got a scholarship and so on. So I said, OK, fine.

My father was horrified that I had gotten into Harvard, and Yale, and Princeton, and I was going to Cornell. And he said, are you out of your mind? What's the matter with you?

You must be insane. How do you know what you're going to do later? And when you go for jobs, it's not in the same league as the other schools, and all of that.

But I looked into it a lot. They had a very, very strong physics department, where I was already leaning-- extremely strong-- and a very, very good engineering school, much better than Harvard and Yale, and probably as good as Princeton. So I decided to go there.

And it was a wonderful choice. I was very happy there. I did play ball. I did meet my wife, my first wife there. And I felt I got a very, very good education.

Starting in about my third year, I opted for nuclear physics. And I took basically almost all the graduate courses in nuclear physics as an undergraduate. So when I went to Yale, in nuclear physics, they made you take the courses anyway. But I finished all my course requirements in one semester, and was basically extremely easy compared to Cornell. I think Cornell was a far superior school to Yale in those days in physics.

You didn't ask me this. But I guess I should tell you. Where did computing come from?

House: Well, before we go there, let me just pursue the baseball. Because I recall a story, and it may be apocryphal. That you actually flirted with the idea of a baseball career--

Birnbaum: Oh yeah. If I'd just been a little better, I would have been a baseball player. I was All Ivy. I was the pitcher. I was seven and one.

And we tied for the Ivy League Championship my last year. I had a couple one-hitters, and so on. So I was a pretty good college pitcher for that level of college ball.

And that got me a couple of major league tryouts. And I had all sorts of wonderful notions about how I was going to spend the next 10 years of my life, and subscribe to the Scientific American. But my father was absolutely violent about, I scraped and saved and sacrificed-- which he had-- for all this.

And you have a chance to be a college professor, and you're going to be with a bunch of guys that read comic books, and don't use words of more than two syllables for the next 10 or 15 years? No way. You do that, you're out of here.

But nevertheless, I went to the try out. I tried out for the Yankees. And they just said I wasn't good enough. So my father was ecstatic.

It was a kind of an interesting story. Because the way that the Yankee tryout worked was they divided people up. There was about 100 guys there. And they divided you up into little groups. And then you

essentially, the pitchers went out to the mound. And they pitched to six batters. And it wasn't a game. But the batters ran out whatever they did.

So in other words, there were people in the field and they would-- so it just turned out, I don't know why, that I was the first pitcher of the day. And I think it was the best I ever pitched in my life. I struck out five of the six guys.

House: Really?

Birnbaum: And the sixth guy had a swinging bunt. I was not the greatest and most nimble fielder. But that day I picked it up. I handled it, threw the guy out at first base. I said, shit, I'm in! I'm in! OK, here I go. Unfortunately-- so then they cut the 100 guys down 50. And of course they kept narrowing it down.

And the way they did it was they read out the names of people who should stay, not the ones who should leave. And none of the other pitchers that came after me did remotely as good as that. And he didn't read my name.

So I figured, geez, this must be a mistake. How could this be? What can you do better than that?

So I went over, and I talked to the guy. And I said, excuse me, I didn't hear my name, but I thought I did pretty well out there. What happened?

And he said, oh yeah, against these palookas. He says you throw that 90-plus-whatever mile straight fastball to a major league hitter, we'll be getting out the tape measure for the 500-foot home runs. And he said, you don't really have any concept. You're just overpowering these guys with speed.

You're not really pitching. You're just throwing. I guess that was probably true.

But it didn't have any action on it. And that's what they were looking for. And so they sent me home. And so I became a physicist.

House: OK, now time to go to computing.

Birnbaum: OK. Well, in my last term at Cornell, my last year, which was a fifth year, there was a guy there that you probably have run into from time to time in your reading, named Richard Conway. And Dick

Conway and his pal, Bill Maxwell, were great guys in industrial operations research, management, and so forth. And he taught one of the very, very early computer courses, computer programming courses.

There was no computer science department, or anything. But he taught this introduction to computing. And he taught it on a Burroughs something or other 550, Datatron.

And I never knew anything about computers or programming. But I figured that people told me was a wonderful lecturer, which indeed he was one of the best lecturers I ever had. And it was a really, really inspiring course.

And I took to it instantly. I found myself not only doing the required problems, but doing the extra ones. And I just loved it. I got a 100 in the course.

And in some time around in February or March, I went to see him. I'd never spoken to him before that. And I said, you know, I really love this. And my plan, I've been accepted to Yale. I'm going to graduate school in physics.

But I really-- could you think of some kind of summer job I could get where I could do some more of this stuff? And he said, well, best summer job you could get in the New York area would be to go to work for IBM. And their programming headquarters at the time were in the Time Life Building, which is on Sixth Avenue in New York City. They had several floors there in a big data center downstairs.

He said, I don't think you're going to get a job. Because it's really late. They fill those jobs a year in advance, and not that many of them. But if you can get a summer intern job, if not at IBM, try one of the other computer companies. That's what I'd recommend.

So it was right before spring break. And I went into the Time Life Building. And I signed up to-- and they immediately told me, no, all the summer slots are filled.

But as long as you're here, why don't you take our aptitude exam? And we'll keep it on file for next year. I didn't know what I'd be doing next year. But I figured, OK, fine.

So it was one of these really crazy sort of tests. it was something like, it was supposed to judge programmer aptitude. And the first thing was a vocabulary test. I'm not sure what that had to do with it.

But I guess it was to see if you knew anything about language. And it happens I had a reasonably good vocabulary. So I found that easy.

Then they had these little math problems of the sort you get in the college boards. Well, I was expert on those. And then they had a crazy test was really hard, in which they showed you a picture of, like, two hands and fingers all intertwined. And you had to say whether it was the right hand or the left hand.

And I thought it was really hard. But I kind of sitting there doing it, and I chucked it all off. I turned the thing in, and I said, what the hell has that got to do with programming? I didn't understand.

Anyway, I was packing up my stuff. Of course, they scored it immediately. Because it was one of these fill in the blank things.

And they handed me the score. And it said 100%. And the secretary said, would you just wait a minute? A guy, whose name was Steve Nicholson-- I remember, because he was the head of the whole business-- later he worked for me.

House: Irony.

Birnbaum: Yeah, as it turned out. But he was a very good guy. And he said, we'd like to talk to you.

So he called me in the office. And he said, we've been giving this test for three or four years. We've never had anybody get a perfect score before. I just wanted to meet you.

So he said, “Tell me about it. How did you do this?” And I said, “Frankly, it's like the morning section in the college board. It was really easy.”

I said, I don't know about this left hand/right hand thing. Oh, there was also syllogisms. This is to this as that is to that. Easy stuff.

So he said, well, he says, we don't have any spots for this summer. But I kind of hate to let you get away. Would you be willing to be on my staff for the summer?

And I said, yeah, absolutely, great. So I didn't know what that meant. I said, what would I do?

He said, I don't know. He said, I can't think about it now. I'll think about it. By the time you get here, I'll have a job for you.

So I did. And the job turned out to be the (would go??)-- I was writing library routines for the big IBM operating system that was called IBSIS at the time, 7090 class machines, in Poughkeepsie. So one or two days a week, I would drive up to Poughkeepsie, and run on the machines at night, in the third shift. And I was supposed to finish these library routines by the end of the summer. And I did.

And he liked what I was doing enough that he said, would you come back again for another year? And that year I joined a rather famous compiler group that they had there. And the thing that was interesting about it was that I was about to be a really, really lousy programmer.

I probably was about as bad a programmer as you could possibly be. My programs hardly ever ran properly. And even when they did, they ran slow. And even when they ran properly, slowly, they took up twice as much memory as a really good programmer would do.

So the programming aptitude test was a total sham. And it didn't measure anything that had to do with programming. It measured whether you had a good vocabulary, and which was your right hand. And as a result of how bad a programmer I was, they did away with that whole testing procedure.

House: Is that right?

Birnbaum: Yeah, later on. But anyway, I was valuable to the physics guys at Yale. Because nobody there knew anything about computing. And they were getting a new accelerator, and a so-called tandem Van de Graaff, the world's first.

And they needed to put computers on it. And they didn't have anybody who knew how to do that. And they also didn't know how to reduce the data using the existing computer center.

And so I kind of got to be the physics department computer guy. And unlike all the other graduate students, who had either teaching assistantships, or fellowships, and so on, mine was converted into a computing assistantship, in which I sort of was available to everybody to help them with computing. And I was ordered to work for IBM in the summertime, as opposed to being allowed to leave the campus. So I worked for IBM for all four years of graduate school.

House: So stop here. And tell me, when you took the first internship, when you passed this hand test, what year was that? Would that be 1960, roughly?

Birnbaum: 1960.

House: OK.

Birnbaum: 1960. So it was for four years. By the time I finished, System 360 had just come out.

House: OK.

Birnbaum: So Fred Brooks was in charge then. And I learned that I didn't have any training at all. I'd never taken a computer course. I started reading books about logic, and so on. But I hadn't really studied any of that, either.

And I was a really strange guy to be this sort of so-called computer guru. Because I didn't really know very much about anything, except what I had taught myself. But I did OK.

And when it came time to graduate, I was so in love with computing and computers that I was really sort of wondering whether my father had had the right dream for me. And I started saying to him, how would you feel if I became an engineer, and I went into computers? And his face fell. He said, computers, well, gee, I don't know. I guess so.

But I'd gotten quite disenchanted by the whole academic ladder business. I'd seen what was involved in becoming an instructor, and an assistant professor, and how hard you had to work, and how few tenure slots there were, and how many years it took. And I'd built up a big debt from graduate school, gotten married, had a couple kids in the meantime.

And so I decided that I couldn't afford to be a professor, that I would come back to being a professor, after I had worked somewhere, ideally Bell Labs, and published a bunch of papers. So that I could come in with a tenure track, maybe five to 10 years later. And I explained to my father that that's what I was going to do.

And he kind of grudgingly said, well, I guess you know best. But, boy, I don't know. Sounds crazy to me.

So IBM offered me a job at the research center. And by coincidence, they were interested in getting into real time computing, which they hadn't ever done any of. And they were looking for a joint project. And my thesis adviser and I cooked up a project by which we would automate this tandem Van de Graaff that I had been helping to prepare the ground for.

And it was a three-way joint project, between IBM, the Atomic Energy Commission as it was called in those days, and Yale. And I was in charge of it. And we had about six or eight guys, some of whom went on to become pretty well-known later. They were just young guys just out of school that I hired.

And we built a new computer. We built a new language. We built new interfaces. We built new A-to-D converters. We built a bitmap display, probably one of the world's first.

And we automated the accelerator. And one of my, I think, the things I ever did that I was probably the most proud of is that that lasted about 16 years. It became-- it was so good as a data gatherer interpreter that they were able to crank out something like 90 theses in something like 10 years.

And 12 years after we finished that project, Yale produced more PhDs in nuclear physics than any other school in the world. And it was mostly because of that 360/44 that we built a new channel for, and a new front end for. And out of all that came an instrumentation standard for the world, which I'm sure you've seen many times.

But it's the modules that plug into, that creates the (???). They were called NIM modules to start with, Nuclear Instrument Modules. That had been developed at Oak Ridge and at Los Alamos, and places like that.

Well, we took those, and we digitized them. And that became, eventually, the European standard called CAMAC. And so I was the Chairman of the CAMAC Committee. And we did all that, so on, so forth.

So after three years, I was offered a faculty position at Yale. And so here was my dream coming true, way ahead of time, to teach physics. And I said, oh my God, who wants to do that? So I said, no thanks. And I went to Yorktown.

And the project had been successful. A lot of IBM products had come out of it. And the AEC had voted it their outstanding project of the year, and so on, joint venture. So they made me the head of a new group that was called the Laboratory automation and Real Time Systems.

And we eventually grew to about 15 people. And we figured out ways to do time shared, real time, timestamp nanosecond scale laboratory automation on things like bubble chambers and cloud chambers at Brookhaven on scanning electron microscopes, of which we had one of the early ones at Yorktown. I worked with a lot of people on a lot of different kinds of instruments, which were sitting on a virtual machine based time-shared operating system.

A lot of very novel work-- I can't remember ever being happier than in those years. We had about six or eight of us.

House: You were like 28 or 30?

Birnbaum: Let's see. That was about '67 to about '70. So I was about 30.

House: You've come quite a ways.

Birnbaum: Well, I really loved it. And I still didn't know anything. But I had good-- what I found out about myself along the way was that-- it came back to haunt me many times. I'm sorry. This is a digression from your early question.

I would interview people for jobs. And by this time, computer science departments were appearing at places, or had appeared at places like MIT, and Carnegie Mellon, and so on. And guys would have had undergraduate and four, five years of graduate school computer science.

They'd written 10 compilers. They'd done this. They knew theory.

They could-- I hardly could understand what they were talking about. And I felt like such a fraud, like a total impostor. God, I'm a senior manager at IBM computer research, and I don't know anything.

I never took one of these things. I decided I better take some courses. But I didn't have enough time.

And yet I discovered that although they were very deep, they were also very narrow. And they didn't have good taste. They didn't have a good feeling for what to work on, what might work, or important if it worked.

And that somehow, without ever having studied it, I had somehow assimilated that from all these smart people I'd worked with all these years. And so we started a whole lot of different projects. I became the head of all experimental systems, research, at IBM, which was about 30, 40 people.

And then I became the head of the computer science research. And then I became-- if I had stayed, I would have been the head of all IBM research. So I was kind of an anachronism. And I worked for such a powerful computer company. Remember, IBM was as big as all the others put together at that point.

House: Right.

Birnbaum: When I left IBM, I had 500 computer science PhDs working for me. And just for scale, that's about 25 computer science departments' worth. Except that they were guys called [INAUDIBLE], on and on and on, a lot of your people on the wall-- Fran Allen, John Cocke, et cetera.

And I had all these many, many projects. But I always kept about three or four for myself, just because I enjoyed it so much. I just thought it was so wonderful. So that's how I came into computer science. I mean, I never really got to be a computer scientist.

House: You aren't a computer scientist. You're just a fraud. This is--

Birnbaum: I am a fraud.

House: That's great.

Birnbaum: Many years later, Barbara Waugh, the personnel manager at the HP labs, whom I know you know, Chuck, well, I was telling her. I said, this is really impossible. I just interviewed some kid from Berkeley. And this kid has forgotten more than I know. And I said, Jesus, how do I go home, and even look at myself in the mirror?

And she said, I got a book for you. And she gave me a book. It's a wonderful book.

If you've never read it, I really recommend it. It's written by a psychotherapist, I guess he is, or a psychologist. It's called *The Impostor Syndrome*. And it turns out that many, many, many people suffer from this.

The only reason I'm the anchorman is because I'm black. I never would have gotten a job if I wasn't a woman. Gee, they needed a token this.

I never knew this. I was just lucky. I had a smart guy that worked for me.

So all the things that people tell themselves about what frauds they are, and have so little self respect, instead of saying, I know enough that these guys want to work for me. And they listen to what I say. And it really was my idea, even if I didn't know exactly how to do it.

So that helped me a lot. And I stopped thinking of myself as a total fraud. Because I recognized that even though I didn't quite know what I was doing much of the time, it got a little better by the time I'd gotten to HP.

Guys would come in to me with these schemes, projects. And I could listen to them. And I usually could ask the two or three questions that they themselves would recognize would be the reason why in the end it wouldn't work. But anyway, that's how I got to be a so-called computer guy.

House: So just to stop you there. I think what you just said is so key. We try, at the museum, to educate the youth, if you will, about why these are interesting fields, and what to do, and how do you get in it. But this little soliloquy around, how do you develop self-confidence when you don't know is powerful.

Birnbaum: Yeah, I think so. You know, the American Physical Society, which I retained membership in for a number of years. And finally I stopped. Had a symposium when the people who were applying for degrees in the United States was falling precipitously. They were going into the biological sciences, and to other fields of physics, and into computer science, and so on. And they wanted to know what they could do about it.

And one of the ideas they had to have a symposium that had the premise that, if you got a really solid science education-- and of course they were featuring nuclear physics-- it prepared you well to do other things. And so they invited about 10 people whom they thought had had interesting other careers in other fields. They had a guy who became a famous doctor, my office mate of many years, who became one of the Apollo scientist astronauts of the first group. They had someone who started the rocket science, real estate sort of stuff, mathematics-- et cetera. And I was one of the speakers for that.

And so I had to really dredge up what it was that you really learned how to do. An I guess this will reveal something of a prejudice of mine. But for at least the first 10 years, when I was in a situation where I was hiring people-- and I've probably hired thousands of people over the years, I was far less interested in the computer science people than in just smart people who had enough mathematics and enough background that they could pick it up. Because at least in the early days, a lot of the computer science guys didn't have a good feeling for how you conducted a proper experiment, or how or whether you could create a general case out of a special thing.

They were really good at writing compilers. And they could tell you how the register should be set. But they couldn't solve a register coloring problem (???). Because they couldn't even understand what it was.

So I hired mostly mathematicians and physicists, and occasional physical chemists, and occasional aeronautical engineer, and so forth, just because they seemed like the smartest guys. And that was probably the best thing. I mean, I did it instinctively. But that was probably the best thing I ever did.

Because, if you look back over the history of early industrial computing, you'll find that a lot of the great inventions or innovations-- both, actually-- were not done by computer science people. And it's popular to say, well, there weren't any computer scientists. But it isn't really true. There were lots of `computer scientists. They were mostly working in AI, or something like that, or proving something NP-complete.

The guys that were figuring out disk drives, and chain printers, and how you made a network respond, and how you did a real time interrupt, and so on and so on were not computer guys. They were engineers in some cases. They were scientists in other cases. So I was sort of one of those guys, heading another one.

Once one of the enraged computer science professors called me, in a very large public meeting, a defrocked third class nuclear physicist. And to some extent, he was correct. But I always felt that having a background outside the field gave me a little bit of an edge.

It would've been great if I could have gone back, if I had ever had the time to go back and get a master's degree, let's say, in computer science, and really put in some of the solid fundamentals to practice the craft. But by that time I was a high enough level manager that it didn't really matter whether I knew it myself or not. I knew enough to hire the people.

House: I love the philosophy. Let's take the IBM time now. You were there, leaving aside the internship--

Birnbaum: '65 to '80.

House: '65 to '80. And you wound up running a major chunk of the research program.

Birnbaum: I was called the-- the official title was Director of Computer Science Department.

House: OK.

Birnbaum: But that included computer science and computer engineering. And it was basically most of the computer research in IBM.

House: This is the heyday of IBM computing. I mean, this is it.

Birnbaum: We were-- the computer industry was called IBM and the seven dwarfs by some wag. And it stuck. The revenues of IBM were as large as the next the seven companies combined.

And it was about-- when I left IBM, I think it was a $46 billion company. Number two was DEC. And I don't remember how big DEC was. But I'd be surprised if it was 10% of that, 15% of it[1].

House: So what were some of the projects that gave you particular enthusiasm, or things that you feel shaped you, or shaped the world, or both?

Birnbaum: OK. My early years at IBM were characterized by connecting things to computers-- instruments, in some cases scientific experiments.

House: So the real time element, and all that.

Birnbaum: And one of the very earliest influences on me was observing. I observed it at Yale before I ever came to IBM. And then I really observed it during the joint project with Yale.

I loved computers. I was having this love affair. It's lasted all my life.

Most people hated them. Guys that I thought of as immensely smarter than me, theoretical nuclear physicists, famous guys, a couple Nobel Prize winners, Willis Lamb, guys like that, they couldn't understand the simplest things about programming languages. They couldn't even understand the difference between a compiler and an operating system. And they didn't want to learn. They thought it was beneath them.

And when it came to having to specify to the computer how it was to run the accelerator, or how it was to analyze the data, they-- Fortran was just not something they wanted to be bothered with. Assembly language was like something that had been created by some kind of masochistic devil somewhere. There was no conceivable way they were going to lower themselves.

And so they would hire somebody out of a research grant to so-called do the programming for them. And of course it was tremendously inefficient. And they were stuck with very pedestrian sort of things.

So I kind of started thinking very much about why is it that I and many of my computer friends are having this love affair, and these other guys think that they're almost in the clutches of Satan? They have to resist all this. And I got started to think about user interfaces, and what meant user friendly, and what

[1] [Interviewer's note] Annual reports for fiscal 1980 show IBM with $26,213 M total revenue, $21,367 from computing products, and $3,561 M in net profits. DEC had total revenues of $2,368 M, HP $3,099 M. The BUNCH that year (the Seven Dwarfs had dwindled to five, and they had a collective revenue of $15,624 M {$2,857 (B); $2,552 (U); $2,857 (N); $2,765 (C); and $4,593 (H)}, not all of which by any means was computing. IBM did $40,180 M in 1983, $45,937 M in 1984.

meant ease of use, and the terms we came to years later. But all I was trying to do was, could we produce something that a physicist would find relatively intuitive, and that he could think of in terms of his experiment or his theory, and which could then, by the machine, be compiled down or interpreted and then compiled down into something which would make the machine run?

So very early on, both at Yale and at IBM, I got involved in user interface studies. And when I got to IBM, I found out that I had inherited a very tiny group of about three people-- I think it was 15 when I left-- of cognitive psychologists. Probably was the first group of people that were-- among the first groups of people. And was headed by a quite famous guy named John Gould, who was the president of the American Cognitive Psychology Association, and so on, and a guy named Steve Boyce (Boles???), and another few guys. They're famous guys in the field of computer interaction.

And they were studying user interfaces. And I was really interested in what they were doing. I showed them the two or three languages that I had created for the physicists, which was how all those theses got written.

Because they would say things like, take output three of analog four. Look for coincidence within this many nanoseconds with that. And if so, invoke this. So they were writing very high level statements which got compiled into whatever the proper languages were.

And they liked it. Because it was just a of rules, and words that they understood. It was easy for us to do that. When we talked to the microscopists, we would use the same techniques. But we would use a different vocabulary or them.

So these psychologists started looking at all this sort of stuff. And a lot of things came out of that. The one that got the most famous-- oh my god. I'm going to actually go on tape, and admit that I did this. I-- do you protect people?

House: Sure.

Birnbaum: Against vengeful assassins.

House: Yeah.

Birnbaum: I'm the inventor of voicemail.

House: Huh?

Birnbaum: I invented voicemail.

House: Oh my god.

Birnbaum: Yeah, exactly.

House: We're going to blame you?

Birnbaum: You're going to blame me. Every time you pick up-- you got to do it. So this is a story here.

Do we have time for a little story? It's interesting. Because it'll tell you what we did at IBM.

OK, so IBM in those days had, as an important business, the dictation machine business, which were these little things you carried around with these belts on them. And you spoke to them. And you gave them to a secretary. And she had a thing that she could do with a foot pedal. And she would type on a typewriter-- no word processors-- what you dictated.

I was a user of those things. Sales were flagging. A company called Lanier was overtaking them. The Japanese had entered the market.

They came to the research division. And they said, what can you do to make our dictation unit different than everybody else's? And so I had the idea, how about if you could dictate not onto the belt, but into a phone? And it went right to your secretary. And then she could have a touch thing, so that she could can play back the phone call, and so forth.

So we took an IBM series one computer. We hooked up to the phone system. It behaved like a little PBX.

It just sort of sat there. And the idea was you would be able, whenever you were on a business trip or anywhere else, you could send your dictation in. And that this would be a whole new thing.

So then the problem was, how do you use a touch tone pad-- touch tones were relatively new-- to control this thing. And so I called in the psychologists to work on that problem. So the psychologists figured out some clever things.

Think of the five as the fulcrum of a joystick. And then go forward would be 5 plus that number. This one backwards.

Use the key names for replay, dictate, all the things you knew about. Hadn't been done before. Wrote a paper on it. They wrote a paper on it.

And we started using the system. And we did a field trial. We did a field trial with the field engineering division of IBM.

They hated it. Secretaries hated it. Everyone hated it. And they hated it for a lot of different reasons, not the least of which was the loss of privacy, and so forth and so on.

And so people stopped using it. Except we found that they used it to leave messages. So they would leave a message for that secretary that says, hey, I changed my flight. Can you call the limo guy, and tell him to pick me up an hour earlier?

So suddenly it became a speech messaging system. And then we said, well maybe we could store the messages. And some people could dictate.

So to make a long story short, it became a project that was a speech filing system, done entirely by the cognitive psychologists, and a few computer guys that I hired to help them do the different tests. Immensely successful.

And someone said, well this ought to replace email. Because we were early users of email, using the VM system, which was Virtual Machine system we had developed. And she said, OK, we'll call it voicemail. And so IBM developed this voicemail thing. And it was supposed to be a product.

There was nothing like it anywhere in the world. It was supposed to be a product out of the Rochester, Minnesota Office Products Division. So they looked at it. And they decided-- we had given them some recommendations as to how much you should charge for it, and so and.

And they didn't want to do that. They charged an outrageous price for it. They had almost nobody that signed up for it. They didn't run a training school, which we had recommended, on how to use it.

And so after one year, it was a horrendous flop in IBM. And so the head of IBM came to Yorktown. And he said, we've got to get rid of this thing. We're going to sell it.

And we're going to give all you guys outstanding contribution awards, and stock options, and so on. But we're getting rid of the stuff. Please stop your research on it. So they sold it to a company called Octel.

House: Yeah.

Birnbaum: So the Octel, which was one of the first successful things, that became voicemail. We had on it features that even today aren't on voicemail. I could do fireside chats with the whole research division, for example. We had plebiscites for, would you like the coffee to come at three o'clock instead of two o'clock-- hit one for this. This was 1971.

House: '71?

Birnbaum: '71, '72, something like that, the early days. But the success of the system, in terms of its usability, got me to thinking in terms of projects in which people could use a computer as if we were an appliance, which is when I coined that phrase, without having to know how the appliance works. You don't know how a washing machine works. You didn't know how a typewriter worked. But you can learn how to use it.

Why couldn't we develop a set of interfaces? About this time, Xerox PARC was starting. I had read all the stuff from SRI, and so forth. I was quite familiar with all that.

And I said, why don't we just do that? Let's really see if we can make computers usable by ordinary people, to do things that they care about. Because that would make them really appliances. Name them by what they do, rather than by what they are.

All those things, you heard many, many years later in a different environment. But that started, all of that, at IBM. And so we did a lot of projects that had, as the underlying *leitmotif*, sort of the user interface for ordinary sorts of people. And ordinary was in quotes. It meant somebody who was a specialist, and an expert in something.

So for example, we did an interpret x-rays automatically for doctors, where the doctor was the ordinary person. And he had a whole vocabulary of how he would specify things. But he didn't know anything about the computer.

We did many, many projects of that sort. So I don't know which one-- the one that I was the proudest of was the query by example. But there were many others. We did a signature verification project that was quite unique for its time, in which we looked at the dynamic zero-crossings of acceleration of a pen in order to-- I'm telling you about the projects that were my projects.

House: That's perfect.

Birnbaum: Not the corporate projects, and not the--

House: We're interviewing you for you.

Birnbaum: And then we realized along the way that we didn't really understand how the computer was spending its time, and why things that somehow seemed to be very fast and easy for people to do, often things that related to pattern recognition or recognition of a behavior pattern, took computers forever. We just didn't have a good feeling for why that was. So we started a thing called the Experimental Systems Lab, which I think was the first one in IBM certainly, maybe the first one in the world. In the past, when the computer scientists wanted to test something that they had done, they had to get onto the main systems that were delivering service. And either run on them in a special privileged partition mode, which of course by Heisenberg principle immediately upset the experiment, or run in the third shift, or something like that.

And I made a case that i wanted a multimillion-dollar computer just to be used like it was our cadaver, or our guinea pig. And it would never do any useful work for anybody. We would instrument the hell out of it. We would take all sorts of traces.

And I started the IBM performance group, which got to be very famous. It was headed by a guy named Don [? Fraser ?] to start with, and then by Hisashi Kobayashi, who later became Princeton's dean of computer science. We invented the diffusion algorithms for measuring computer performance, and so forth.

So that was my other project, was in computer performance. But it was really in support of getting the performance, so that the computer cycles could be applied to make the computer natural to use for a person who was a domain expert, but not a computer expert. And we did many such projects.

House: So there couldn't have been very many labs doing that anywhere.

Birnbaum: I don't think there was any at the time. I didn't know of any. I later found out NASA did that. And I later got to work with NASA quite a bit. It turned out that on the moon shot, they had people doing the same sort of stuff.

And in fact, one of the things we did was we got the operating system from them for the manned orbiting laboratory, which was a real time operating system of quite unusual design for the time. And we adapted that for the system 360, which turned out wouldn't run on the 360 channels. And we invented a new channel, and so on.

So we did a lot of that kind of stuff. But it was very unusual to have a system which was so instrumented. And that's where I found out about HP.

Because-- that's where I found out about you, Chuck. Because we needed to have logic analyzers, and samplers, and sniffers. And Tony Lukes, who you met later, was part of that group, and so forth. These were people who had mostly electrical engineering degrees. Some of them had statistics degrees.

And they were experts at trying to gather enough data about what a system was doing that, by inference on different workloads, you can understand where the pressure points were, and where you could best invest your effort to improve the performance. That's where the second level cache got invented, for example, one of those projects. Again, I say invented by us. Maybe someone else did it somewhere else independently. But we thought of it as--

House: This is a fascinating thread you're painting. But you went quickly past query by example. You said one of your favorite projects. So why don't you elaborate on that briefly? Because that's really the core of some of this user interface perspective.

Birnbaum: Yeah. That's what changed my thinking about everything. Query by example was unique in several different ways.

One of the things that was unique about it was that it was non-procedural, which meant that you could ask the questions in any order you wanted. And you could fill in the variables in any order you wanted. And it essentially inverted the matrices for you afterwards.

So you didn't say things like, do a projection, do a join, do this, do that. It did all those things. But it did them so easily and so naturally that you could apply it to a relational database. Ted Codd, who had done the theory just fell in love with it instantly, understood it immediately. But it had bad performance, because we did it by interpretation.

House: Sure.

Birnbaum: The other question was, when we got in the dog fight, would be San Jose people, they said, well, it's not complete, in the sense that you can't do any query. It can be shown formally that this does not do what's called transitive closure. You know, logically speaking.

House: Sure.

Birnbaum: So I said, really? Well, Yorktown's got the best theory group in the world. That didn't report to me. But there was a young, super bright guy, student of Don Knuth and Manny Blum. He got a PhD. From both schools-- Berkeley and Stanford simultaneously-- named Ashok Chandra.

House: Yeah.

Birnbaum: Ashok Chandra later became the head of Almaden Computers.

House: I was going to say, he did just fine.

Birnbaum: And he did Bing after that. So Ashok worked for me. He was my staff assistant.

I just let him work on math problems. So I said to him, could you go away, and tell me whether this was that? And it turned out to be a really, really hard problem. He worked on it for quite a while. And he eventually proved that it did.

So computationally we had an advantage. In terms of the ease of use, we had an immense advantage over SQL. And in terms of its extension, because the things that could go into the boxes didn't have to just be variables, they could be whole equations, or even whole systems.

And we built such things. We built office by example. We built many, many different things. It was just a beautiful project.

But unfortunately, as I said before, IBM didn't think it could support that project as well as the whole System R thing. And they legislated against it. Chandra eventually left IBM.

And he went to work for a software company. I can't remember it's name. It started with an F, a big one. (Wasn't Verity, was it?)

And they made a similar sort of project. And then it got to be-- they started another company called exemplary??? (Autonomy?). But I learned a lot from that. Because I learned that if you could do something by example, where you could fill in the blanks in a phrase that you understood, this gave you a tremendous head up.

And if the computer wound up using a gigantic amount of cycles, who cared? And now the question was, build a computer that had enough cycles to do that. Of course, that's what I spent the bulk of my career

working on, was making computers faster and faster. But it all came out of that driving force to try to do things that would--

House: I'm hearing really two threads for really this whole IBM time. One is this user centric view. And I'll use user centric view, as opposed to user interface design. Because that's the derivative of taking the user's view.

And the second is, how do you get enough performance out of the machine to really support that kind of background behavior, right? And so one derivative might well be RISC architecture, and that kind of thing. But it's really the question of, how do I handle the complexity as cleanly as possible, or something like that? Am I right?

Birnbaum: That's correct. And for several years, the thread that we followed, because it was the one that was being written about by most of the computer science departments, which by now were springing up with many, many good people in them, was to make a machine which you programmed at a very high level, and which was well matched to the problem you were trying to solve. So in the sense that Symbolics machines were written to run LISP very efficiently, which was a language that was well matched to a certain class of symbolic problems, and [INAUDIBLE] machine, and this machine.

So the question was, maybe we should build a Fortran machine. Or maybe we should build a BASIC machine. Or maybe we should build this machine. How high can we push the user interface?

And this idea get started in my group at Yorktown. Unfortunately, it really took hold. Because it was a bad idea.

And the company decided that it would march into the future, doing what it called Future Systems. And Future Systems was the largest program of its type in the world. And eventually it course a billion dollars, approximately, of IBM, although they never admitted quite that number publicly. But that's about what it cost. Involved about 10,000 scientists and engineers at IBM for about four years.

And the idea was, you build a machine which, through particularly complicated vertical and horizontal microcode, could take these very high level application-specific instructions, and compile them directly onto the hardware, so that they would run immensely faster than having to go through the microcode. It's called a direct execution model. Had one the first one level store in order to accomplish that. Wonderful, wonderful people, legends in computing, worked on that program.

Now my best friend at IBM, and probably the most inventive and smartest guy than I ever personally have known is John Cocke, who is a polymath genius of many sorts, a very unconventional man, and one of

the earliest IBM Fellows. Basically won every award there is to win in computing, and beyond. And John said, I don't think so. I don't think it's going to work.

When John said I don't think so, you had better listen. And John had the office next to mine. We had lunch every day when we were both in town, often had dinner besides.

He spoke in strange disconnected non sequitur seeming sentences. And eventually I got to be about the only one that could understand. Because he'd pick up a conversation a week after he had ended in the middle of a sentence.

I kind of got good at listening to him. And not all the ideas were right. But a staggeringly large percentage were.

And by no means did they have only to do with computers. He was very, very good physicist, superb mathematician. He had a PhD in mathematics from Duke. That was his real training. But he knew everything.

And John said, I don't think so. I don't think it's right. I said, why don't you think it's right, John?

After a long, tortuous examination, he told me that his great friend at Stanford, Donald Knuth, one of the most famous computer scientists ever, author of the famous series of books, had written a paper which he had shown a galley of to John, which was an analysis of the instruction frequency execution in a set of Fortran programs. And that it completely-- with, by the way, the same objective-- trying to figure out which high level instructions would be worth building into the machine. And he had found, across all sorts of jobs, that you never executed these fancy special instructions. You only executed the simple ones-- the adds, and the shifts lefts, and the compares, and the multiplies.

And no matter how wonderful the high level instructions were, you couldn't write a program that would execute them frequently enough to do it. Particularly since, on average, every fourth or fifth instruction would be a branch. And many of those would be a branch whose direction could not be easily predicted at compile time, but was a runtime branch dependent on previous results.

And so John says, I just don't think it's going to work. He said, I think the idea is not to build a more complicated machine, but to build a simpler machine. And we talked about that for a long, long time.

And in my mind, the reason for talking about it was, first of all, my god, IBM is spending a billion dollars going in the opposite direction. And we're research. And I'm the strategy monitor. And I'm supposed to be telling them they made a mistake.

Well, we became so convinced of it that I started appearing at various corporate meetings, saying, I don't think so. I don't think it's going to work. You're putting all this--

And sadly, in the end, after several years, they finally admitted that it didn't work. It almost killed IBM. Because they had no new machines in the pipeline.

They came out with some machines that were interim machines that they basically sold lower than normal profit, until that came out with 370s that had a lot of the new stuff in it. So John, in the meantime, kept thinking about this idea. We kept talking to the group.

And John was, as far as I know, the inventor of optimizing compilers. He did it with a guy named Jack Schwartz at NYU, who worked for us during the summers, and Fran Allen, who was Jack's student and later wife.

House: Oh really? So they married?

Birnbaum: Oh yeah.

House: OK. Still married?

Birnbaum: No. But he died. They had gotten divorced before then.

House: OK.

Birnbaum: He ran the Courant Institute at MIT.

House: Oh, I did know that. Yeah.

Birnbaum: And we had a lot of very talented people in the optimizing compiler group. And we started out with simple things, like dead code elimination, and redundancy. And we got eventually to the point where we thought we could build a global optimizer, and then optimizers that would look ahead, and

[INAUDIBLE] branches, and all the things that are taken for granted today. But it was new science at the time. Grew that to be a quite large group.

And John was just all over it, all the time, always thinking about this. And meanwhile, in the experimental systems lab, on our built in computer that was our own, we could do whatever we want on, we started running our own workloads to see if we could reproduce that in Fortran. We ran them in COBOL. We ran them in PL1, and we ran them in this. Same for everything.

House: Wow.

Birnbaum: 70 to 80 instructions got executed 95% of the time. And you could have an instruction-- we used to joke, John used to say, put in an instruction that said, turn out the lights, stand up, turn around, flush the toilets, and go on to the next. And it would do everything in one, as he called it, foul swipe, as opposed to fell swoop.

But you maybe only execute it once a day. So save your 10 cycles. Who cared?

And so we got the point where we started thinking that all these complicated instructions would best be done in a single instruction. And that the best kind of microcode was no microcode. This was heresy.

And the whole basis of system 360 was that the microcode hid the differences among the machines, so you could use a common programming language and supposedly the same operating system. It really wasn't. It was too inefficient. And the code would scale across the machine. It was a wonderful idea.

But in fact, the price you paid for it was an incredible rat's nest of microcode. It was really slow, and really hard to fix. And that led, in fact, to a whole separate lab that I started, a whole group that became quite a good group, which did the early verification of microcode, using formal methods. [INAUDIBLE] a lot of the guys that became founders of that field later went on to be done. And we were trying to verify microcode, just because the 360 were becoming so impossible to deal with.

So John's idea, the fundamental idea, was no microcode. Every instruction operates in one cycle. And we only put in the most basic instructions.

Now the question was, could you build such a machine? What would it look like? And that's what became RISC. RISC is a horrendous misnomer created by Dave Patterson at Berkeley.

We called it RCC, which meant Reduced Complexity Computing. And that's what it really was about. It turns out that to accomplish the reduced complexity, you have a reduced instruction set. But that it's not a goal. That is an artifact of what happens, as you know.

So that's sort of where I was at. And there's a third leg to the stool, that I should tell you about. Which was that I inherited a project that I fell in love with.

And it was my own private project, even though I was a director. I wasn't supposed to have projects. I just liked it so well that I made myself the manager of it.

And I had these guys who sort of managed themselves. They were senior guys. We called it the Modular Computer System, MCS. And it was one of the very early distributed systems.

Carnegie Mellon was doing something called CM Star, or CM whatever they called that. There were other experiments going on at MIT, at Stanford, and Berkeley. But this was IBM's little experiment.

We used IBM series one, terrible 16-bit minicomputer. But that wasn't the point. And a lot of the bus structures of what became similar to ethernet later on, collision avoidance, and so on.

And we found that having 10 of these machines attacking different jobs in parallel, and being connected to the time sharing systems and similar, gave us immense advantages, immense, of all sort, that all were used. It was probably the most unpopular project at IBM. And the president of IBM he told me to stop the project early. Stop talking about it.

Because, he said, let's just imagine that you're successful. He says, what you've written is a blueprint for the failure of mainframes and time sharing, which of course was where all the money was coming from, and storage. Because storage was shared on the system as well. And the blueprint for the success of Data General, and DEC, and all the companies like that that are saying similar sorts of stuff. So stop!

And I said, but it's wrong. This is the future.

House: Cary?

Birnbaum: John Opel.

House: Opel, OK.

Birnbaum: He was president of IBM. And I said, doesn't it bother you that people are going to be working on things that in the end are going to be disruptive to you? And he said, no.

He said, first of all, it may never happen. And we've got such economy of scale here that we will always undercut them. We will always underprice them, and we will always beat them. And I said, I don't think so. I don't think so.

House: OK we're picking up again. And we're right at the juncture of your leaving IBM. You're thinking of-- no, you weren't thinking of leaving IBM. But what happened is you're open to invitation. Because you're pretty frustrated.

Birnbaum: A guy had been calling me, who was a headhunter hired by HP, to help them to become a computer company, something they had been reluctantly forced into, against the wishes of both Hewlett and Packard. Because digital controllers had become important in the instrument business. They had bought them from other people. They turned out to be weak spots in the reliability of their famously reliable instrument lines. And so they felt that they needed to make at least instrument controllers.

That had turned into three or maybe four different fledgling computer lines, each of which had some merit, but none of which were connected to each other. They were smart enough to recognize that although at the moment those things were doing OK, they didn't have the resources to support all of them. Because there was very little common software and hardware among them. Because using the time honored HP "every tub on its own bottom" division model, they all had different power supplies, and different standards, and different computer languages, and different operating systems, and so forth.

And so they had decided that they would start the computer component that was at a higher scale than the one that was currently in the research labs, HP labs. And at the same time, the company was embarking on a big project to unify the separate computers into what would become one computer family. For scale, HP was somewhere in the neighborhood of $2 million a year at that time, maybe a little bit more. More than half of it came from the very successful and original part of the company, which made instruments of a variety of sorts-- electronic, medical, analytical, chemistry, and so forth. And the other part came from these different computer lines.

They decided to interview six different people for the job of starting a computer research laboratory. And I was one of the names. The guy at Bell Labs was another name, and they guy from DEC was a third name, and so forth-- obvious people. And so this guy had been approaching me. And I basically had no interest.

I said, I use an HP scope every day. I use HP instruments. It's an instrument company. Why would I leave IBM?

Well, we want to become a computer company. I said, excuse me. You may want that.

I'd like to have my own jet, too. That doesn't mean that we're going to get anywhere. So I had kept him at arm's length.

But after the succession of very disappointing complimentary refusals to bring to fruition a lot of very nice research projects, and at the same time IBM not allowing us to publish some of those-- because they said, well, we just don't want this out in the public that we're doing this stuff, or we don't want to give aid and comfort to the enemy, so we'll just keep it on the shelf. So you couldn't publish it. Couldn't get any recognition for it. You had to sit by and watch other people reinventing what you'd already done.

So I was ready to do something. But I still wasn't interested in HP. If I had gotten a call from Amdahl, or I had gotten a call from a real fullscale computer company, I probably would have gone.

But as it happened, my wife was an opera singer. And she was still singing pretty much of that time, although she was talking about retiring from active touring around the world. And there was a potential job for her in San Francisco. So she was thinking about taking her first trip ever to San Francisco.

And I said, gee, free trip to San Francisco. I'll go see HP. I had read so many wonderful things about the company. It'll be fun. Not really serious about job. But I will just go out there.

So the headhunter called me. I said, OK. I went. I was sixth of the six people.

I met and was subjected to the famous Barney Oliver interview, in which he basically felt that whoever gave you your PhD exam hadn't done a good enough job, that he would do his own. We reached a stalemate. Because I felt-- I was 38 years old at the time-- I didn't need another PhD exam, let alone from him.

And so I started asking him questions. And in the end we both laughed, and became friends. And it turned out that HP offered me the job.

Eileen really liked California. And we decided that if I could structure it as a job that had a chance of being successful, maybe I would test them to see how they felt about these main threads which IBM was so unfavorable towards-- the user interface, high speed computation to provide the resource to make the interface as invisible to the user in terms of intricacies, the networking that eventually led to a utility or a national infrastructure, and most importantly for me at least the idea of the appliances, the things that would be particularized for a particular kind of use, to do one job, and do that job really well.

And they loved it. They loved all those ideas. They said, come do that.

And then came a lot of negotiations as to how many people would it take. How long would it take? I said, this is research. You don't know where it's going to lead us. We don't know whether it's going to come out.

Well, it sounds great. We'd love for you to come. And I got convinced enough that there was at least a fighting chance that this scrappy little company full of really, really great engineers and smart people and visionary management at the time-- Hewlett and Packard were retired, but still very active.

John Young was a few years into his job. But he wanted very much to do this. He was very supportive. Met a number of other people that I liked.

And I decided, OK. What the hell. We'll roll the dice.

And when I told IBM that I resigned was when they told me that now I was in charge of everything. And I said, well, that may be. But the guy I work for, Ralph Gomory, is one of the best executives I've ever seen.

And he couldn't make it happen. So why would I be so foolish as to think I could? You're just not structured to do this.

Well, what would you like? Well, I'd like to be able to make RISC machines. Call them IBM machines, or call them something else, and compete with your machines on an equal basis with my own sales force, my own manufacturing.

And I said that to my second level manager, who was Frank Cary, because the research division reported indirectly to the president and chairman of the board. And he said, well, there's no reason not to do that. We bought some land in Somers, New York. We could easily put another building on it. It's only five miles away from our research center.

How many people do you think you need? I said, I thought you might ask me that. 400, about half of them from research. They're all set. You don't even have to hire anybody. And about 200 that would come from manufacturing and marketing.

And I think we could break even. And he said, how long would it take you to produce a business plan? I said,, here it is. Because of course we were working on this.

And he said, I like it. I'll bring it up to the board. We have a meeting later this week.

And you call Hewlett Packard. Tell them that you've got a new offer. And you need a couple weeks to consider it. So I did.

And the board said, go. Do it. I didn't know that when Estridge and company in Boca had done the PC on just those terms-- that was a secret even for me-- systems monitor, and head of computer research, I didn't know the existence of that. I had even--

House: So just like the ESS1 thing, they did a lot of these secret things, it sounds like.

Birnbaum: Apparently.

House: Several.

Birnbaum: That was the first one. And they were not so far from going to market with it. This was the end of '79.

It came out in early '82. So they already had running prototypes. And they could see how it was going to work, and so forth.

And it was just that way. In the beginning it had its own-- later on, they integrated it into Atlanta, and it became a thing. But not in the beginning.

So he didn't tell me that. He didn't give me that information. He just said, we've always wanted to try something like that. He said, we can do it.

So I said, what's involved? He says, why don't you just call HP, and just tell them no? And he said, I'll write it down for you. Wrote it down on a piece of paper, 400 slots, you do this.

Give you five years. Make something out of it. Guaranteed funding for five years. He was a man of total integrity. I had no reason to doubt him for a second.

And I said, OK. I'll call HP. And he said, congratulations. You're a vice president of the research now.

He said, it won't actually happen until the end of the month, when we have our next board meeting. It won't happen at this board meeting. It'll happen at the one after that. But--

House: Done.

Birnbaum: So I called John Doyle, the head of personnel at HP, who had been acting as the intermediary for my two visits I had made. And I told him. He said, oh dear. And he said, well, I can't blame you.

Boy, talk about risk reward. You get everything you want. And you've already hired the team. You got a $50 billion company and a CEO behind you. Wow.

Good luck to you. It's been nice knowing you. But do me one favor.

He said, don't say yes. He said, I have to tell Dave Packard about it. Because he really took a shine to you.

I said, what are you talking about. I haven't even met him. In my mind, I said, I never met that guy. I met Hewlett, but Packard was out of town.

He said, he's going to give you a call. So just, he'll call you in a little while. Well, he called in about 10 minutes. So he must have gone running right over there, or something. So in 10 minutes, my phone rang, Dave Packard.

And big drawl, and he said, well, you know Frank Cary and I were on the board of trustees at Stanford together. And he's a wonderful man. And if he tells you that's what he wants to do, you can go to the bank with it. He's a 100% gold standard executive.

He said, so you'd be kind of crazy to accept our little offer to try to start a new business here. He said, except for one thing. He said, this is my company. It's got my name on it, mine and Bill's.

And he said, I know if I try to do something like that in my company, I wouldn't have a chance. The employees would never let me take their best people, and set up a new operation that were going to be directly in competition with what they had spent their whole careers building up. He said, I don't think he can do it.

He said, I know I couldn't do it. And, Christ, my company's a little peanut compared to IBM. IBM was several hundred thousand employees at that point.

He said, so let me make a recommendation. I'm flying to Washington on the corporate plane with John Young, our president, in about two to three weeks from now. How about we just divert the plane over to White Plains? Isn't that near where you are?

I said, yeah, it's right near there. He said, yeah, I landed there once. He said, and you come have dinner with us.

And he said, you spend the next two or three weeks trying to make it happen. Identify the people. Try to line it up. Try to get the budget. Try to get the manufacturing clearances, all that stuff that you need.

And if it's going well, and you think it's doable, and so on, we'll have a nice dinner, shake hands, no hard feelings. Nice to know you. We'll go move down our list to one of the other guys. He said, but if you don't think it's going to work, we'd really love to have you. And he hung up.

I said yes, OK. And of course, to myself, I said, you son of a bitch. Because I knew exactly what was going to happen.

And of course, I tried. And Frank Cary put me in his chief of staff. And he put me at the head of Poughkeepsie, and the head of Kingston, and the head of Boulder, and the head of Tucson, try to get the guys I knew I needed to make everything-- the manufacturing guys, the quality control guys, the shipping guys, make a business, startup.

Are you kidding me? Are you joking? We've got these two new guys we hired out of Podunk University. You can have them about a month from now.

But no way. I don't care what Cary says. Is he willing to excuse me my profit targets for this quarter? I don't think so. No way.

So it was a nightmare. It was an absolute nightmare. Then I went over to look at the site. And I went to see the construction guy. He told me I could be in there in about a year.

He said a year? A year? What kind of the year?

It's going to take three years. And so I'd have to be in Yorktown. But Yorktown was already full and overcrowded. So I had to rent a place, blah, blah, blah, blah.

So by the time they arrived--

House: The plane, at White Plains.

Birnbaum: At White Plains, which turned out not to have Dave Packard on it. It was just John Young. And we went and had dinner. And I said, take me. I'm yours.

And when I came out here, I found that they had a very different idea than I did about what my job was. The job I was really hired for was to be the head of research in place of Barney Oliver, who was a year or two away for retirement, being about 67 or something at the time. And that sounded like a nice job.

I liked that. But they felt I had to earn my spurs by starting a computer laboratory under the direction of a center director, whose name was Paul Stoft. And that if the lab was successful, and if after three to five years we had a reasonable plan, then I could be a candidate for Barney's position.

House: Really?

Birnbaum: So I just laughed. I said, listen, all I have to do is pick up this phone, and call IBM. I'm back there in one second. They were very unhappy about my leaving, very unhappy, mostly because of what I knew, not because of me. And I said, that's a ludicrous risk you're asking me to take. No, I won't do that.

I said, first of all, I'm not going to work for Paul Stoft. I'm going to work for whoever is the head of the labs, presumably Barney. And I'm going to be a center, not a lab. And I'm not going to be a 250-person lab. I'm going to be a 250-person center in three years, with three or four laboratories that are dedicated to different things that we're going to do.

And I want it in writing that if that's successful then I'm Barney Oliver.

House: Yeah.

Birnbaum: So they said, oh gosh. I don't know if we can do this. I said, well, you know what, I'm going back to my hotel. Let me know.

So they called in about an hour. And they said, OK, you got it. So that's how the computer research center at HP Labs was formed. And that's how, three years later indeed, in 1984 I got promoted to replace not Barney, but the guy who-- John Doyle, who had succeeded him in the interim.

It was a wonderful time. I had, once again, left being a high level manager to where I could spend probably 80% of my time working with the technology guys on RISC, and on all the things that you needed to become a computer center computer company, which was doubly hard problem compared to the one at IBM of building the 801 machine. There were three or four issues, which I'll outline to you. Because they're relevant in thinking about how the companies differed from each other, and what I had to do.

So the first thing that was different was, I had to be backwards compatible with the three disparate product lines that HP already had. We had to take all four of them-- HP1000, the HP3000, the Rocky Mountain thing, and the desktop 9000 kind of thing, and the Motorola based machine besides-- the ones that weren't in the desktop. So I had to figure out to get all those together, in addition to being, of course, performance leader, and the thing to go forward.

In addition to which, HP's biggest and most lucrative customer set was running COBOL. And a lot of the experts said that COBOL couldn't run on a RISC machine. The COBOL is a language which has variable length instructions, some of which have changed as the program goes along.

House: Yeah, they're dynamic.

Birnbaum: Really bad news, OK. Second of all, the HP1000 had a dynamic channel, so that you could reprogram the channel in real time based on interrupts. Wonderful for process control, nightmare for an optimizer, nightmare.

Because you could not optimized globally before you ran. You have to have stuff to do it in real time. It wasn't clear that it could be done.

The other problem was, could you do real time interrupts? Because a lot of the operating system stuff that we had done hadn't worried about that. In principle you could do it.

We just had never done it before. So that was the first thing. Was, technically, could you even do this job?

But the much harder part for me, which turned out to be a great blessing, was that when it was clear that I was finally going to leave and go to a competitor that IBM didn't even think of as a competitor, because

HP was just a little tiny blip on their computer chart in those days, I was surprisingly, for my exit interview-- which is usually done by the legal guy at your own division-- they asked me to go to our world headquarters, and be interviewed by Nicholas Katzenbach, the former Attorney General of the United States, who was now chief counsel for IBM, and whom I'd gotten to know reasonably well during the telephone business. Because he was highly involved in that. And I was very surprised at that.

And he sat me down, very friendly. And we talked about it. And he said, look, I'm not going to insult you by telling you don't steal this, and don't do this, and don't do that, and you can't take this, and don't raid our people.

He said, I know you. I think you're a guy of good integrity. You're not going to do those kind of things. In any case, my saying it to won't make any difference. It's all in these contracts, exit things that you're going to sign.

He says, so what you're here for is for me to tell you that one of the most valuable things that you know is what not to do. Because you know all the trails that IBM has gone down, spent research and development money on-- like Future System, which wasn't public-- that don't work. And if you tell a competitor or an audience, oh we tried that at IBM and it didn't work, you're saving them years and potentially millions or tens of millions of dollars. And that's off limits.

So he said very nicely, he said, we can't have you forget what you know. And obviously to earn your living we can't tell you you can't use your best judgment of things that are in your head. But those things which are not only patents, but which are trade secrets or just proprietary knowledge from having been there and having been a strategy monitor and so on, you keep your mouth shut. Because if we hear that you're telling people what not to do, we're coming after you. And you'll be sorry.

House: By the way.

Birnbaum: Big rumpled bear of a man. He got up and loomed over me. I had never thought about it. It had never even entered my mind that I wouldn't be allowed. I wasn't planning to tell them, here's how you optimize the registers. On the other hand, I wasn't planning on not telling them, here's how you don't do it.

So I said, wow. OK. I understand.

So I went to HP. And the first thing I got confronted with was this big project, called Vision, which was a way of using a direct execution high level language machine with a one level store to integrate all the architectures-- exactly what I had spent six years at IBM fighting against, except that the IBM project was immensely more sophisticated, involved 10,000 people at its end, and it cost almost $1 billion.

So I looked at what they were doing in Cupertino, and Fort Collins, and elsewhere. And I felt like I had gone back to kindergarten. I knew all the places they were going to run into trouble.

And I could even see that they have done many things much worse than IBM had done them. And that they were going to run into even bigger problems. I just didn't know what to do.

And I had what was probably one of the most important meetings of my life, my professional life, with Bill Hewlett. I'd formed a nice friendship with Bill.

My office wasn't far from him. He was sitting in what became the research headquarters later. I was over in a nearby building.

And I told him my dilemma. I told him the story about Nicholas Katzenbach, and so on. And I said, I see this on all sides of me. I see people going down trails that I think are going to lead to dead ends, not just on the Vision project, but on a bunch of others as well.

And I don't know want to do. Because I feel awful not saying something. On the other hand, I think that I really, for ethical reasons, shouldn't say something. And I don't think it's right for me to say something, anyway. So to my surprise, he got very angry, or kind of angry.

House: Really?

Birnbaum: He said, what are you talking about? He said, of course you shouldn't say anything. He said, there's all the ethical reasons you shouldn't say anything. But you shouldn't say anything because, who cares if IBM failed at it?

What makes you think we won't come up with a better way? What makes you think we don't have better engineers, or somebody that would have a unique idea? And who are you to be so smart that you can look at something and say, we tried this, it failed?

No one else can ever do it. He says, do you think that's the way Edison operated? Do you think that's the way the science moves along?

House: I can just see Bill doing this.

Birnbaum: He was walking around the room, and jangling the change in his pocket. I'd never seen that side of him before. He said, so you keep your damn mouth shut.

I said, well, what do you think I should do? He says, well, what you should do is just ask the right questions. Ask them hard questions. And keep your damn hands off of them.

I bet you're going to get plenty of good surprises. So in other words, if you think something is going to be a performance problem, as I was just telling him about a particular performance problem, you should say to them, and what's the performance going to look like when you do that? And then get out of the room.

And if they can't-- if they're not smart enough to figure out that there's going to be a performance problem, that's their problem. But maybe they'll be smart enough, and they'll figure out a way around it. And you might get some very pleasant surprises.

And he said, I got 40 years in this company that tells me you're going to get more surprises than not. So I left. I felt like an idiot.

House: What a treasure.

Birnbaum: Yeah.

House: I mean, in retrospect, what a treasure.

Birnbaum: And I followed that for my whole career at HP. And unfortunately a lot of times I was right. I was right about Vision.

But a lot of times I was wrong. And people did come up with some good things, and some startling things. And the thing that eventually made HP a computer company is an exact example of that, which I'll come back to.

So anyway, we went in. I had vowed I wouldn't hire anyone from IBM. And for about two years, I only hired one guy. And he had already left IBM. But he had just left IBM, looking for a job.

Name was Bill Worley. He became our chief architect. He's probably, more than any single person, the reason that in the end the RISC machines did come out, and were successful. Did, in some sense, along with a couple of other companies, change the way most computers are built today, including all PCs.

And I was this very strange guy coming to HP. Because IBM, in those days, was a very, very paternalistic company. There was a dress code-- white shirts, blue suits and ties.

The computer repair guys wore suits. And they carried leather attache cases instead of tool boxes. Salesman, there was no drinking on company premises.

If you went out for a computer lunch as a salesman, for a lunch, and you had a Martini, you were supposed to take the rest of the day off. The company took an immense interest in its employees, all of whom it thought it had hired for lifetime employment, and many of whom stayed for their lifetime. IBM jokingly stood for I've Been Moved. But it wasn't a joke. Most people had gone all over the world working in different jobs, always moving up the ladder.

Good pay, good working conditions, wonderful benefits, and so forth. And people thought of it as the most uptight, unpleasant place to work that you could possibly imagine. I never worked in IBM, except in the research division. IBM was smart enough to know that guys that go to research divisions, most of whom have PhDs-- we had something like 1800 PhDs there when I left, in 1980, and lots of guys that didn't have the degree but had equivalent experience-- a PhD wasn't a requirement. It just turned out that because of the nature what they did, most people had done that.

And it was nothing like the rest of the company. Nobody wore ties. There were guys that locally were called wild ducks, who were at that thin line between genius and insanity, guys that had won Fields prizes in mathematics, but you wouldn't trust them to buy a movie ticket, they were so weird. And all sorts of unusual things going on.

Dress was sloppy. Today it would be just like the way everyone dresses. At the time, even most people who went to GE labs, everybody was in a white shirt and tie. Not at IBM.

And it just was the loosest, freest place. And since there was so much money, and so much resource, you could basically pretty much, at the discretion of the local manager, work on anything. So the disadvantage of that was the company didn't care a lot about what you were doing. So it was hard to transfer technology, or transition it into a product.

But on the other hand, if you thought it was pretty interesting to work on cosmological relativity-- very famous simulation of a spiral nebula that appeared in the Cosmos series, that was done in my group. I had a very smart mathematician. Do you mind if I work on this on weekends? I need infinite computer time. So we did it.

Why? Who cared? Smart guy, an interesting problem.

Did it. Published it in Science magazine. It had nothing to do with IBM.

So that was the climate at IBM that I was used to. And unlike what people thought, I was a bit of a maverick in that I loved-- because I felt bad about becoming a manager. Because I didn't feel like I was involved enough. So I used to walk around all the time. I was always walking around.

And when I would go to an IBM management school, they would say, that's the worst possible thing you can do. We call that micro-management. You cannot undermine an existing manager, because you're his manager. You can't go in and talk to his people. Do this.

And they had this whole set of IBM rules as to how you were supposed to interact through the management chain. And they were probably on average good rules. But they weren't good rules for the research division. And they sure weren't good rules for me.

So I had this reputation for being this guy that was going to show up any time, say anything, didn't want to have slides. In those days, we had the overhead foils, mostly. Just tell me what you're doing.

So it was always criticized. It was criticized by the personnel people. Ralph Gomory liked it. But he knew he wasn't allowed to like it. So he would say things to me from time to time.

And I used to play tennis every day at lunch. We had tennis courts there, in the season when you could do it. A lot of people would come out and watch the matches.

We had four courts, or whatever it was. I was a pretty good tennis player. And people thought it was really strange.

Here's this guy. He's managing 500 guys. He's one of the four senior managers. He's out there cursing like crazy, double faulting on the court. What's going on here?

And I just didn't care about any of that stuff. I just figured, screw it. At one point I guess I thought I was successful enough. They invited me to be the head of Poughkeepsie, which is a big, big deal. It was the main thing. Run Poughkeepsie.

No qualifications whatsoever. But IBM didn't believe in qualifications. They believed in general management.

Once you get to a certain level of management, you can manage anything. You just have to know who are the right people to choose to advise you. And then listen to them.

And of course, there's business things. They said, we'll send you to business school. Get an MBA at night, or something. They said, don't worry about it. You can do it. It was like a 10,000 people, 20,000 people.

I said no. I said, no, I don't want to do that. I hate the idea of doing that. I love what I'm doing here.

So then they said, we're starting a new division in Tucson. Would you like to do that? No, I don't want to do that. I want to be in research.

So I was strange. So I came to HP. I didn't know what to expect. I heard a million stories about Silicon Valley. Of course I'd been out here, I don't know how many times for IBM, and visiting various universities, a couple of advisory boards, and so on.

And I walked into HP labs. And to my amazement, most of the people were wearing ties. And they were sitting in their little offices. And I'm-- it was more like what people thought IBM was like.

I couldn't find any wild ducks. Everyone was smart. But I didn't see any of these crazy characters running around, none of them, not one. And I said, this is the wrong mix of people.

You got guys who are development guys doing research. And they're doing research like it's advanced development. No wonder nothing comes out of this computer part. It's crap.

It's either so far in the future. So I went, and I said, I'm going to change all of this. Is that OK? And they said, be my guest.

And I said, do you guys care if I get really involved, and walk around. And then John Doyle tells me, we call that management by walking around here. You'd think we invented it. I thought I'd died and gone to heaven. They were telling me that my job was to do what I liked to do.

So I had no adjustment necessary whatsoever in management style. In company style, everything was completely different. But I guess I was adaptable enough.

For example, in IBM, everything got written down. If three people went to a meeting in which there was some discussion, not total agreement, each of the three would produce a memorandum to file, summarizing his view of the meeting, and what would be done. And it would go on and on. You'd have a task force to evaluate a task force to evaluate. There was always pushing the bump under the rug further away.

It took a long time to make decisions. The decisions always escalated up. They'd often be made by high level people who had no competence to make the decision at all. I came to HP.

I couldn't figure out who was making the decisions. But they seemed to be made in a timely way. And in general, they seemed to be pretty good.

But I didn't know who was doing it. Because there was no obvious mechanism. Nobody ever wrote anything down. They just sort of shook hands. And people seemed to do what they wanted.

So I loved it. I thought I had died and gone to heaven. I thought it was so fabulous.

And people from the press for a while kept interviewing me. How have you made this immense cultural change of coming from stiff, buttoned-down, blue suit, alligator suitcase IBM to crazy Silicon Valley? And I would say, well, there are some differences. But everybody's been very nice.

In fact, it was the easiest thing in the world. I just had to learn where the buttons were that you had to push. And that took me a while. But eventually I learned that.

The other thing that really was different, wildly different, IBM regarded-- I once heard a speaker say that IBM had the air conditioner theory of research. Have you ever heard that?

House: No.

Birnbaum: That means that if you build a building, and it's a certain size, and it's got a certain number of floors, you put in air conditioning. Even if it's in Alaska, you put in air conditioning. That's-- if you're a corporation of a certain size, with a certain revenue, you have a research division. Whether you ever count on them to do anything, or whether you ever expect anything to come out of it, who cares. You just have one.

So research was considered like this plaything of the corporate guys in Armonk. The real blood and guts IBM guys, that were in places like Binghamton, and Endicott, and Poughkeepsie, and all the other places

that cranked out the billions and billions of dollars per year worth of real machines and services, that ran the factories, and ran the VLSI lines, and all that, research was, with few exceptions, 1,000 miles away from them. Every once in a while, something would come out of research that would justify its existence-- a new process, or a new type of transistor, or something.

And people would adopt it. Because they weren't stupid. If they saw something was really good, they would use it.

But in general, there was no joint projects. There was no camaraderie. And there was no expectation, zero.

You were like a joke. You were the well-educated prodigal son that nobody really cared about. Stick them up in their nice little [INAUDIBLE].

I came to HP, and it was like everybody-- what is HP Labs thinking? There had been such an immense history of instrumentation under Barney Oliver being developed and transferred into to successful products, or the labs coming in and using mathematics to solve a problem that had t-- it was like a love-in, a love-in.

And there was no difference in salaries or anything. You could go from the divisions to the labs, and from the labs to the divisions. They even had a sabbatical program. I couldn't believe it. It was so wonderful.

And so people were expecting, right from the beginning, that if we were going to start a computer research center, we were going to really make a difference. And what were we going to work on? So suddenly I go to a meeting-- any meeting, didn't matter what it was-- and if somebody happened to mention IBM in any context, everyone in the meeting would turn around and look at me, like there he is.

Or if I said something, every head would turn around. It was like E. F. Hutton. It was a really unusual situation for me.

But I made believe like this was business as usual, that's OK. But it was hard. Because I was by myself. And I didn't have anybody else there, and nobody to tell me what to work on, and what not to work on.

But the key reason that they wanted me to be there was to try to figure out a way to unite the computer lines, and give them a cost-performance advantage. That's what they said. So we did figure that out, how to do it. It was a high-risk project, no pun.

Most people thought we were crazy. The press thought we were crazy. I was very limited in what I could talk about.

I couldn't talk about IBM. I could only talk about the principles. And in fact, I couldn't tell any of the people, because of what Katzenbach had told me, how to write the compilers, how to do the emulators, how to write the operating system, which instructions, and so forth.

And to my great pleasure, properly asked questions that posed the problem to them. Smart people went away and thought about it, hired other people that would know more than they did. And the design which eventually came out was immensely superior in every way to the IBM design.

And eventually it did run COBOL. It did do real time. It did unify the product lines. We did migrate the customers. And I don't know how much HP makes today out of that product line, but probably $20 or $30 billion a year, I would guess, out of what it's become.

And for a while, it was the fastest computer. And it did have the best cost-performance. Subsequent management at HP managed to make the field a little flatter than it needed to be. But that's a different story.

So I think that the cultural change was totally in HP's advantage, with one or two really bad things about HP. And I like to think that at least I set in motion the change. The IBM research guys were really research guys.

They weren't retreaded divisional engineers. In many cases people who either were, or had been, or later became distinguished college professors. They were really thoughtful scientists in whatever field they happened to be in.

And they read the literature. And they attended the world's conferences. And they were in connection, physical contact, with all the people in the world that didn't work for a competitor, and in many cases even with the ones who worked for a competitor, through organizations and so forth.

They were well read, well schooled. You could, even before the days of internet communication, boy, the day one of the journals came out, if there was something in there, you'd hear it at all the tables at lunch. People would be talking about it.

Came to HP. Nobody even had subscriptions to the journals. They didn't even know what a journal was.

Most of them had never been to a conference. Or if they'd been to a conference, it was the specific one conference a year that was-- no one had told them not to. It's just there was such an immense pressure to do something relevant, and to get it to make a real difference in the real world, to transfer, including transfer the people that people didn't take the time to do that. So the result of which was, I found in the HP culture, people were reinventing the wheel all the time. Not because they were stupid, but just because they didn't know what else other people were doing.

So I changed that a lot. I made people join societies. I made people subscribe. We started having lots-- we started at-- they didn't even have a good seminar series, where somebody would come in once a week. And of course as I hired people from other companies and some university people that came from a tradition, it took off.

And so I think by the time I retired in 1999 there would be almost no difference between the HP culture and the IBM culture that I have known in those regards. I think that became-- first of all it was fun. It's fun to see what other people are doing, and go to a conference. We changed the travel budget, so that people could attend a couple or three conferences a year. We encouraged people to join professional societies.

I couldn't believe it. I inherited 70 people in the so-called computer research lab. And I said, how many of you are members of the Association for Computing Machinery?

And I realize I'm talking to a former president. I don't remember the answer, Chuck. But I think it was 20.

House: Yeah, wouldn't have been very high.

Birnbaum: 15? And they didn't do anything, except get the journal. I mean, they weren't on committees. Standards committees, nobody. There were people in the company that worked on standards committees.

The Instrument divisions had learned a long time ago how important that was. But not the computer guys. No way. They had nothing to do with standards. Standards was the name of the game, especially if you believed in an infrastructure that was going to become a utility.

House: Oh yeah.

Birnbaum: That's the standards game. You have to do that. So I say, comparing the two companies, I think HP had better engineers. I think that when you finally figured out what you wanted to do, and you really wanted to build it, these were the cream of what I used to call super implementers

You would tell them what you wanted. And they might not build what you wanted. They would build something better.

They'd figure out a different way to do it. They would really-- just everywhere I went, I'd run into people like that. And small numbers of people would accomplish astonishing things in computing and in instruments, and in physics, like the atomic clock project, and so on.

House: Yeah, just to pick one.

Birnbaum: Three or four guys change the whole way the world tells time. It's something to be really proud of. And unfortunately, while they had the great engineers, they didn't have deep thoughtful scientists that really understood the whole thing, with a few exceptions. In computing, I'm talking about.

And I hired a bunch of those. And in order to do that, I had to change the HP culture. Because HP believed that everybody was equal-- nice democratic theory. And that if you hired somebody that wasn't just out of school, which was a favorite way of hiring, so you could do on the job training, you made believe that he was just out of school.

So for example, when they made me my first offer, they offered me two weeks of vacation.

House: Is that right?

Birnbaum: I had five weeks at IBM. And I said, are you making believe that I haven't worked for 50 years? He said, yeah, absolutely.

You have to start all over again. I said, screw it. I'm not coming. What are you talking about?

And they wouldn't let me make offers to advanced people at higher salaries. And they wouldn't let me do housing packages. So I had to fight for all of that.

But in every case, I found that they were quite reasonable. And in the end they gave me some packages. And I hired people whose names are well known to this museum-- guys like Bob Rao and Josh Fisher, and Worley himself, and Frank Carrubba.

So these are guys that were world class guys. And you couldn't expect them, at the age of 50, to come in as if they had just graduated from school. Yet I needed their expertise. Because the company did not have the time for on the job training.

It wasn't that the kids out of school might not have gotten to be as good as these guys. But it might have taken them 10 years. And we needed to do it in 10 months.

So we built a team. And it was the most fun I've ever had. And I was--

House: Was it really? Most fun?

Birnbaum: Yeah. It was even better than Yorktown. Because I existed in this funny world where I couldn't say anything, but I could ask questions. And it was so much fun to see these guys, and ask them a question they hadn't thought of. And almost instantly, I mean, like the next day or the next two days, they'd be banging on my partition there, saying, what do you think of this? What do you think of this?

And some of those turned out to be really fantastic ideas. I'd had that experience at IBM a lot. But I didn't know that I would ever have it at HP. And I don't think I would have, with the original set of rules I inhabited. A quick example will be helpful.

House: OK.

Birnbaum: And we'll move on. At IBM it was called to my attention that there was a mathematical 19-year-old genius living in Buenos Aires, Argentina, who had never been to college, but who was working as the nighttime operator at one of IBM's installations in a bank. And that they really recommended that this guy was so brilliant that we ought to take a look at him.

So I called up some guy in Argentina. And I said, tell me about the guy. And I had one of our top mathematics guys talk to him on the phone. And he came in. And he was really shaken. And he said, just get this guy here.

House: Really?

Birnbaum: Get him to come for an interview. So I sent the guy a ticket to come from Buenos Aires to Yorktown. He was, I think 20 by that time. I think he had one year of college, if he had that. Name was Gregory Chaitin. He's very, very famous now.

House: OK.

Birnbaum: And he just had kind of taught himself mathematics, like Ramanujan, or one of these guys. He was just unreal, how smart he was. So we hired him on the spot.

And he went into the math department, not my department, even though I had-- he went in the math department. And he started working with a lot of the famous mathematicians that were there.

And even to them, it was astonishing. Because things that they thought of as ABC run, Spot, run were unknown to him. Because he had never taken a course and studied them.

OK, so I was very interested in distributed systems. And I started thinking about-- I had gotten promoted to be the director of the computer science thing, somewhere along 1975. But even before that, I was in charge of all the systems stuff.

And I decided-- I had to write a document ever year that said, make a 10 year plan for where you think the computer industry is going, and how your research projects are supporting that. This is the IBM annual plan. But it was updated every (wrote??). But it always had a 10 year-- it had a three year horizon, a five year horizon, and a 10 year horizon.

And IBM was rich enough, and I think smart enough to think that 10 year horizons were important. And that you should do projects that you didn't expect to see anything from for quite a while. Maybe you didn't even know exactly where they were going to go to.

So I started thinking a lot about it. And I guess somehow it occurred to me this there were three ideas that were kind of connected together here. One was that computing was going to be distributed and network based. I thought that Ken Olsen and DEC and all that, exactly the right way to go, the VAX, perfect. Someday those would be standardized, and they would turn into some kind of an infrastructure that not just one manufacturer could plug into, but that everybody could.

Didn't know how that would happen. Didn't know what it would look like. But I imagined that there would be an infrastructure.

I was totally convinced, based on 20 very positive experiences, that if you built the kind of user interface that wasn't easy to use but was intuitive in terms of the problem you were trying to solve, and your domain and knowledge, so you had to learn something, but you learned it in terms of what you already knew or could easily learn, rather than having to learn what shift register was doing what, or something, that that was gold. And then the third one was the way to get the computer performance for that was to distribute the computers, but not to distributing them locally, the way we were doing in that first project, but to get the network bandwidth up to the point that we could have long distance connections. So that you could bring specialized compute power to bear on special tasks that you did only once in a while.

And when I wrote-- in other words, let's say you were doing--

House: You haven't mentioned that before now. That's a big thing.

Birnbaum: That's where this all started.

House: OK.

Birnbaum: And so I started thinking about distributed computing, which was not just for a work group, but which was for a distributed work group, or for a whole company. And I don't know who first used the word in my presence. But somebody said, my god, it would be like a utility.

And I think I said, utility, what does utility mean? And I started thinking about the electrical utility, where there was a standard for the plug. But you, as the user, didn't have to know what was behind plug.

You didn't have to know whether the electricity was hydrodynamically generated, or nuclear, or what. You just needed to know that it was the right voltage and the right current. And if the world had only been smart enough to agree on one, you wouldn't have even had to know that.

And similarly, the guys who provided the electricity couldn't have cared less about what you plugged in. So there was an interface in which you had complete independence on both sides. And I said, well why shouldn't there be a data utility? Why shouldn't you be able to provide data from the infrastructure, just like we provide electric, so what comes out of the wall is a bitstream instead of an electron stream, and what plugs into it are going to be all these appliance things, which are particularized to what they do, based on the need for the user interface, and the need for specialized performance?

And the way to accomplish that is to distribute the computer power. So the thing that is the appliance has only enough computer power to do things that have to be done locally, like maybe run its display, or turn on a microphone, or actuate a sensor, or do something. Whereas all the rest of it is behind the wall.

And sometimes it's in the room next door. Or maybe it's physically in your room. But sometimes it may be on a different continent.

The problem is we don't have good enough bandwidth. So that led to starting all the fiber optic work at IBM, started all the transmission. So I started preaching this story-- appliances and a global information infrastructure that would behave like a computer utility, where the key act of the utility is that you pay by usage, and that you substitute a subscription for capital investment. Well, it was as if I had let a snake loose in the room when I gave this presentation to the IBM executives.

House: I bet.

Birnbaum: What? You're going to rent computer time instead of us selling these things with a 25x markup on memory, or whatever it was they were selling? They were making, who knows, 20% net profit or something in those days, minting money. So they said, this is the worst thing we ever heard. This is just awful.

And the time frame for this, I can tell you almost exactly. It first started around 1971, '72. I started thinking this way. Because I was in charge of systems, I was the systems monitor for the annual IBM corporate strategies research part. I wasn't yet the head of research.

They hated it so much several of them kind of said, it would be kind of good if you didn't talk to any customers about this kind of stuff. And we had entertained customers at research all the time. And that's what we were working on.

They didn't tell me not to work on it. They just told me not to talk about it, secret project. And it really rankled me a lot.

And then a funny thing happened. And I got called into corporate. And my boss, Ralph Gomory-- no, he was my boss's boss.

We've been approached with something that's totally secret. You can't talk about it. There will be no publications.

It'll probably take six months of your life. But we're starting a really high level task force. It's going to report directly into a guy named Gil Jones, who at the time was the Vice Chairman of the board of IBM. A guy named Learson, who was the head of it.

And he said you will report to him during this time. You can not talk to anybody about this-- not your wife, not anybody. It's really secret stuff.

And we'd like you to head it. Because we think that a lot of your background-- we're going to give you 15 slots to hire 15 people with skill sets that we may not have in IBM. And you have an unlimited budget. I remember that very clearly. I said, what do you mean, an unlimited budget? He said, you could spend whatever you need to.

So what is going on? Well, it turned out that IBM had been approached by Marcus Wallenberg, who was at the time the head of Saab, or Volvo, or one of those car companies. I think it was Saab. And he was the head of the Ericsson phone company.

House: I taught in Wallenberg Hall at Stanford for five years.

Birnbaum: AT&T-- it was named after his son, Raoul, who hadn't yet died at that point. AT&T had come out with the world's first large scale switching digital tandem exchange, so-called ESS-1. It was a 44-bit computer that used time domain multiplexing to switch its information.

It had its own language, its own memory technology. It used a thing called twisters. And it was labeled as the future of telephony.

And Ericsson, which mostly made PBXs at the time, but made also big exchanges for Europe, was one of the world's top telephone companies in Europe, maybe in the world, South America. felt that if they didn't get into this field, AT&T would consume the whole industry. And that they needed to have a partner, because they had no digital expertise. So they said to IBM, how about if IBM and Ericsson secretly form a company that will be a joint venture to design an ESS-1 type switch, and eventually a whole family of digital PBXs, and so forth, and so on?

So there were enormous antitrust implications. It wasn't clear whether it would help or hurt IBM's antitrust situation. We had huge gaps in our knowledge, but so did they.

How would you share the profits? Who would do which? How would it be marketing? What would be its label?

So a whole giant sales force was put together with very high level people from each of the relevant departments, all secretly. And I was put in charge of the technology. And so we flew to Europe. And we had secret meetings with people who were from Ericsson.

I brought along, of course, John Cocke, my smartest guy, and half a dozen others, including a guy named Maurice Karnaugh, who was from Karnaugh Maps, who had worked at Bell Labs, and had built one of the first folding switches for ESS. We had come out with a little PBX in France. So I had that guy, that designer came along. So I had like five or six telephone switching savvy guys along with-- and we listened to their proposal.

And the whole time we were sitting there, my friend John Cocke was going no, no, no, wrong, wrong, no, no, wrong. And they saw him doing this. And they said, well, what's wrong?

And he got up, and in his crazy way tried to explain. And he confused everybody, and so on. Anyway, the upshot of the meeting was, instead of trying to work together, which looked impossible, why didn't we each spend three months, and each of us design our own system?

And we'll see. We'll come together. And we'll compare the designs. And we'll see how much commonality. Meanwhile the business, the marketing, the legal, et cetera teams would work on market dynamics, and was this a big business. But obviously it was intended to be a big business-- a big role of IBM if it got finished. And it would be a major challenger to AT&T.

So I headed that team. I hired some really, really wonderful, wonderful people, wonderful people from the telephone industry. And we had a lot of meetings. And we spent the first three weeks studying the AT&T ESS design, which we thought, eventually, once we understood it, was the stupidest design as far as the computing and memory part was concerned. It was just a design done by amateurs.

No reason to have a special purpose computer. The computer language they had was ludicrous in terms of optimization. The pathways were terrible.

The performance was terrible. The memory was stupid. It was expensive, and needlessly complex.

And we said, I think we don't have to do any of that kind of stuff. This doesn't need a thing. And John said, I want to design the computer.

And then he winked at me. And he said, RCC.

And he went away, and he designed that whole thing in two days. He came back with an instruction set. I then took eight or 10 of my best computer guys, put them together with John, never telling them what it was for.

They just were working on a computer, John's idea. I got George Radin, who was an IBM fellow from P01???. Marty Hopkins was an IBM fellow -- I had a super, super optimizing compiler team. Could we build it?

So the net upshot was we designed the whole exchange. We designed the switch. We designed the memory. We designed everything.

And John designed a RISC computer-- we didn't call it that-- to sit in the middle of the telephone exchange. And how you measure the capacity of these exchanges is a unit called an erlang, which is a measure of information entropy. And we took the ESS-1 as the standard, that we had to at least do that.

And John's design came in at seven times the ESS-1. So it was really awesome. And furthermore, because it was a RISC machine-- meaning that the microcode was the instruction set-- you could build the same machine for all sizes.

You could build a big tandem exchange. You could build a regional concentrator. You could build aconcentrator for a telephone pole.

You could build a PBX. It all would run the same code. Unthinkable.

So we put the design together. And we travelled to London for a secret meetings in Claridge's, under false names, coming in the back door, as if we were one of the royalty that was going there for a tryst with the whole Ericsson team, and our team, the lead guys from them. And then it came time for me to give a one hour presentation, and for a guy named Bjorn Lundvall, who later became the president of Ericsson. But at the time he was the CTO -- to give their presentation.

And IBM was to go first. So I presented this stuff in an hour. I rehearsed 100 times. Because in front of me was the IBM Chairman of the Board, and all these top guys. And I finished.

And here was the exchange. Here was an early simulation of it. Here was the performance numbers.

Here was the computer. Here was the language. Here was the memory. Here's how we did the line concentrators. Here's how we use digital gyrators instead of using [INAUDIBLE], and so on.

And Lindvall got up, and he said, there's no reason for me to present. He said ours is just like ESS. It's primitive. This is clearly the way you build an exchange.

And then he said I would just be wasting our time. Let's do it. So a lot of applause. Everything was great.

At a meeting two months or three months later, we hired Morgan Stanley to be the investment banker to form a new corporation. Greenwald was the guy at that time, young guy, was doing it. And got together.

They ironed out all the business things, and who got which, and so forth. Because it turned out that the digital part of the system produced a minor part of the revenue. Most of the money was in all the outlying stuff. And we had to figure out how to do all that. So they figured all that out.

And what would be the first size machine? How we would do it? How long it would take, and so forth?

And IBM was going to start a new place. They bought some land to do it. I was going to be the head of it-- not of the organization, but of the development team. And the Ericsson guys would come to the United States to do the part that we needed to do together. And they would work on the part that they could by themselves.

And we had this incredible dinner. They served-- I remember to this day, the only time I've ever had it of my life-- Chateau Petrus. We were drinking like crazy. Everyone shook hands.

And in the morning, we were coming together for the actual documents signing. And in the morning we all showed up. It was about 12 or 14 of us from IBM, of which I was by far the most junior person there. And Wallenberg came in. He said, I didn't sleep all night. I can't do it.

He said, I can't do it. I'm so sorry. I've wasted all your time.

But if we do this, there will be no more Ericsson. There will be no more anything. You'll just swallow up Sweden.

And you'll swallow up my company. And, my goodness, what you did in three months, I have 1,000 years of experience in my research team. And in three months you leapfrogged them like crazy.

And he said, I'm sorry. I just can't do it. The deal's off.

What? So we went back to Yorktown, hardly believing it. And nobody knew anything.

We weren't allowed to publish anything. We never could talk to the papers about it. I'm not even allowed to be talking it about now. It's never appeared in print, as far as I know.

House: I've never heard this story. It sounds like--

Birnbaum: Now the question was, what should we do? So by now, it was 1975. The six-month task force had dragged on for three and a half years, the first year of it with them, and the next two years designing a digital exchange as if it was a standalone IBM business. And so then I made the presentation to the IBM board of directors. And by now Frank Cary was the CEO.

And they asked very good questions, like, what's the development expense for this likely to be? And of course I had many experts that had looked at that. Manufacturers who said, I don't know, somewhere between $100 and $200 million. Peanuts for IBM, but still $200 million.

What's the time to break even? Well, making reasonable market projections, and allowing time for building up a sales force that we don't have, and all the rest of that, probably three to five years. So in other words, this thing might break even in seven years, is that right?

Yes. What's the size of the business after that? Well, $100 billion a year. Blah, blah, blah, blah, blah.

What's the level of risk? And they talked. And they talked.

And then either Opel or Cary or one of them said, seven years? I'm going to be retired by then, aren't you, Frank? Or something like that.

And they said, nice work, sonny. That's one of the best projects we've ever heard about. By the way, we'll take care of you and your team. You've lived for four years on this, and so on.

It's really a great stuff. But we just can't do this. And instead, they went out a couple years later, and they bought ROLM Corporation.

House: Right.

Birnbaum: Instead of doing it themselves. Terrible design, I mean, compared to what we were talking about. So meanwhile, back at the ranch, the question was, what do we do with all this great work we did?

We'd invented a process. We only needed single processors. We got Sam Winograd and Jim (Gray???) [INAUDIBLE] to invent a digital Fourier transform as part of this project for a single processor-- Abe Peled did the basic parallel stuff that turned into RP3??? in Watson later on.

We had a whole new class of global optimizers. We had new computer language. We called it PL.8, because it was about 80% of PL1.

But I don't need the simple instructions. So you felt like you were programming in a high level language. But you really weren't. You were programing to the built in hard-wired single cycle machines.

What should we do? So we had another task force, just of research guys. And after a month, we all decide, OK, we're going to build that machine.

And that's the machine that got to be called the 801, which was the first actual physical RISC machine. And it took another four years, three and a half years to get that to work. Frank Carrubba, whom you know, was the engineering manager of that machine. Many, many-- Fran Allen, and many of the others worked on that machine. We got guys out of the old stretch computer team to deal with the operating system, and so on.

The target was the IBM series one minicomputer, which was a disaster, and an embarrassment. And the question was, how much better did you have to be than the series one in order for IBM to do this? I'd been burned so many times by IBM executives bailing out because they didn't need to be so aggressive-- the company was so successful without being so aggressive.

I said, we're going to do two projects. We're going to make a series one minicomputer lookalike, except it'll be 32 bits, extendable later to 64. But we have got to be not twice as fast, which was what they told us. We've got to be 5 times as fast for the same cost. So cost performance 5x, one fifth.

I said, but I don't think that'll make it anyway. I think that's going to die in politics. So we're going to do another one.

Is that we are going to replace the microcode of the biggest 360-- well, it was 370 by then-- with a RISC machine. Nobody will know there's a RISC machine. There will just be this thing under the cover, [INAUDIBLE] thing.

That'll work 10 times as fast as the microcode used to work. In other words, use the same software. But instead of compiling it down into the microcode, we'll put one of those.

So it ran 11 times faster than the series one on the NASTRAN floating point stuff. No surprise. We had all kinds of incredible floating point stuff in there as hardwired instructions, single cycle. It was like beating up old women. It was just ridiculous.

And it outperformed the-- we never built the one for the 370. We just simulated it. But it was a really good simulator.

And it was just way, way, way faster. And of course it meant that the 360s could scale. And so the amount of money that you saved on field support by having to train everybody on only one machine paid for the whole project in one year.

They said no to both. OK? No.

House: Now who is they? The executive--

Birnbaum: IBM corporation. Research proposed it. We couldn't do it ourself. We needed manufacturing.

We'd done the research. Now was time to turn it into a development, manufacturing. So specifically, a guy named Chuck Branscomb, who ran the-- not Lou Branscomb, Chuck Branscomb, who ran the series one--

House: Related?

Birnbaum: No.

House: OK.

Birnbaum: Smart guy. Just said, gee, we're doing great with the series one. We're making a lot of money.

House: Thanks.

Birnbaum: Five years, seven years, forget it. And the 360 guys said, what the heck would we do that for? No one's complaining about the performance. We can't make them fast enough to sell them.

What the hell would we do that for? Not a bad argument. But it's not good if you're a research guy.

So I came to feel like I was a demagogue. I had gotten all these people to do all this stuff. And they had succeeded against outrageously hard odds.

And I wasn't having any fun anymore. I had no projects of my own anymore. This was a full time job, managing all of this stuff.

And I'd gotten promoted. I'd gotten a raise, and all this kind of stuff. And just around that time was when I got the call from Hewlett Packard.

And this was one of five similar things. Another one was speech recognition. I had hired, because of my friendship with Raj Reddy, two of his best students from Carnegie Mellon-- Jim and Janet Baker.

They brought their PhD thesis, which was called "Dragon," to Yorktown. And we built that system. And IBM said, no, we're not going to do that.

So they went out, and formed Dragon Systems, which was very painful for me to observe. And we had the signature verification project, which was fantastic, better than anything that exists today. They decided not to do that for a different set-- they always had a good set of business reasons as to why these things were too high risk. But if you're a research guy, especially research manager--

House: You're dead.

Birnbaum: Even though, gee, it was a wonderful place to work. The people were smart. The environment was good.

They treated us really, really well, and with great respect. They just would say, here's a stock option. Or here's a this, or here's a that. No.

So I was really ready to do something. In addition to which, the distributed system stuff had started. The internet wasn't called the internet. It was called the ARPANET.

I wanted to have an IMP at Yorktown. And they didn't want to put one in IBM. So I couldn't get one.

House: Guys there wouldn't put one in?

Birnbaum: No. I don't know why. But they wouldn't do it. IBM had a separate part of a company called Federal Systems that did all the interaction with that. So we didn't have that.

So we weren't a part of that. And here I saw my dream was actually starting to come true. Not in its initial form.

House: Sure.

Birnbaum: But we just had nothing to do with it. Meanwhile, Xerox PARC was changing the whole world with all the things that they were doing. And I felt, my god, I'm working in this company that thinks the future is time-share terminals, and over-pricing. Not that the research wasn't wonderful, and not that they didn't have super smart guys. But the company was run by very, very skilled and clever business men who had created an enormous business that was immensely profitable, but that didn't look 10 years out.

The notion, that disruptive technology thing, had not been written. But to a lot of us in Yorktown, we spent all of our lunches talking about the train wreck that was going to happen. And the argument was, is it five years from now? Is it 10 years from now? Is it 15?

There wasn't an argument as to whether it was going to happen. The question was, can IBM survive when that happens? And the answer we came to was no. And as you remember, the very nearly didn't.

House: That's right.

Birnbaum: So anyway, all of that was kind of what led to my leaving for HP.

House: Well, that led to you being interested in looking, or at least to being open to listening.

Birnbaum: I really wasn't interested, in spite of all that. I thought I would go to a startup. I had been invited by Ed de Castro to join Data General when it started. I had turned that down.

I had had feelers from DEC at one point or another. There were quite a lot of startups, just because of the title I had and job I had that it-- but I was-- my personal situation wasn't so stable at the time.

I was going through a divorce, and had little kids. And I needed a salary. I couldn't afford to take a--

So I never went with one of those. I wasn't really looking. I was thinking about it.

What I had decided to do, after I got divorced was research, and to just go out to try to improve my financial lot in life. And I had just about accepted a job with McKinsey to become one of their consulting guys.

And as it happened, the phone call came from Hewlett Packard. And I liked that better than working for McKinsey. So I came out here, even though a lot of people thought it was crazy.

Because I found out, once I resigned-- I never negotiated with IBM. I just came in and resigned one day that I had, not to my knowledge, been promoted to be the head of all IBM research worldwide, which was 3,000 people, and not just computing, but all the other stuff as well. And Ralph Gomory, my boss of the last eight or 10 years had put me in for that job. And he was being promoted into a very high level corporate position, which he eventually took.

And so it seemed crazy to leave that, to come to a place that had 50 people working on computers that most people didn't even think was a computer company at the time. But I did anyway.

So it turned out when we were doing the early RISC machine, one of the questions was-- it turns out in RISC computing that high speed registers are very important. And at the same time, they're built out of the same kind of logic, the same speed logic as processor, not the memory. So they're very expensive. So the trick is to have the fewest number of registers you can, to keep the cost of the machine down, but enough registers that you don't spend all your time swapping information in and out, in and out.

So they had a variety of ways of doing it. One was the way that Sun eventually ended up doing it, by having sets of register windows, and just switching the whole set with one instruction. It's very expensive to do. And it's kind of slow, and so on.

And here was this guy, Greg Chaitin. And he always would come by my office to thank me for hiring him, even though he wasn't in my department. And say, you know, if you ever have any mathematics oriented problems, or something that has to do with languages and compilers, I really have so much time.

I'm a bachelor. I love to work at night. He said, I love to have a computer science project in addition to what I'm doing in the mathematics department.

So he came by. And I just, I don't know why, but I said to him, Greg, here's this problem with the registers. Is there some way that you could think of that we could have some large number of programs, like seven or eight or 10 multiprocessing in the machine, and only have 32 registers, and not pay a really heavy price for it?

The question is really, which registers do you keep? And which registers do you get rid of? And how many do you need? And how can you characterize it? And how do you do all this in one cycle kind of stuff, without paying this tremendous price of-- running up the cost of the machine on the registers?

He said, oh my god, is that an interesting problem. Oh my god. He said, do you mind if I kind of interact with your computer team, and I find out all the stuff that's involved? I said, no.

I didn't hear anything for about a month. And then the compiler guys broke into my office, with him sitting here, kind of looking down. And they said, oh my god. You won't believe what Greg has done.

So what Greg did was somehow or other he had the insight-- apparently he had the first day, although it took him a long time to show it-- that this was completely equivalent to the four colour map problem. You know, the question is how many colors do you need? So no two registers adjacent, if they're not from the same program, and so on.

How can you color the registers with four colors, so that you have the minimum blah, blah, blah. Anyway, today's register allocation schemes are done by coloring algorithms, and that's all Greg Chaitin. He later went on to do all sorts of things. He won many prizes. And he's considered one of the world's great mathematicians today.

He had sort of done that. And so the lesson I learned from him and maybe four or five others was that if you got some guy, and he was just off the charts smart, don't be embarrassed to ask him the question that you think he doesn't have any knowledge of. It doesn't matter. He's going to come at it from some angle that you're not smart enough yourself to understand. And you probably would not understand it even if he told you.

So, years later at HP, in the RISC story, we solved a lot of problems that had to do with architecture, languages, compilers, IO, so on. But the problem that it didn't look like we could solve, and it looked like it would end the whole story was we didn't know how to migrate the existing customer set off the previous crappy machines without holding their hands for a year each, which we didn't have the personnel to do, to either reprogram them into the new architecture or to do some kind of an emulator for what they were doing. The problem with emulators is that they ran slow. And meanwhile, IBM, and DEC, and Amdahl, and Fujitsu, and Hitachi, and everybody else was all over our customers.

Because we were late to market for a couple years. Saying, why switch to Hewlett Packard, for god's sakes, and have to do reprogramming? As long as you're going to do reprogramming, here's our machine. Come to us.

We'll help you reprogram. We've got reprogramming centers. We'll get you up in six months, and so on.

You want to take a chance on a new architecture, a new company, and rewriting all your applications? You must be out of your mind. And little by little, all of our customers were dropping out. They were saying they were not going to wait for us.

So we needed-- and I figured, oh shit. I didn't think about that. And migration is going to be the death of me, and this project, and RISC technology. Even though we had achieved the performance goals, and the cost goals, and all the rest of that, if we couldn't move the existing customer set in some reasonable way onto the new machines, game over. Or at least--

Birnbaum: Everything we had worked on was going to be gone. And so I hired a whole lot of really smart guys to work on that problem. And they took all the standard approaches, a lot of them. A lot of companies had been worried about this for years, and years, and years.

IBM had put 1,000 people on that to get people onto the 360 from the 790, just to give you a feeling. But they never really had a good solution. The 360 ran 7090 emulators for 25 years, 1401 emulators for 25 years after that. Because it was so hard to convert the programs.

So we hired a bright young kid out of San Diego. He was a friend of a lawyer. He was the son of a next door neighbor of one of our lawyers, Ron Griffin, that was the lawyer's name, who said this kid is so smart. He does everything. He gets perfect grades.

He's just a brilliant, brilliant, brilliant kid. I wish you'd look at him. I think you ought to hire him in the labs.

I was very reluctant. He had a bachelor's degree. We were looking for people-- but he had no experience, never worked for a computer company. He graduated with the highest honors in three years, and had scored off the charts in everything. His professors said he was the smartest kid they'd ever seen in UC San Diego in that particular computer science curriculum at the time.

But we hired him. He made such a good impression on everybody. We hired him. He was a young kid. And I said, OK. Well, maybe this is the new Greg Chaitin.

Although he wasn't a mathematician, he was a super programmer, super engineer. Stick him on the migration problem. What the hell have I got to lose? We're going back to New York anyway if this doesn't work.

So I tell him about it. He didn't even know what I meant by migration. I had to explain what the word meant to him.

And he looked like he was about 14 years old. He had a very young look to him. He was actually 22, or something, 23.

And he said, OK, well let me think about that for a while. He thought about it for a while. And then he came into my office about, say a month later.

And he said, I think I have an approach that maybe could work, and I'd like to tell you about it. I said, what's that? And he said, oh I'm going to-- do you know what object code translation is?

And so the idea here is you take the object code, which is the code that is compiled down to the actual hardware instruction set of one machine, and you translate those instructions. An add instruction looks like this. We'll translate it into an add instruction over here.

And lots of people had done that. It's a standard technique. And I had done it myself. I'd actually worked on a project like that.

And I thought, oh shit. I had 10 guys at Yorktown working on this thing because of the 360 problem. Whatever you do, it comes out slow. No matter how smart you are, it comes out about 10 times slower than when you started.

But I wasn't allowed to say that. And I had learned not to say it anyway by this time. So I looked down, and I said, well, a lot of people have worked on that.

He says, yeah, I've read every paper I could find. He said, in fact, I read a lot of papers from your old group at Yorktown. I said, oh you did? I said, so you know about it. He says, yeah. He said, and I've done about three or four test cases on a variety of different languages.

And he said, here's what I found. He said, if you're really clever about it, you get about a factor of between eight and 10. And the code length takes up that much more memory, and the path lengths are that much longer.

And he said, so you can do it. But it's really slow. And that's why people like Intel and others put in hardware emulators instead of doing it, and so forth.

He said, if we do a hardware emulator, we can't build a RISC chip anymore. Because it won't fit. And any way, it won't be that good.

So I said, so what's your idea? He said, well, I don't know when the last time you looked at global optimizers. But they've really come a long, long way since the days you worked on them at Yorktown. There's been like 10 years worth of research on that.

He said, we're really good at global optimization now. So if we take the object code in the crude translation, and I write a global optimizer for object code, as opposed to what you usually write it for-- or sometimes you write it for that-- he said, I can take 10 down to roughly a factor of four. He said, the machine is somewhere between five and six or seven times faster than the HP 3000. So the net result will be that the guy's code will come over.

He won't have to do anything. It'll be very close to an automatic process. It'll run twice as fast as before. It won't run six times as fast. But don't you think twice as fast would be enough?

House: And it's all ported.

Birnbaum: And it's all ported. I said, you can really do that? He said, why don't you come over to the lab? I'll show you a couple.

So I figured he was going to show me like a little 10-word program. And instead he shows me this whole huge application suite from a bank. And he just goes, whsshh. And it goes. And it runs twice as fast as it did on the HP 3000.

So, make a long story short, we glommed onto that. We started migration centers all over the world. We got to the point where we could move a great majority of customers in one day.

House: In a day?

Birnbaum: In a day. We advertised a month. Quickly it was a week.

There were a few pathological cases where it could take a month. There was even one that took five months. But it never took a year. And it always ran faster. And for a lot of people, it just went across.

So I went to the guy that ran marketing at HP, Jim Arthur?. And I said, you should be advertising this. This is the greatest accomplishment any computer company has ever done, to do this.

Sam Prather was a manager of the migration centers. And a guy named Bruce Campbell worked under him. And that's what they did. They built these all over the world.

And customers would come in. We had a team of specialists that learned how to do this. We had all these tools. And usually-- always within a week-- they would leave with everything they had before running faster than before.

Sometimes it took the experts to go in and tinker a bit. But sometimes it happened, just be done in one day. And that's what saved HP.

And that's what saved our job. And the company decided it was not something they wanted to advertise. So we never told the world about it. And we were told not to publish any papers about it. Because it was like a secret weapon for us, that we knew how to do this.

So that guy's name is Dan Magenheimer. He later became head of all compilers in Cupertino. And he went on to do some other things in Colorado. And I think the last I heard he was a college professor somewhere, I think in Colorado. And he is the great hero of that whole story.

House: Oh yeah.

Birnbaum: And if I had just said to him, oh, Christ, that's a stupid old idea. I know about it. Don't even think about it, in the beginning, it never would have happened. And I don't think HP would be a computer company today.

And I probably had half a dozen experiences-- none as dramatic, or with as much a stake as that one-- but where the HP culture said, just because someone else has tried it and failed doesn't mean that we shouldn't see if we can't have a new look at it. And if we had more time, I could give you a lot of examples. But you get a lot of surprising results that way.

House: Let's hold that until next time.

Birnbaum: Yes.

House: But let's have a next time. Joel, this is just amazing. These stories are the epitomeof the kind of thing that the museum loves to capture.

Birnbaum: OK.

[INTERPOSING VOICES]

House: Thank you very much for today.

Birnbaum: Thank you. Thanks for making the time. I'm glad you're feeling better. I didn't know you were not feeling well. I'm sorry.

House: We're both still walking. Thanks.

END OF INTERVIEW

