AI: Expert Systems Pioneer Meeting
Session 3: AI Science and Technology, 1965 to 1985

Moderators:
Burt Grad
David C. Brock

Recorded May 14, 2018
Mountain View, CA

CHM Reference number: X8652.2019

Table of Contents

Participants:

Name	**Affiliation**
David Brock	Moderator, CHM, Director Center for Software History
Burton Grad	Moderator, SI SIG
Brad Allen:	Inference
Avron Barr:	Aldo
Denny Brown	Teknowledge
Ed Feigenbaum	Intelligenetics/Intellicorp; Teknowledge
Mark Fox	Carnegie Mellon
Peter Friedland	Intellicorp
Paul Harmon	Newsletter
Peter Hart	Syntelligence
Gary Hendrix	Symantec
Fritz Kunze	Franz, Inc
Doug Lenat	MCC/CyCorp
Brian McCune	Advanced Decision Systems
Peter Norvig	Google
Alain Rapaport	Neuron Data
Herb Schorr	IBM/ISI
Reid Smith	Schlumberger
Monte Zweben	Red Pepper/Blue Martini
Stephanie Dick	Historian, University of Pennsylvania (attended remotely using the Beam Robot)

Colin Garvey	Historian, Rensselaer Polytechnic Institute
David Grier:	Historian, George Washington University
Thomas Haigh	Historian, University of Wisconsin at Milwaukee
David Hemmendinger	Historian, Associate Editor IEEE Annals
Hansen Hsu	Historian, CHM, Center for the History of Software
Dan'l Lewin	CHM, Executive Director
Paul McJones	CHM, Special projects
Len Shustek	CHM, Chairman
Ed LaHay	SI SIG, Meeting Project Manager

Introducing Dan'l Lewin, President and CEO of the CHM

Burt Grad: I have the privilege and pleasure of introducing Dan'l Lewin, who is the new president and CEO of the Computer History Museum who's our host for this meeting. Please say a few words about the museum.

Dan'l Lewin: Well, thanks so much. Yes, I haven't acquired the title of Museum Director yet because I'm learning what that means. I'm relatively new to the institution, a couple months here.

For those of you who I don't know, and there are many names that are familiar names but I don't think I've spent that much time with too many people in the room, I've been living and working here in Silicon Valley since 1976. I stumbled into a group of people when I moved out here that fall. Some of you probably know Cordell Green. Cordell owned the house at the top of the hill where I was living, and people like Peggy Karp and others were there. Probably some of you know Peggy was working with Larry on things. So, I bumbled into this place. Some of you are here purposefully, but I sort of stumbled into it and got lucky.

As some of you know, I ended up going to work at Apple and got to work on the Mac. I took it out into higher ed and asked a lot of smart people what it would be good for - and they told me. They figured out how to bridge it into the Lotus network at Stanford and kind of help Cisco -- things like that. So, I've had a nice history in the Valley, but I'm not an engineer by training.

I was back in Upstate New York for Mother's Day this past weekend, and it turns out they did a thing at my high school. I spoke at my high school. I said I was fortunate because in second grade I had a teacher who taught us binary, so I got lucky. Then I went to Princeton and I realized very quickly that I was maybe good at math, but I was not a mathematician. I ended up with a degree in politics. I thought I might be a lawyer, but in the end, I just found that I love market development activity, and as I said earlier, I had the really good fortune of just bumping into really interesting people who were very helpful to me. I learned in a hands-on kind of way what it was like to play with SmallTalk and systems when the Lisa was not yet, and the Macintosh was not really an idea. I got to play with those things and me some really nice, bright people who still remain great friends.

Fast forward, after the last 17 years at Microsoft doing various and different things mostly on the political and technical front with people in the Valley and then around the world, I spent my time in an area of building a practice around technology and civic engagement and the idea of public-private partnerships where data was relevant and fundamentally important and regulatory and social systems really mattered. That's kind of the world that we're living in today. I got to play with both sides of the aisle on this last presidential campaign where the voter file was up in Azure in both cases, and they did what they did with them on each side. We probably need to drink beer if you want more stories about that because I saw a lot of it firsthand.

I also worked on the environmental sustainability agenda where the company built an internal carbon market and taxed people for using energy, and the company was carbon neutral as a result of it. I got a bug for the broader implications of computing in society.

Then after having been friends with the Museum for a long time and working right up the street, I was coming to a lot of programs because I was going to leave Microsoft. In the fall when I was off payroll and doing what I wanted to do, which was to think about what I wanted to do next, I was at a program here where Leslie Berlin was talking about *The Troublemakers*, her new book. I listened and learned, and a couple of the trustees pulled me aside and said, "What do you think?" For me, it's just a real pleasure. It's an opportunity to play with the topics that I love, to be with really bright people who care a lot about where technology's going and those who built what is.

The Role and Contributions of the Computer History Museum

Lewin: With Len and the rest of the board, I'm now about two months into a process. Ed asked me briefly over lunch, "What have you been doing for the last couple months?" I've been learning about the things that I don't know, so I'm finding out what I don't know about museum operations, and I've been spending time with others out in the market at the other museums and learning about the skills and the capacity of the organization.

My belief is that we have an incredible frontier in front of us for the next 10 years: how we interpret and present what's captured here in these archives, both the physical artifacts and the oral histories and the activities that you're all bringing about through workshops like this, is just astounding. That'll be my challenge to think about how we go along and build this institution from the assets that exist and the new assets we can bring to the table.

That's a little bit about me in the context of the Museum. I wish I had more time to spend and listen and learn. That's one of the benefits of being here. Happen to be in the middle of a busy week for a variety of topics, unfortunately, so I won't be able to spend too much time, but I'll have the opportunity to be debriefed as well.

Grad: You're going to spend dinner with us.

Lewin: I will be at dinner this evening, and there are some slots tomorrow where I think I can wander in and out, but it was tough for me to project and predict a whole day. I've been told as long as I sit down and I don't complain about the cold, it'll be okay.

Grad: Any questions for Dan'l?

Peter Hart: The obvious one. What's the biggest surprise you've had?

Lewin: How much opportunity there is before us is the thing that's surprising. The pieces that exist here are extraordinary. In many cases, they're the definitive foundational elements to the history. Gordon Bell and I have been friends for quite some time going back to when Apple talked a little bit to Digital a long time ago. Then I was at Microsoft for 17 years pretty much in parallel, but a little bit behind Gordon.

Just the depth on so many different fronts, it's fascinating. The physical plant, the building—we sit here on this 7-1/2 acres as the hole in the middle of the Google donut if you will. There's so much we can do with the physical presence that we have. The oral histories are stunning, and we're capturing more of them. I've challenged David to help figure out how to scale some of this stuff, because I think there's a way to scale it.

What I see are elements that have been built up purposefully and individually, and I described this in a board meeting. I have had one chance to talk to the entire board and a number of meetings with the executive committee and others, but the thing that I've found fascinating is the depth and the strength of the tap root in the institution, the histories, and the archives. The things that have grown up from it in some cases may have been kind of grafted on. It doesn't make them not strong or good, but they're kind of grafted on. What I don't see yet, which is the big opportunity, is the bark that makes it look from the outside in as if it's fully integrated.

As that grows and scales, what do we do to keep the root system growing? As you know, as the canopy grows, the root system needs to be appropriately fed and stretched. That process of architecting a strategic plan, a long strategic plan, is based upon the capabilities and capacity of the organization. What I've been running is a series of exercises across the organization to understand the work functions and the individuals, down to the individual level—it's 75 people, plus volunteers and all the other activities.

How we partner to drive that scale? The place on the outside in looks larger than life in many ways, but when you get inside, you realize, it's not larger than life, it is what it is. It's like any organization of this size and scale. How we partner and build up to that? The scale could be incredible because we're not too far away from the idea that we could have an opening on a regular rhythm in this facility that appears instantly in 10 places on the planet. There's no reason why we can't do that from here. We don't have the technical infrastructure that we need. That's doable. There are some people around here who could help us fix that. You know, it doesn't take a huge amount of money. We can partner to make some of those things happen.

In every area that I look, I see raw potential but I don't see the integrated, fully captured story, which is the tree that looks like the beautiful tree as opposed to, again, things that might be grafted on in a particular way.

<u>The Future of the Computer History Museum</u>

Thomas Haigh: I'm a historian who's working on a book covering the overall history of the computer, and one of the challenges of doing that is that as far as a computer scientist is concerned, there are way more computers doing more things everywhere in the world than ever before. I think you have a parallel challenge with the museum. But as far as an ordinary person is concerned, they don't see computers anymore. Their phone is a computer. The computer as a category, the end of that is in sight. As a museum that's called the Computer History Museum, how are you thinking about dealing with that challenge?

Lewin: That's an open-ended question you pose that I certainly see. I look at this as, in the next 10 years, we need to be in a place on the other side of five years from now where we look at this institution and the fact of the matter is computers have faded into the fabric.

I mean, 10, 15 years ago, with the beginning of web services, using web protocol to move data elements around and Ajax and all that stuff, we started talking about, "It's the rise of the big machines and the little machines. It's the rise of the machines." The big machines are not visible anymore because they're in the cloud. And the little machines are the sensors that don't need power because of they're pulling it from the ambient environment. We're going to be in a place, which we're experiencing today in spades, when you look at with people who have these devices, the digital divide from the standpoint of the way we think about computing, where you put things into a computer.

It was really about automating rational tasks for a long time except for some of the work that lots of you have done. Start to think about how machines can do things in a different way. But so much of the industry that I grew up in was about automating words and numbers and pictures and things like that. That's interesting but not where the world's going, and there are so many people who don't know anything about opening or closing a file or saving anything, they just consume information.

I envision a world that you all do as well where what should happen around you should be based upon the permissions that you grant. You are a market of one as opposed to what the industry did, which was to default the identity position to a couple of keepers and the industry basically emerged for better or worse the way it is today.

Maybe we need to think about that as an industry: what does it mean for your presence on the network? That gets into the social and regulatory structures, which is where I spent the last four or five years of my time. What are the nation, state, and regulatory boundaries that we're all going to have to live within when systems, again, present themselves to you based upon who you are and where you are? Complicated.

As an institution, we have a great place from which these stories can be told. We have convening power that hopefully will be neutral and visible through the partnerships, because we won't be able to do it all ourselves. We'll need other storytellers to work with us on that, and I think building that ecosystem out is important.

Grad: Dan'l, thank you. I'm throwing you out. You can still sit there, but if not, we'll see you at dinner.

Lewin: Okay, I'll be back at dinner. Thank you all so much. Thank you. Thanks for having me.

Grad: Thank you so much, Dan'l.

David Grier: Author and Historian

Grad: We have one other person to introduce: David Grier. Three minutes, maximum. Why are you here?

Grier: I'm David Grier. I'm here because Burt seems to like me and wants me at his meetings. I am the sort of engineer that Middlebury College produces where it considers poetry to be a practical experience.

I have been President of the IEEE Computer Society back when it had money. Editor of the *Annals*. I have taught expert systems at George Washington University and logic programming. I live in Washington, D.C.

Grad: David wrote a wonderful book called *When Computers Were Human*. His grandmother became a math major at Chicago University. Do I have it right?

Grier: Michigan, please. Go blue.

Grad: I'm so sorry. I've been telling people this for years. Anyway, she motivated him to look at this. He wrote a wonderful book, if you ever get a chance to read it, a lovely book about the people who did computing and were called computers up to when, 1955?

Grier: Oh, much later than that. The federal government had a category for them up to 1988. The last government organized them, and I need to say that *When Computers Were Human* is about organized computing labs. There are lots of people who did it for somebody as a grad student or something, and they're lovely and I don't care.

The story that Burt refers to, in my mind, is what I have constructed as my last dinner with my grandmother before her passing. We were gossiping about cousins—you have cousins, and you know what they're like. They're solely for us to gossip about. We were talking about Laura, who remains a dear friend, who was on her fifth of six husband or something like that, and my grandmother put down her fork, stared me at the eye and said, "You know, I've always been sorry I never used my calculus." Grandmothers often feel regrets in their life, but the use of higher mathematics is very rare among any gender. For reasons I don't know, I let that go. I had a Ph.D. in Mathematics at that point, and for some reason, I let that slide away.

When she was gone, I asked my mother, "What do you know about this?" She looked at me and said, "She didn't go to college. What do you mean?" I said, "She told me this," and she said to talk to her brother. So, I called up my uncle and I said, "Uncle Chuck, what do you know?" He said, "Well, she was not well at the end." That was not at all true; she was hale and hearty to the last moment. She kept the books for our family gas station in Ann Arbor, Michigan. If you have bought gas in Ann Arbor, we thank you and hope you'll come again and were satisfied with the service. He had me talk to some cousin of his that I'm not sure how I'm related to, who told me the whole story—the immigration from Ireland; The sea, the sea, the wine dark sea; Ellis Island, and so forth and so on—and said, "Dearie, your great grandfather was a drunk, and your great-grandmother cleaned hotel rooms. Your grandmother did not go to college."

I thought, "Well, here's the deal. I'm a professor. Let's try it." I called up the University of Michigan, and I said, "Hi. I think my grandmother might have gone to college there. Can you give me her transcript?" As a warning to all of us, they said yes without batting an eye.

Five minutes later they were back and they said, "Yes, yes. We have a transcript for your grandmother. Did she graduate?" I said, "Yes. 1921." I'm going, "Mom was born in 1923. This is it. This is the right woman." "What did she major in?" You hear the flipping of pages then she said, "Huh. She was a math major."

I went to look at her records and I discovered that in 1921 Michigan graduated 12 math majors. I did this work in 1995 or something, and that year they graduated 12 math majors and a whole bunch of other majors that use math. Of the 12 math majors, six were women.

Grier: The reason for that is largely due to the fact that women had just gotten kicked out of medicine. After the Flexner Report of 1910, women did not have full doctor rights across the board. They couldn't be surgeons or found it very difficult, but they could do the things we would associate with women's work: babies, pregnant women, dying old people. There were a whole bunch of women who expected to have that job. They were professionals and they could prescribe and they were being denied it and they were mad.

Then as now, being desperate for enrollments, math said, "You could become a high school math teacher. Come on down." Immediately, a whole bunch of mathematicians got feminine.

Grad: Wow. I'm going to cut David off. He's a wonderful raconteur. He has all kinds of blogs and stuff he writes. Trying to shut him up is not an easy thing. Anyway, David, thank you very much for being here

Expert Systems versus Knowledge Systems

Grad: Okay. Here we are. We're finishing up the morning session, and then we'll go into the companies. Let's start with where we were. There were three people who wanted to comment, if you remember after lunch what you wanted to comment on.

Doug Lenat: Okay. My comment was that just because we've been calling these and have historically called them expert systems doesn't in any way mean that there's something special or necessary about modeling an expert's cognitive model of some subject. This technology and this approach can be and has been successfully and importantly used to model things other than an "expert" level of knowledge and what concepts they know, what rules of thumb they use, and so on.

For example, we use that in one of our applications to model various types of middle school students' knowledge of math. It's very important to model the typical variously confused middle school students so that we can help diagnose and teach them. We don't want the correct way to do long division; we want to model the various common incorrect ways that kids do long division.

Grad: Okay. Next.

Mark Fox: I just wanted to reemphasize the perspective I have on expert systems and, by doing that, in contrast to the broader term that I think we are using here in some sense, which is knowledge-based systems that Ed talks about. I've always viewed knowledge-based systems as having as its major component knowledge, but not necessarily a commitment to a rule-based approach to reasoning, not necessarily a commitment to a particular style of memory, etc.

When we talk about expert systems, in my mind, it's a narrower interpretation, which has a much more rigid commitment to a particular style of reasoning, a particular way of representing knowledge, a particular way of doing inference. But a lot of us weren't limited to expert systems in that narrow sense. A lot of us built knowledge-based systems that took a variety of technologies, a variety of ideas that exist within the AI community to build systems that solve problems.

To me, that distinction is always very important because when we talk about expert system tools, at least back in the 1970s, 1980s, we were talking about these more rigid rule-based systems than we were just broad AI systems. That's one thing I want to say.

The second thing is, Burt, I think you were talking about linear programming and how that is different. Well, when I started working on scheduling problems in the late 1970s, early 1980s, the very first thing that we did is we went to OR [operations research] and we said, "Okay, linear programming, it gives me an optimal solution. Oh, we have a problem. We can't represent the problem. Okay, integer programming. Well, we can represent the problem, but there's no good solution techniques. We had to go elsewhere." That led us into constrained approaches, problem bases, heuristic search, things of that nature. There were severe limitations back then. Fast forward to today, mathematical programming is a lot more powerful, but it still uses a combination of heuristics along with other techniques.

Blackboard Systems

Fox: The last point I wanted to make is that I had the opportunity to work on the Hearsay-II Speech Understanding System in my first year as a graduate student at Carnegie Mellon. I don't know to what extent people view it this way, but the Hearsay-II was an expert system. It had blackboard, which is your working memory. It had rules. They were very heavy rules—that is, they were called knowledge sources, which contained a deep amount of knowledge about their particular task. What they did is they had precondition, which was looking at the blackboard to see what's there. That would fire the knowledge source, which would then post new things on the blackboard, which was our working memory. It had a representation of uncertainty and operated very much in the mode of what we refer to as expert systems. But nobody used it that way. Did any of the blackboard technology survive the 1970s?

Grad: Sure.

Fox: IBM Watson is a blackboard system. If you look at the architecture, it's a blackboard system. There are hypotheses. There are these rules in there. There's posting hypotheses, levels of uncertainty; all that type of stuff is going on, so this technology hasn't disappeared. It continues to evolve and get embedded in systems. Nobody puts up a flag and says, "blackboard system," "expert system," or whatever the case may be on these things, but the technology's there.

Heuristic versus Expert Systems

Brock: I think Ed wanted to address Tom's earlier question about the import of the phrase "expert systems" itself.

Ed Feigenbaum: Yes. Sorry, everyone, I have to force these words out from the larynx. I apologize for that. Tom asked a great question because it provoked two lines of thought in my head, both of which had not come up in the morning. Now the first of which did come up with Doug and Mark's remarks.

After doing a number of systems where we were aiming to replicate the behavior of the best people in the field, we found that there were a lot of applications where it didn't matter whether it was the best in the field: the computer could do it and it could do a little bit of reasoning, a little bit of Aristotelian logic, and so on, and it would be useful. So, we changed the name of our laboratory: Heuristic Programming Project morphed into Knowledge Systems Laboratory, because that's what we were building, knowledge systems, not expert systems.

The other thing is much more subtle. I only speak for myself, but if any of the other AI people around the table feel like I do, maybe they can say it. But in my mind, in my strategic plan in my head, what I was doing in the 1960s and the 1970s and the 1980s was to aim for the smartest possible X, whatever X was. If we were going to be a diagnostician in blood infections, then it should be the country's best diagnostician in blood infections. In fact, we did metrics like that. When I worked on a classified project, the DARPA one detecting Soviet submarines off the coast, DARPA did an actual test of that expert system to show that it was better than any of the people, any of the human beings who were manning the shore stations. Of course, the Navy didn't use it, but there really was a subtle thing there. What comes to mind is I.J. Good's paper on the ultra-intelligent computer. That was sort of in the back of my mind all the time, and that's why at the outset the word expert came up as opposed to just anybody, a knowledge system.

Defining Knowledge

Grad: Thank you. Hansen, you had a question?

Hansen Hsu: Yes. I have a question that's similar to Tom's. Ed, you sort of spoke to this as well. You used the word "knowledge," so my question is what exactly do people in this room mean when they say knowledge, capturing knowledge, and knowledge base? I had a conversation over lunch with Reid here about that. I'd like to know how other people in the room define knowledge and what exactly is being captured. Is it knowledge purely in terms of problem solving? How does it deal with tacit versus explicit knowledge, etc.?

Grad: Thank you, Hansen. Anyone like to tackle that?

Feigenbaum: I'm going to start the answer to that question with another way of looking at expert systems, not the way I initially thought of it. I remember one day I was giving a talk and some person in the audience came up afterward and said, "Oh, now I understand. An expert system is a program that can explain itself. That's an expert system." Now the answer to your question is: the knowledge is what constitutes the explanation, the level of granularity that the expert in the field recognizes as an appropriate level of granularity from step to step in the explanation.

Incidentally, I've spoken to a Justice of the Supreme Court in California who was asking me about neural networks, and I told him what he would get as an explanation from a neural network as to, let's say, why the Tesla crashed or something like that? He said, "Ed, that won't do. Lawyers need stories. Judges need stories."

Grad: Interesting. Anyone else want to comment on that question from Hansen?

Brock: Knowledge or the phrase "expert systems?"

Fox: I'm going to come at it a little more technically than Ed did. You have to look at it from the way the field was shaping up in the late 1960s, early 1970s. In addition to the more rule-based approach to reasoning where the focus was more emphasized the rules, there was also the semantic net knowledge representation side that was being pursued in parallel. — Relatively speaking, for the field at that time, there was a large number of knowledge representation languages that were being created. The focus on that was not on rules; the focus was on representing prototypical descriptions of things, whether it's a car or a chair or whatever, trying to get to create definitions of things using a more precise language.

Now that morphed over time into the use of logic for describing this, description logic, etc., and stuff that's going on in the semantic net on the internet. When we talk about knowledge, we talk about two types of knowledge. We talk about the things that we want to represent in this world, whether they be physical or conceptual, and we talk about the types of deductions or rules that we want to have. Together they constitute the knowledge that we want to acquire, and depending upon which system you focus on, it will be heavy on the rule side, light on the knowledge representation side, or it may go the other way, or a combination of the two.

Normative Expert Systems

Grad: That leads right to where I wanted to go. That's why I started the question with you. What are the elements of what we've called an expert system? One, you talk about the knowledge base itself and its representation. Is that the wording that you used at some point?

Lenat: Yes, I'm sure.

Grad: Number two, there is the capturing of the rules—he likes that word—or the logic that people use with that information to come to some kind of conclusion or decision or recommendation. Third is the inference center and the mechanism that's used to carry that out. Am I at least on the right wavelength with those three things? Peter, I figured you would tell me I'm wrong.

Hart: You can tell him he's wrong.

Brock: Peter Norvig, please. Begin the assault.

Norvig: I was going to introduce another distinction. I noticed Eric Horvitz's name tag is outside, but he's not here. In the mid-1980s, he came up with this term of a normative expert system. The distinction there is, should you ask the expert what do you do in this situation? Then you're encoding the expert's problem-solving techniques. Or should you just encode their knowledge and then ask them, "What do you know, and what is it that you're trying to maximize or optimize? But don't tell me how to do it because maybe I can come up with a better way to do it, given what you know."

That's a normative system. It's trying to say, "Given everything I know, here's the optimal path". That's how we get checkers-playing or chess-playing programs that are better than the experts because they take what the experts know and then they build on top of that.

Grad: They use a difference process instead of the expert's reasoning process for solving it.

Norvig: Right. You don't ask the expert, what you would do? You ask the expert, what do you know? Then we'll figure out what to do beyond that.

Grad: Peter, you would disagree or agree with what he said?

Hart: No, I agree with that. I was going to make a different point.

Rules versus Logic

Hart: Burt, I think your first two system elements can be collapsed into one, and it can be a little bit simpler. You used the term, I think, knowledge base.

Brock: Yes.

Hart: And rules or logic.

Grad: Yes.

Hart: They're the same. The knowledge base is just another name for the body of declarative information. Not procedures, in other words, but assertions that constitute the expertise that you've gotten from an expert—for example, diagnostic medicine, medical expertise. It's just one and the same thing. Then the inference engine is the thing that operates on that.

Grad: Let me go back to my example earlier. You have a particular product line. I have a copy of all the blueprints, all the bills of material; I have all that I considered my knowledge base. That was existing information. I then talked to each of the engineers and got their reasoning: "Why did you decide this? Why did you decide that? Why did you use the shaft this way? Why is the shaft this long? What information did you get on the input side that enabled you to make that decision?" I considered that the logic and the rules.

Hart: If I understand your example, the way I would describe that is that what you're calling the knowledge base, the blueprints, unless they're interpretable by a computer program, they lie outside of this discussion.

Grad: But they are interpretable.

Hart: Well, did you have a computer program that could look at that blueprint and decide how to repair it or operate it?

Grad: No. But I could look at the blueprint and I could look at the bill of materials and say, "Oh, I need this many screws, I need this many bolts, or I need this kind of grooving to have to do it." That's what we could do. Now, I'm talking about a very old problem back in the 1950s.

Hart: The problem is fine. It's a fine example. But I would suggest that the way the expert system community would probably look at that… and here we have the expert system community, so tell me if I'm wrong.

Brock: Do you agree with him?

Hart: The way I would describe that is, Burt, when you said, "Here we have a 632 screw," or something like that and so on and so forth, and you put that in a form that can be interpreted by a computer program to solve a problem—whatever your problem is, how you assemble it, how you troubleshoot it, whatever the problem is—that information about the 632 screw and what it's for and when to move it or tighten it or something, that's part of the expert system knowledge base. The blueprints that you, the expert, looked at to say, "Oh, here's what that 632 screw is for," is not probably going to be.

Grad: You're a step too far. I'm behind you.

Traditional Programming versus AI Programming

Brock: Please correct me if I'm wrong, but I think Bruce Buchanan would talk about what you were doing there, Burt. In that situation, you were doing the kind of knowledge engineering or you were extracting information from the experts that you were going to then encode as a rule. That would be an if-then. If you're in this situation, use that shaft. That is the kind of knowledge representation.

Grad: We created a new bill of material. We created a new blueprint for each new order that came in. We created blueprints that were unique for what that customer said he was looking for. I was trying to see where that fits into your dialogue.

Monte Zweben: I think what you're getting at is that this whole field and everything that we're talking about—and this is something that AI's been doing for a long time—is an abstraction on programming that is a very specialized perspective. Peter said the magic word of "declarative." That's what we've always been trying and driving to do. Every computer program has a persistent state, a set of inputs and a set of outputs, and it exhibits certain behavior. Traditional programmers use programming languages to take that transition function from the inputs to the outputs, and you change the state that it's in. Some programs use databases. I work on databases now. Some programs use knowledge bases. Some programs use files. There are all kinds of ways of representing persistent state and mapping inputs to outputs to exhibit behavior.

But what we're talking about here is a particular approach to codifying that persistent state and to codify that behavioral function of inputs to outputs. What we've always been trying to do, I think in expert systems, is make that as much declarative as possible. You weren't writing lines of code; you were writing chunks of in some cases rules, in some cases mappings of probabilities and maybe graph structures for semantic networks. You were just trying to make it more declarative so you can apply this program to more than one specific problem. That's what we're trying to distinguish with you.

Norvig: The way I often try to explain it to a lay audience is regular programming is for when you know what to do. It's like having a recipe for how to bake a cake. AI programming is for when you don't know what to do, when the main complexity you're dealing with is uncertainty rather than just the amount of data or number of lines of code or whatever. There, traditional languages don't work because if-then that can only be Boolean true or false, and if you don't know if it's true or false, you're stuck. So, you have to break up that code into smaller pieces where you can say, “Here's a rule and maybe I'll use this or maybe I won't, and I don’t have to restructure my program. I can just throw in more rules and let the inference engine worry about dealing with the uncertainty.”

Brock: Tom, do you have a new question, or were you on this?

Haigh: Working on the history of ENIAC and Colossus and other machines from the 1940s, that's when people first started talking about programs. There's been some historical discussion—David actually contributed to it—about why we call it a program. My answer with Mark Priestley is that it's a very simple extension of how we talk about programs for, say, a degree program, a lecture program, a concert program: selecting and sequencing operations over time. A concert program says we play this, then we play this, then we play this. A computer program says you do this, do this, do this.

Now, obviously, the meaning of program has evolved over time. I have a personal belief that within that original sense a Prolog program isn't actually a program. You were saying, “Well, Prolog you're not writing code- it's declarative.” I just was wondering, is the opinion of the expert system community that a Prolog program is a program, or do we need a different word for what a Prolog program actually is?

Brad Allen: Yes. No. I mean, I think it goes along the lines of where we've been getting down to in this discussion, which is the thing that's distinguished whether there's a declarative portion of what you're building. Prolog is essentially a set of Horn clauses define the Prolog program. The control that actually applies that to a problem and comes up with an answer or does what it's doing is something that is not necessarily modified for the particular task at hand.

The idea there was by moving the bulk of effort into the declarative specification of what you knew you could have the inference engine or whatever the procedure was. Because it could be very prosaic, it could be something just kind of running down through a set of checklist items or whatnot. But that was something that you didn't have to go and pay attention to. You dropped the knowledge in, and you would have the thing that would be able to draw conclusions directly to satisfy the needs of whatever task was trying to be performed. The bet was that that was a big step forward in terms of how we could build those kinds of systems.

Methodology for Collecting the Knowledge Base

Barr: Before we leave the topic of what constitutes expert systems technology, we've talked about the knowledge base’s way of making declarative knowledge available and the inference engine that uses that knowledge. There was another piece that's important and that we should not miss, and that is the methodology for acquiring knowledge from human experts.

Grad: Seriously, how did we collect and store the information? How did we represent it? How did we make it available? How did we update it?

Peter Friedland: A couple things that I don't think were mentioned very much so far. One of them was your question. The other is just a note that one of the things about expert heuristics is that they enable you to leap through a search phase, avoid the combinatorics of a search space, avoid backtracking at times. So where traditionally AI systems and AI texts had talked about ways of navigating a very large space, search space, combinatorics, with many of the expert systems we developed, the rules, the heuristics, however they were encoded, enabled experts to make leaps through a very large search space and to avoid backtracking.

Knowledge Engineering Tools

Friedland: The other thing which Avron brought up and then you brought up was the tools for knowledge representation. They were touched on a little bit, but a very important part of the technology being developed were tools like that. For example, there was the KL1 language. Who developed KL1 first?

Brian McCune: Brockman.

Friedland: Yes. Thank you. Ron Brockman's system. At Stanford, it was something we called the UNITS package and which I worked on and Reid worked on and Mark Steffek worked on. There were other tools like that which simplified the process of encoding, so they were both heuristics, but also the declarative knowledge about the problem. What were the important concepts in molecular biology, for example, in my system in MOLGEN? That was an important technology to build.

Grad: How did you find out what those were?

Friedland: Well, okay, that's the other question you've asked. Most of that was done by the process that I think is in one of Ed's books. Didn't you talk about Penny as the prototypical knowledge engineer?

Feigenbaum: Yes.

Friedland: In your book on the expert? Was that in that book?

Feigenbaum: Yes, that's in that book.

Friedland: So, read Ed's book if you want the details. Or get him to give you a copy for the display.

Grad: Okay, fine.

Friedland: But mostly it was a human process. It was a process we as knowledge engineers, as we proudly called ourselves, used to interview, to talk with, to bribe domain experts.

Grad: You interviewed and you talked. You got the information. Then you recorded it in some form.

Friedland: Yes. In almost all cases, it was a human-facilitated process. We created knowledge engineers.

Grad: There were a bunch of different forms of how you represented that information you collected?

Friedland: There are a bunch of different tools.

Grad: Each place had or did their own?

Friedland: We shared.

Grad: MIT [Massachusetts Institute of Technology], did they have their own? Did Carnegie Mellon have their own? Did you do your own just to be different?

Zweben: Yes, I think that's where everything sort of differed. The subtlety of the representations was how everyone differs, how we represent it as knowledge. Some people use rules. Some people use semantic nets. Some people use frames. They were all really in the same thing.

Friedland: Right.

Zweben: Some people use logic because everyone's different.

Grad: There was no standard representation. That's what I'm trying to get at.

Friedland: We started evolving standards as we started building tools. One of the things that will come up in the next topic was a very important part of what was being sold by the company **were** standardized tools for in theory making it easier for people to become knowledge engineers.

Grad: Good. That's what I was hoping this would lead to that.

<u>Control Structure</u>

Lenat: When we get to the point of talking about expert rule, elicitation from experts, Peter Hart really is by far the best person I've ever personally observed doing that. I'm interested to hear what you have to say about that. What I wanted to comment on was I was just waiting until either Avron or Brad or Denny said the words "control structure," and then I immediately raised my hand.

There are additional components besides the main ones we talked about. Control structure is one of them. The methodology and process and tools for elicitation of expert knowledge is another and so on. But there's an important point which is actually moving in the same direction as Peter's last comment, which is that when possible, we should keep applying the same methodology recursively, repeatedly so that in fact the best kind of control structure is not a wired-in control structure but one where you have basically declaratively represented meta-level rules that can in effect as you run them be your control structure and so on.

Instead of, say, speeding up an expert system by extirpation, by removing parts of its brain or something, you should speed up an expert system by adding additional meta-level knowledge or meta-judgmental knowledge or something like that, the same way that presumably experts get to be experts: not by removing parts of what they know about the world, but by adding more experience and more wisdom and so on.

The same thing applies even to methodology for acquiring knowledge from experts. More and more we should have knowledge-based tools, expert system tools for interacting with experts to help them to give knowledge.

Abduction and Explainable AI

Lenat: I'll just mention one such example which is using not deduction or induction but what I think is actually the most important category in a way of knowledge in our expert systems, which is abduction.

Reid Smith: I knew somebody was going to mention abduction.

Lenat: For example, a subject-matter expert watches on a system. Do something wrong and he says, "This is the right answer you should have gotten." Instead of having to look at the trace of rules and so on, the system itself knows enough ideally to be able to say, "Here. Given everything I know already—domain knowledge, common sense knowledge, whatever—here are four or five possibilities which if any of them had been true would have gotten me to the answer you wanted." If the expert is able to say, "Oh, yeah. It's number four," then the system itself has helped with the knowledge-elicitation process.

Grad: I see.

Smith: I want to come back to the explanation. I was asking myself what was really exciting the first MYCIN paper I read, which is one of the first I read. Well, you know what it was? It was that the system could explain its reasoning. I think that's really super important because it speaks also to the way in which the expert knowledge is encoded.

My experience with this primarily is in the world of geology and geophysics. The rules, which we consider to be a fundamental part of the knowledge base as Peter has suggested, they made sense to other people who were practitioners. They were devised by using elicitation methods, knowledge engineering methods you all know very well.

Let me back up a little bit. The particular expert in this case was a Gulf Coast in the U.S. dip meter interpreter, a geologist kind of guy. I could show the rules, but it wasn't like, "Here's the code. You read it," like a computer scientist would. These guys are not going to read code. They couldn't read code. But they could look at the rules and say, "Yes. Okay, I understand. That makes sense," so the explanation then was easy. In the MYCIN case, it was running through backward chaining and saying, "Here's how we got from one step to the next going backwards," as you suggested.

I think there's something about the granularity of the knowledge that's captured that makes sense as an explanation mechanism to other practitioners; it's a confidence builder. But of course, as Doug indicated, it is a way of helping to debug. You're now talking to a debugger in a completely different way than any computer scientist is familiar with.

Let me speak just a little bit to abduction, since we actually did write a paper.

Friedland: Before you do that, can I react to Reid's first comment?

Smith: Yes.

Friedland: This reminds me, since we're all old and thinking about history, except for Monte who's still very young.

Zweben: Well, some of us are looking it up now.

Friedland: One of the things that is a very important hot trend in modern AI—where everything old is new again—is XAI, explainable AI.

Brock: Right.

Friedland: Going to the deep learning systems, it's not enough just to have a right answer, you have to explain why. For a big DARPA project, big investment, there was a huge number of people at a workshop on the topic at the last IJCAI/ECAI. The notion of the system being able to explain itself has gone from being a hot topic 35, 40 years ago to now being a hot topic today.

Smith: Let me speak to abduction for a second, since Doug brought it up. We wrote a paper in 1985 on abduction. We were looking at the rules of the dip meter advisory system and asking why they're true. Because of course, the cruel, dirty, underlying fact is that they are compiled knowledge of a sort.

In other words, they are correct with a certain amount of context built into the rules. If you were to ask any one of them, "Why is that true? You said that this pattern had to be this long and over that one and so on." In fact, when you backed it out down to other fundamental geology, there was a whole lot of local context being built into these things. But it was a useful way to go and look at the knowledge. You could look at it independently of how it was applied in any given reasoning situation. To me, I thought one of the fundamental components of expert systems that I still believe is a fundamental component is the ability to explain.

Job Aids versus Expert Systems

Grad: Go ahead, Paul.

Paul Harmon: Let me give, again, an outsider's point of view in a sense. I came from a training background, so when I first encountered expert systems at Teknowledge, I was fascinated with the technology. But the thing I immediately thought was, in addition to whatever else is going on here, I can take this back to International Society for Performance and Instruction, a training organization, because here are tools that they can use to build what we called job aids or checklists.

Today, checklist is the popular term. Training people found that one of the best ways to get people to perform reasonably complex tasks was to give them a checklist that simply walked them through do this, then do this, then do this. Most at the time computers were new. Most people in the training field couldn't program. All of a sudden, here were tools where somebody could just enter rules: If this is true, then do this. If this is true, then do that. For job aid purposes or checklists, you could write a set of rules that was maybe had 20 rules, and they would provide six or 10 questions and they would provide an answer.

A year after I started working at Teknowledge, I gave a paper that won the award that year at the training conference on how to build knowledge-based systems. I would never have called them expert systems. I used to speak of expert systems as having thousands of rules and job aids as having 10 or 20 or knowledge systems, if you would. The tool supported itself. The inference engine was built in.

If you were trying to deal with human experts, you had to represent relatively complex knowledge. If you were trying to deal with people in many training situations, it was a matter of 10 or 20 rules. The tool could handle that. You simply taught people how to design rules. They would plug them in. The inference engine was built in. It was automatic. You'd turn on the tool, and it would run. All of a sudden here was a programming tool that allowed training people to quickly build systems that could be used in a training environment. At the time, the expert system people thought it was absurd that people would even want small tools, but the small tools proved to be very valuable in doing these kinds of tasks.

Eliciting Expert Knowledge

Grad: Let's go back to where I was. Help me here. As we talk to each of the companies, we want to talk about what it is they were contributing and doing. Apparently, they were sort of doing these things on an ad hoc basis, whichever the problems you solved, you worked out a way to talk to the expert to gather the information to build the knowledge base. You had to have an inference engine of some sort that would execute what you had done. Now, we're talking here that somehow you saw the possibility of having companies make money out of either offering solutions or offering tools of some form. Is that a correct statement?

Friedland: Yes.

Grad: In each of these topics, obviously, we could have gone on for hours. In some cases, we'll explore deeper. We're doing a few oral histories and maybe some others that we'll do later on. We'll go in depth in some of those with you individually. Here, we're trying to get an overview, trying to cover a lot of ground. We can't dig that deep in any of these things. I'm hoping that some of these things will come out as we talk about the individual companies. Anyone else have anything that they feel they need to add before we go into the companies?

David Hemmendinger: In a discussion of knowledge engineering, I kept expecting to hear the term "tacit knowledge" and that didn't come up. Meaning that a lot of the knowledge that experts have is tacit. It's hard to elicit it.

Grad: What is tacit, implied? Understood? Assumed?

Feigenbaum: The myth is that it's hard to elicit, that you have to buy into that whole tacit story. It's a myth that it's hard to elicit. Be a good knowledge engineer, then you can release it and elicit it.

Hemmendinger: Yes. But it's not always easy to elicit it.

Grad: Stop. I got lost here somewhere. Peter, can you clarify what he's saying?

Hart: I've got to make one comment about Ed's thing, which is generally that it is easy, but I have to tell you one short story about the opposite. When I was interviewing the Senior Vice President of Underwriting at one of the largest commercial insurance companies in the country, very quickly I saw I was going to have to get down to very simple cases to get the terminology right. Here is what this underwriter told me. This was a Senior Vice President of Underwriting, a very highly paid guy in a very responsible position. The conversation got to the point where we were talking about coin flipping because I was trying to get probabilities into this, just understanding probabilities. And he said, "Oh, no, Peter. If you flip a coin nine times in a row and it comes out tails nine times in a row, it's due for heads."

Hemmendinger: Oh, my God.

Hart: I saw right away I was going to have to adapt my approach.

Grad: Tell me what insurance company so I can avoid them. <laughs>

Hart: I mean, I was just stunned. This is a sophisticated underwriter.

Feigenbaum: I want to modify what I said. If there were tacit knowledge that we couldn't elicit, then that's a really good reason for AI systems to be man-machine systems, not simply autonomous, because you have that tacit knowledge. You can watch the program unfold its reasoning.

For months Doug [Lenat] did this with a program. He can tell you about the game that he was playing, but he would steer a program with knowledge that he had not yet articulated and put into the computer. He had to stop at the computer every morning on his way to work to give guidance to the computer for its so-called tacit knowledge. I think AI systems are just going to evolve that way. I don't see much future in autonomy. We humans just know too much.

Grad: And have a difficult time classifying, categorizing, and representing. Brad, we've got to get on. The last topic.

Allen: Sure. Then very quickly. The issue of communication and improving human-machine communication is an interesting way to look at this. The move to go to declarative and to incorporate processes and elicitation and so forth was an attempt to solve a problem.

In the long run, the idea would be that you would have natural-language understanding; it would all work and you would just talk to the machine. As an intermediate stepping stone, being able to get to a place where you can involve more people and communicate what the machine need to know to perform a task is really kind of where we're getting to with this issue of focusing on the declarative, having a standard automating procedure that would apply to all that. It was an attempt to make it easier to be able to convey information to the machine. The explainability bit of it was coming back the other way. It was a signal step in improving interchange between man and machine; that I think is a way to think about it, in terms of the differentiation of the work that was done in expert systems. It was an attempt to pragmatically improve that problem.

Transition to the Commercial Sector

Friedland: Burt, there's one very important thing that you need to understand historically to make the transition from the university work to the corporate work, a nontechnical topic. Most of the work we're talking about in the early days was funded by the federal government, research grants to universities. Until about 1977 to 1978, it was viewed that all of that work done under federal funding was in the public domain; it was owned by everyone.

Spurred by people like Niels Reimers at Stanford who was Head of the Office of Technology Licensing, the argument was made to the federal government successfully that being in the public domain meant that nobody would take it into commercialization. They convinced the government, starting at the NSF [National Science Foundation] and the NIH [National Institutes of Health], that universities would own the ownership rights with the assumption that they would license it to people, often the developers, to enable them to take it into the commercial sector. In many of the cases that I know about, that made it possible for the people like us who had worked under federal funding to then build the companies we're talking about. I just wanted to make sure that transition was understood.

Grad: That's a wonderful segue into discussing the companies. I have a group of companies that theoretically, according to Ed, were the earlier companies. I may have screwed this up, so I want you to help. Here's a list of companies that I think were earlier, and I'd like to do it in somewhat chronological order, except I got to make sure I get Mark Fox in before he goes today. I got a list of all these 12 companies here, maybe more. Here's the one I have for the first session: Machine Intelligence. How early?

Hart: We were just saying, Gary, this morning MI was when?

Gary Hendrix: 1978, I think.

Lenat: Yes.

Grad: Okay. Fine. Don't argue any finer than that. AI Corp., anybody remember what the timing was on that?

Hendrix: Approximately 1981.

Grad: 1981. Carnegie Group?

Fox: 1982–1983.

Grad: Symantec? Gary?

Hendrix: 1982.

Grad: Intelligenetics and IntelliCorp?

Friedland: 1979.

Grad: 1979. Teknowledge?

Lenat: 1981.

Grad: Advanced Decision Systems, under original name, AIDS?

Fox: About 1979.

Grad: Thank you. Inference?

Hendrix: 1979.

Grad: MCC?

Lenat: 1983.

Grad: I'm going to go through the others as well, so we have a complete picture on it. Neuron Data? Alain?

Alain Rappaport: 1985.

Grad: Syntelligence?

Fox: 1983.

Grad: Cycorp?

Lenat: 1994.

Grad: You're practically off the board.

Friedland: That's how we think of you, Doug, off the board.

Lenat: Thank you.

Allen: The first time he's ever heard that.

Grad: Aion? I think I have a date on them somewhere here. Aion was founded in 1986.

Harmon: I have all the dates of all these companies in here.

Brock: Oh, wow.

Harmon: I'm not going to try to grab it and look it up, but they're all there.

Grad: I don't have time to go into the book so we're going to do it here. 1986. Does IBM consider a product at any point in time here, or when did your work start?

Schorr: I don't remember. I was gone by the time they released products.

Grad: Well, Herb, when were you made the AI evangelist?

Grad: Unknown. Fritz? How about with Franz?

Fritz Kunze: 1984.

Grad: Red Pepper?

Zweben: That was 1992.

Grad: Blue Martini?

Zweben: That was 1998.

Grad: We did a pretty good job on that. I feel very good. All the earlier ones are in session four. Is there any reason for me not to go chronologically? Then I will. I'll see if I can figure it out here.

Grier: Could I just throw you three other dates to help sort of round the discussion?

Grier: Bayh-Dole is 1980. It assigned research rights to universities. The book that I refer to as the MYCIN book, and Ed just left, is 1983. The book said this is an expert system on this and how you do it. And *IEEE Expert* magazine is 1986. And knowing how the IEEE works probably better than anyone else here, that meant the proposal had to be in in late 1984.

AI Products, Services, and Tools

Smith: Burt, you didn't ask, but I'll tell you anyway. Schlumberger was selling AI-based products from 1983.

Grad: You were selling it as a product?

Smith: Absolutely. I can tell you what the price was. It was 50 cents a foot.

Friedland: That's a product.

Grad: Oh, that's interesting. That I consider a service.

Smith: Well, all of what Schlumberger sells is a service, but it's founded on particular technologies. Some of them are nuclear, some of them acoustic, some of them electromagnetic, and some of them AI.

Grad: Are all the companies selling an actual product or selling a service based on the product?

Hemmendinger: Product.

Friedland: Product and service.

Hendrix: Product and service.

Lenat: Product.

Grad: Were any of those companies I mentioned not selling a product at all, only a service?

Fox: Again, these are fussy categorizations.

Lenat: They're developing a product and then selling the services.

Grad: Someone pays me for solving their problem instead of giving them a tool for them to solve their problem.

Friedland: I forget, was Aion selling a product? I don't remember.

Allen: Of course. Right.

McCune: Okay, ADS didn't sell a product until 1986 or 1988.

Grad: You were selling a service before?

McCune: We were doing our research, and then in 1986, we had the topic kernel and we started selling that as a product.

Grad: In the normal software world, there's a very clear differentiation for the company that's selling a physical product—that's code, a program. Today, it's not so clear. The one who was selling was a service bureau, was a timesharing service, was selling a service, the use of a program but not the program itself.

Grier: This might be an interesting dichotomy in sort of understanding the difference between it, because Schlumberger was selling technology, was selling a service to read dip meters. Von Neumann with I guess it was then ESO, in 1956, to talk about computer rising dip meters, when they have basically no technology, no expert systems, nothing. So, this is a long history. There are a number of 1960s eras companies that were designed to help prospectors looking for oil. That may define a certain category of knowledge in which expert systems had to fit into, as compared to what MYCIN did or something else.

Grad: Look, as you can tell, I like sharp lines. The world doesn't work that way in many cases.

Grier: There may be at least categories here that we want to probe.

Hart: I think there's one additional category that's relevant. Particularly in enterprise sales, you sell the software or literally license a software product that may need extensive customization. You're selling or licensing a product, but you're also selling professional services along with that.

Grad: Professional service is different from what I'm talking about.

Hart: But it's not a pure service. Your service is a pure service business.

Grad: No, but everything went all over the place. It wasn't clean in that regard.

Harmon: In the AI world, you also have differentiation between people selling a language and a tool. That could be very fuzzy as well. Some people would say they were selling a tool, but in fact, a lot of tailoring had to be done before the tool could be used. Others were selling a tool that came complete, and others were selling a tool that came with considerable services that went with it, so you get a continuum. Gold Hill, incidentally, should be on your list.

Friedland: Yes.

Grad: There's nobody here to represent them, so they're gone.

Harmon: I mean, this is a funny way to approach this. If you wanted to get a list of all the companies or all the tool vendors, I could dig that out for you but not right now.

Grad: Be my guest. We would love to have that.

Harmon: Yes.

Grad: For today's discussion, the people here are able to talk.

Intellectual Property and the Bayh-Dole Act

Brock: May I make a clarification about the point about the change in the intellectual property landscape? Is it the case that work that was in the public domain as the result of this government-funded research was in essence privatized to the universities that were hosts to those research programs?

Friedland: To some extent in the transition period. Basically, it was decided that universities would own the IP, the software developed under federal grants, with the understanding that the universities would license it out making a sensible decision about whether to give exclusive licenses or nonexclusive licenses for the best service of the nation, in essence to make that work available to the public.

Grier: Right.

Allen: To what extent was it grandfathered?

Friedland: To be honest, I don’t remember to what extent. I do remember that at Stanford, for example, Niels Reimers who I said was really the pioneer in doing this stuff got permission for stuff under current grants, even if they'd started before the rule was made to be licensed.

McCune: Was this a new law?

Feigenbaum: This is actual law.

Hart: Yes, there's something called the Bayh-Dole Act, which was passed around 1980 that *The Economist* magazine called I think the most enlightened piece of legislation in eons. The point was that before that the attitude of the federal government was that since the public paid for it, the public owns it.

Brock: Right.

Hart: Which is to say nobody owns it.

Friedland: Right. Exactly.

Hart: The Bayh-Dole Act set a bunch of conditions of use and so forth under which universities and nonprofits could get the intellectual property rights provided they actually made use of those intellectual property rights and didn't just stockpile them. I don't know if Stanford got some special dispensation before that, but after 1980, that was a done deal and a standard procedure.

Grier: Bayh-Dole was enacted in 1980. Prior to 1980, it was not clear who owned the actual intellectual property of things coming from the NSF, from Department of Defense research contracts, and in particular, the one that forced it, the Department of Energy Labs. Some universities treated it as if it were their own. Since we did not have the internet then, they could squirrel it away and keep it away. That act did assign the license to the university. Actually, it goes to the person doing the research, but most universities then require the person to assign it to them. It also required an open-ended, no cost license to the federal government.

Friedland: Right. And that's true to today.

Grier: That's still true today. It's been updated twice in sort of minor issues, but the thing that forced it was in particular the nuclear laboratories.

Brock: Just one clarifying thing. Does everyone agree that this change in the intellectual property landscape is associated with the timing of the commercialization in these companies for expert systems?

Grad: No, by definition we have some 1978 and 1979 companies.

Fox: Any code we got from a university wasn't usable. It wasn't production code. I mean, it was fun, it was good to look at.…

Hart: Well, unless it was copyrighted or patented…

Allen: Right. This is before software patents became a thing, really.

Friedland: It was extremely important to Intelligenetics, and when we get to that company, I'll explain why.

Machine Intelligence

Grad: All right. I'm stopping any other discussion. We're now going to start the companies. I have them pretty much in chronological sequence, and because we have so many companies to cover, we'll focus specifically on three things. One, what was the problem or what was the opportunity that you saw? Number two, what was the particular technology that you implemented? Then, we'll talk briefly about your business proposition. I can ask you 30 questions about business alone, but give us a brief one if you want to explore it a little deeper.

Each one of you, we're going to have to run about 10 minutes is all. Some companies we'll go a little deeper, but let's get it at least on the record and then we go from there. Okay. So, we'll start. Number one, Machine Intelligence.

Hart: Charlie Rosen, Gary Hendrix, myself, and one or two other people from SRI's AI Center started this company in 1978 with a breathtakingly naïve business plan. Everybody here knows that, as in real estate, the three key factors for startups are focus, focus, and focus. Instead what we did was we said, “We're going to start a company whose business would be to start other new companies by the multiples of some multiple.” You can see how sophisticated we were. Gary was in charge of natural language. I was in charge of expert systems, and Earl Sacerdoti, who unfortunately is not here, was in charge of manufacturing and inspection, visual inspection.

I quickly decided that expert systems weren't quite far along enough in 1978 to actually deliver solutions, so I never did much. Gary was brilliant. I hope he will describe more what he did with respect to natural-language access to databases and how that morphed into another company of the same name with a different business model.

Earl Sacerdoti kept the visual inspection stuff going for quite a few years. Their biggest client if I recall, Gary, was a Swiss bank note manufacturer, or a printer, I should say. We were doing a visual inspection of the quality of, I don't know, Namibian bank notes, Zimbabwe bank notes, or whatever they were printing. That machine inspection part limped along for quite a few years, and I think finally it's over.

Hendrix: It's over.

Hart: Of course, you should talk about what happened to Symantec.

Grad: Does Symantec come directly out of this?

Hendrix: It did. It did. That's an interesting story how it happened. But before we get to that, the machine vision wasn't just about bank notes. The original thing was to try to get parts that had been patched together. They would be coming down on the assembly line, and we needed vision to be able to figure out which part was which, where it was, how it was oriented. We needed vision to be able to tell the robot arms how to pick them up and then put them into bins so that the parts were sorted. I don't know if you'd call that an expert system or not. It turns out that the AI was pretty similar in some ways.

Hart: Well, I will say one thing. One of the very earliest industrial robotics conferences was an industry conference with a show floor, and somebody estimated that around 80 percent of the entire industry physical production was on the show floor.

Hart: So we abandoned that. It didn't take a genius to figure out that this might not be a huge market for at least a while if we were going to do the visual inspection or the visual control piece of that market.

Grad: Okay. What specific technologies were you using? What tools did you use?

Hendrix: Well, it was the vision stuff. The way machine intelligence got put together, it wasn't like it had a technology that it was trying to do. We were going to do all of AI. It was an idea. It was a very naïve idea, but we could see all these different kinds of things coming out of the research that was going on, and as Peter said, we were so naïve we thought we would be able to start a bunch of businesses and get industry people to come in and just take the ideas and do something with them. All we had to do was just sort of define them.

Grad: That was my question. What was it you were going to do that they were going to pay for?

Hart: Well, okay, now for the two pieces of technology, the vision piece was straight from Duda and Hart. I mean, it was open. Anybody could have implemented it, and it worked as well as it worked. The natural-language piece was straight from Gary's system, LIFER, which he did the initial version of in a weekend to prove to me as his boss how wrong I was to suggest it was even possible. The next Monday morning he came in and said, "It's working."

Hart: I'll never forget that. That was the semantic grammar, and the idea was natural-language query system for database use. That was the foundation of Symantec.

<u>Initial Funding for Symantec</u>

Hendrix: We'll do the transition between Machine Intelligence and Symantec. We'd been sitting around for two years talking about how we might get something started. We were doing the vision thing. We were trying to figure out how to get into the business world. We knew absolutely nothing about business because we were just scientists, and we didn't know anything about how to look at a balance sheet or anything about marketing—just completely naïve about all that.

We had just started the company. We had hired some people who did know about business. They were trying to get the thing off the ground. We had raised just a little bit of money. I think we put up $5,000 apiece, and maybe we had $75,000 total.

One night we were having one of our regular Tuesday night meetings, and Charlie Rosen said that he had gotten this Request for Proposals from the National Science Foundation for their new Small Business Innovative Research program. It was the first or second round of that, and you could get an initial grant of $25,000. If you did well on that, you might be able to get another $200,000. Unfortunately, it hadn't come into his hands until a couple of days before, and any kind of proposal for that year had to be in Washington by 10:00 AM on Friday morning. Earl Sacerdoti and I played sick at Symantec for the next two days. We didn't sleep, we didn't do anything. He wrote the boilerplate.

Hart: At SRI.

Hendrix: SRI, yes. I get SRI and Symantec mixed up all the time anymore. I wrote the part that was the technological part about what we were going to do with semantic grammars, trying to take the technology that was being done on these DEC-10 machines and trying to put it down on something the size of the Apple II, how we were going to go about doing that. Earl, who was much more business oriented than I, wrote all the boilerplate to go around that.

FedEx was a pretty new deal back then, but we got that to the FedEx place. We got it to Washington on time, and we got that $25,000.

Grad: Which company are we talking about? Is it still Machine Intelligence now?

Hendrix: It was still Machine Intelligence. The grant started at Machine Intelligence. We got the follow-on $200,000, and it was still Machine Intelligence. But by that time, we had brought in some more sophisticated business people. They said, "You guys got to focus. We don't want this natural language stuff; we're going to concentrate on robot eyes."

So, Peter went off and started thinking more particularly about what he could do with the expert systems. With the help of Machine Intelligence and with the help of some of their business people, they helped me go and start Symantec. I started that as a completely separate business.

Grad: Now did Symantec go a different direction, or is Symantec taking that contract?

Hendrix: Symantec took that contract with National Science Foundation. We built systems with that. With the systems that we used with that, we were able to take to the venture capital community and say, "Look, we're able to do this on these small machines." They then put in many, many times as much money as the NSF had put in, and that got us launched.

Machine Intelligence Funding and Revenue

Grad: Let me close the story. Machine Intelligence, then. Did it close as a company at that point?

Hart: Much later.

Hendrix: It crippled on for 10 years.

Grad: Really. How much money did you use up in 10 years?

Hart: I have no idea.

Grad: Does anybody have an idea?

Hendrix: Over $20 million.

Grad: Machine Intelligence used up $20 million dollars?

Hendrix: Yes.

Grad: Did it ever have any revenue?

Hart: Sure.

Hendrix: Yes, it had revenue, but it had no profit.

Grad: That's a different discussion. It had a lot of losses is what you're telling me.

Hendrix: It had a lot of losses.

Grad: $20 million.

Hendrix: It was always very promising, but…

Grad: Who was running Machine Intelligence at the end?

Hendrix: Art Lash? Oh, he died.

Hart: Well, Charlie Rosen for a while. Earl Sacerdoti, I think, if not the top guy was the number two guy. Then I lost track, I don't know.

Hendrix: Well, Arthur Lash really took it up to the next level.

Grad: Did he try and sell it? Did anybody buy it?

Hendrix: By that time, I was so into Symantec I didn't have time to think about it.

Grad: Anybody know?

Hendrix: No.

Hart: No.

Grad: Okay. It's one of those things. Someday we may find out, or maybe we don't. Okay, so they threw away $20 million. Where did your money come from? Where the hell did you get $20 million from?

Hendrix: Hillmans were a big backer.

Hart: We call them investors.

McCune: This was not contract revenue or a client revenue, this was actual capital investment?

Hart: I was a founder and shareholder of a company, and you asked about the relation of Symantec and Machine Intelligence. When Symantec spun out, Machine Intelligence shareholders got some equity from that.

McCune: Yes.

Hart: I don't know what finally happened at the end.

Grad: We are done with Machine Intelligence.

Small Business Innovation Research and Venture Capital Funding

Smith: Could I ask a question? How many of you in companies you started have had SBIR funding? I mean, we do. The numbers have changed over the years, but I was just curious.

Lenat: They've quadrupled.

Smith: Quite a few people. It's 150 for phase 1 and update, 50 for phase 2 and then you're in extensions.

Lenat: Well, you can tell it's been a few years for me.

Grad: How many had SBIR funding? Hands? --- Three, four, five. Okay. Not most of you.

Kunze: Could you tell me what SBIR is?

Brock: Small Business Innovation Research.

Friedland: Every federal agency has to put aside a small percentage of its research budget, usually 1 to 2 percent, for SBIR activities. Some agencies hate them; some people consider them their venture capital funds and like them.

Grad: But no outside venture capital money for this case?

Hendrix: Oh, no, no, no. All of us had venture capital.

Friedland: Yes.

Hendrix: Machine Intelligence, that $20 million was venture capital money.

Grad: It wasn't clients putting money in to support your work.

Hendrix: No.

Hart: Well, again, there was some revenue.

Grad: Yes, but they weren't becoming investors in your company.

Hendrix: No.

Symantec Products

Grad: Next. Let me finish up with Symantec because that's an interesting story. What did you do?

Hendrix: Ultimately what we did was a shrink-wrap product. We talked about products, not like a lot of the products you guys had, "We got a product. We'll come and install it for you. We'll hold your hand, and we'll put it on your machine. We'll consult with you for six months or a year." We put stuff in a shrink-wrapped box, and we didn’t talk to our clients directly. We sold it to distributors. It's a very different thing.

Smith: Absolutely.

Hendrix: We all had sort of product and services. This was a product. This is like buying bananas at the grocery store. It's a very different deal.

Hart: They had online updates.

Hendrix: It eventually went rotten, but not too fast. The Q&A product was a little database management system and an integrated word processor. Those might not sound like they go together, but doing mail merges was a huge thing. You could do form letters and back them up with the data from the database. That turned out to be extremely important for all kinds of businesses.

Grad: In any way related to what we now have been talking about expert systems?

Hendrix: Only in that it had the artificial intelligence basis, and there was really a tremendous amount of carryover in the knowledge that you had to have to do either one of them. But it was not an expert system per se.

Grad: You were doing some analysis of the queries or doing something like that?

Hendrix: We were doing analysis of queries. It was all being done with a symbolic kind of thing. It was more along the line of what a compiler would do to take a computer language and translate it from, say, the Fortran language into the machine language. We were taking English, and we were translating that into codes similar to SQL to pull information out of the database.

Grad: Did you have a limited language they could use in the queries and that's why you could do it?

Hendrix: Well, all languages are limited, so I must say yes. But the limited language was a fairly decent subset of the English language that you would use for asking questions about databases. It was not English, but it was sufficiently broad that you could bring in just about anybody, and if their main objective was to get information out of the database and not to break the system, then they would be successful.

Grad: Did you have partners or other people who were key to the company at that point?

Hendrix: Yes. There were other people involved, particularly Gordon Eubanks and Dennis Coleman were very important in being able to help get things done on the small computer systems. I come out of an environment where we were doing everything in LISP. I thought I was making a big step down to do things in Prolog, but they were quite interested in doing stuff in binary if that's what it took in order to get things to be fast. They had no qualms about that. If we hadn't done that, we wouldn't have been able to get it shoehorned into the tiny little machines that we had back then.

Also, there were various business people that were brought in. Gordon Eubanks was key to that. He not only knew programming, but he had also done business.

Grad: But you leave at some point.

Hendrix: I did leave at some point.

Grad: And Gordon becomes the new president of the company.

Hendrix: Gordon became the president of the company early on before we had a product, and I went over and became just in charge of the artificial intelligence part of it. Gordon and the people that he brought with him were very good at knowing how to do marketing and interact with the distributors and all that kind of thing. They just didn't teach us how to do that in the Computer Science Department at the University of Texas where I went to school. That wasn't part of the curriculum.

That was extremely important. You needed to have these different types of expertise in order to form a company. It's one thing to have a technology, and it's quite another thing to have a company. It turns out that most of what we had to wrestle with was not the technological issues.

Grad: It was business.

Hendrix: It was just all this stuff about running a business.

Grad: Do you have any idea what kind of revenues you had at the time?

Hendrix: We started out with revenues in the few millions, and then it went straight up. It was quite a successful company. It still is.

Symantec's Transition into the Security Business

Grad: Symantec, though, went into the security business fundamentally, and that's where its money and its growth came from, right?

Hendrix: It transitioned into that. What funded the transition was the Q&A product to begin with. That provided the cash cow that allowed Symantec to get first into other types of software—more word processing, more computer languages. We did outline processors. We did project planning software. A lot of that was funded with revenues from Q&A.

Of course, these things themselves became profitable, and that all sort of snowballed. Over time, it became clear that the most profitable areas that we were in were utilities. Once we figured that out, we acquired Norton Utilities. We were able to triple the sales of Norton Utilities within about six months, and by that time, we'd hit the jackpot.

Grad: Symantec is quite a story.

Hendrix: That then got us into a position to go into the security business, and Symantec really took off and started going in the direction of the company that you know it as now. I see all these excited looks around the conference room. But I was not excited about that. It wasn't AI anymore. It wasn't fun for me anymore. As it transitioned into security, I transitioned back to Texas and got myself a ranch.

Grad: We have an oral history that I did of Gary many, many years ago, wasn't it? Ten years or more ago?

Hendrix: Yes.

Grad: It was great fun. All right.

Brock: May I ask a quick question? On the Q&A, on which systems did it run?

Hendrix: It was on DOS. Eventually, there was a version for the Mac. I can't remember whether we ever shipped it on the Mac or not, but I know we developed one.

Brock: Thanks.

Intelligenetics and Molecular Biology

Grad: I'd like to now get to Peter Friedland and the people who got involved Intelligenetics and IntelliCorp.

Friedland: Intelligenetics in a sense was a hybrid company. As part of the MOLGEN project, we spent a lot of time with truly world class molecular biologists—Josh Lederberg, Nobel Laureate, and other Stanford professors—trying to understand their reasoning that went into the design of experiments. The purpose of my thesis was trying to replicate the process of designing complex experiments in molecular biology.

At this time, molecular biology had suddenly gone from very little formal data to an explosion of it because of revolutions in the way of doing DNA sequencing that occurred in the 1982–1983 time period that resulted in several Nobel prizes for that technology.

Ed had been asking Josh Lederberg the story for many years, "When is your field ready?" Finally, in 1976 Josh said yes. I was lucky to be then looking for a Ph.D. topic, and I got it, so I was very lucky in that regard.

In exchange for all this time, we started writing some fairly simple programs for analyzing these strings of A, G, C, and T and in the case of proteins, a few more letters, which they found very useful. It turned out it was so useful that a side effect of doing the expert systems work was that their colleagues started asking for this software.

We ended up setting up a free account on the same computer facility that we were using for our AI research, for all of the work in AI or medicine, called GENET. That eventually had so many users that the head of it, Tom Rindfleisch, nicely asked us to go somewhere else.

At about that time, which was about 1979, we both realized that there was also the parallel growth now of companies in biotechnology. Former academic colleagues were now starting companies like CEDES, GENEX, Genentech, Biogen, Amgen, and several others. We felt that even though we could give the use of our software for free to our academic colleagues, when they transitioned to industry, they should have to pay for it, so there was a market. We felt there was a market both for that software for the AI experiment design tools and then for the broader use of the AI tools we were developing like the UNITS package, which was our knowledge representation package that had been developed there.

Founding Intelligenetics

Friedland: At the time, Ed and I actually asked many of our colleagues at Stanford, Ted Shortliffe, I think I bugged Mark Steffek and a bunch of others: "Well, is it time to form a general AI company that this would be part of?" We didn't get much interest, so the four of us—Ed and me and our two main biology collaborators at the time, Larry Kedes and Doug Brutlag—decided we would attempt to form a company.

Being Silicon Valley, we went out for venture capital and eventually raised it. Actually, it was a company that had a connection to Schlumberger. Our first venture capitalist was a company called I think Presell, which was actually not owned by the Schlumberger Company but was the private venture firm of several of the founders of Schlumberger.

Grad: How much money did you get?

Friedland: Ed, do you remember it exactly?

Feigenbaum: Gary, I have to reflect that comment of yours about scientists being naïve about business. I had this connection with this guy because they invested in the PARC computers. Out at Carnegie Mellon. We asked for $1 million, and the answer was, "Yes, but that's way too low." In the end, it was $2 million.

Friedland: Yes, that's right. We got $2 million in additional capital. We had an office. We actually convinced one of the key people on the original mass spectrography project, DENDRAL, Dennis Smith, who was a chemist and had worked for Carl Gerace. He was the main expert on that work. He was at Perle, the pharmaceutical company. He was in Upstate New York. Called up Dennis and asked him if he'd like to come back to California to be the head of the biology part of the company. His wife Ellen was an enthusiastic supporter of that.

We formed the company, and we moved into what had originally been the porno theater in Palo Alto that some of you may remember; the Paris Theater, I think it was called. That had gone out of business, but now it was a place for companies. Moved into there and did several things. First thing was the commercial market for the molecular biology software. In addition, we applied to the NIH for a true resource, a national resource to do this outreach of the whole biology community of the software called Bionet. That got approved and was funded under a five-year grant from the NIH. The biology part of the company wasn't an explosive business, but was a quite profitable business.

I brought a few historical artifacts. *Nucleic Acids Research* is one of the main journals in molecular biology, and by the early 1980s, the use of this software in general was becoming important. Besides being one of the first AI companies, Intelligenetics was really one of the first bioinformatics companies. Here are three special issues in *Nucleic Acids Research* that we had four or five of the papers in. We had papers talking about the software and software in general now being important in molecular biology.

The second part of the company was to develop a specific molecular biology workstation. Remember, there weren't a whole lot of PCs around, but there was a growing market in workstations from Sun, from Apollo, and from Quirk. Because we were scheduled to Quirk, our board member suggested we use that as a platform, so we actually had a project called the Pepper Project. That's the requirements review to build a workstation based on the Quirk to sell to people as an experiment design tool in molecular biology.

The third thing was starting to exploit the general AI field, and we brought in several people in that area, most notably Tom Taylor, who had been at Texas Instruments, to work on that.

Friedland: In the early days of the company until about 1983, the main part of the revenues was from molecular biology software. In fact, the company went public in 1983, the S1, mostly based upon people buying into the biotechnology market.

Starting in 1983, however, late 1983 and early 1984, that's when the world suddenly discovered expert systems were incredibly important. We were talking about this over lunch. Started paying people very large amounts of money to give talks about expert systems, a very large amount of money to attend tutorials about expert systems. Suddenly, the AI conferences that had been academic conferences now started having trade shows.

That culminated in 1985 with the IJCAI at UCLA, which had almost 10,000 people of whom about 8,500 were people who attended an enormous industrial-like tradeshow. Companies like IBM had these enormous booths there, up to the ceiling of booths selling wares in AI.

At that time, our expert system revenues were starting to take over explosively from the biology revenues.

Establishing IntelliCorp and Selling Off Intelligenetics

Friedland: In fact, in 1986, as Ed alluded to before, actually in 1985, we changed the company name to IntelliCorp, of which Intelligenetics was a subsidiary. Then in 1986, Amoco, which was Standard Oil of Indiana, bought a 60 percent share in Intelligenetics.

Grad: For how much money?

Feigenbaum: That was $4 million.

Grad: $4 million. All right, so you valued the company around $6 million.

Friedland: Ed's my money guy. We bought it off and actually moved it a few blocks down El Camino in Palo Alto. That lasted there for a while. It's always nice to see the companies you founded still have names. Now that space is taken over by Social Security. That's where I went to get my Social Security card today.

Grad: Now you were doing this when it became IntelliCorp. The expert revenue, what were they paying for?

Friedland: People were paying for several things. They were paying for customized expert systems themselves. For example, a famous one I got involved in was the 3M adhesives advisor. There was another thing that was being done for some of the airlines for scheduling systems and so on.

IntelliCorp Products and Services

Grad: We'll talk about these applications tomorrow, but you were selling the service?

Friedland: We were selling that. We were selling a tool.

Grier: Could you just clarify that for a moment? You're talking about customized offerings. Is it basically just customized rules of knowledge behind it? Or are you actually modifying the software on top of it?

Friedland: Both. Both. There was a tool at the time we were selling quite a lot of called KEY, Knowledge Engineering Environment. It was actually originally going to be based on the UNITS package, but then Tom and his lead engineer who Gary—I forget his last name—brought over from Texas Instruments, designed the separate platform. That became KEY, the Knowledge Engineering Environment. We were also selling training on the use of it.

Grad: Was that an actual software package?

Friedland: Yes. Actual software package, both KEY.

Brock: Was it shrink-wrapped?

Feigenbaum: No. Not shrink-wrapped. It was a product, but it wasn't shrink-wrapped.

Friedland: Well, it was shrink-wrapped in a sense.

Grad: No, David's question was, did it sell as a unit, or did you have to modify it to make it usable? Was it usable?

Friedland: Yes. But like any large software project, whether it's from SAP or Oracle or anywhere else, if you're a sensible person, you buy advisory consulting time to help modify it to your own environment. So, yes, it was shrink-wrapped, but it would be a nonintelligent customer who bought only the package.

Grad: Was most of the money then form the add-ons, or was it from the basic package?

Friedland: It varied. Early on, it was mostly from the basic package, although as the years went on, the consulting and service revenue eclipsed the package selling.

<u>IntelliCorp Revenues and Funding</u>

Grad: What kind of revenues were you getting?

Friedland: Okay. I have the revenue numbers actually here. Revenues in 1984 were a little over $2 million. That jumped to about $9 million in 1985. I think it peaked at about $20 million a couple years later.

Grad: When did you sell to Amoco?

Friedland: Well, Amoco was only sold the Intelligenetics part.

Grad: But you kept the rest of the business. That's what I was confused about. The pricing of what they paid didn't make much sense.

Friedland: Yes. The pricing they paid was for the molecular biology software part of the company.

Feigenbaum: Burt, they bought the biology.

Grad: I understand now.

Feigenbaum: We wanted to be expert systems.

Grad: That's fine. I understand.

Friedland: Well, I was actually involved with both companies still. My wife's a molecular biologist, and she was involved with the biology part at one point while she was pregnant and didn't want to be around a laboratory with harmful chemicals. One other thing that's very interesting, I think, is this is actually the corporate guide to presenting at the 1985 IJCAI. This was the training manual for all the corporate employees who were going to be talking at the IJCAI at the booth.

Grad: They had to read all those pages?

Friedland: Yes. It was this was sort of standard. I mean, this is the standard thing for any company, whether they're exhibiting at the Gartner ITXpo, which is the world's biggest IT conference, or at IJCAI at the time. That's the training manual for all the sales folks and technical support folks who are going to be manning the booths or wandering the floor at IJCAI.

I have to tell you, it's actually quite interesting; that led to a second offering of the company. The first public offering we had in 1983 was through Bladensburg Founden, a good but second-tier Wall Street firm. The second offering, reflecting the heyday of expert systems, was through Morgan Stanley and Rothschild, Unterberg, Towbin, an absolute first-tier firm.

Grad: How much money?

Friedland: We raised a little over $22 million in the second offering.

Grad: What percentage of the company?

Friedland: A good question. Let me see. I think it was about 25 percent. I don't have that in my figures yet.

Grad: That's okay, but that's a fairly typical percentage for a minority position.

Friedland: Yes. What's interesting is that for the first S1 in 1983—just to show you how rapidly the transition was and the growth of expert systems in exactly that 1983–1986 period—we're talking almost entirely about the microbiology side of the company. This one talks almost exclusively about expert systems rules and so on.

Grad: How big was the first one, about $2 million?

Friedland: No, no, no. The first offering raised about $7 million.

Grad: Well, that's pretty big then.

Friedland: Yes.

Grad: Then you sold the company.

Friedland: We sold the Intelligenetics part of the company.

Grad: Yes, but that was how you transitioned. You said it was now an expert systems company.

Friedland: It had always been also an expert systems company, but that transitioned from being a minor part of the revenues to by far the major part of the revenues in that time period. It was actually quite interesting as a founder of the company to watch that transition go on.

Grad: By 1986, your company was basically valued at around $80 million.

Friedland: At its heyday, yes.

Grad: Yes, that's what I was thinking. That's a quite a nice growth. Okay, what happens to IntelliCorp?

Friedland: I went to NASA in 1987. I didn't lose my connection with the company. I was still a significant stockholder and so on. But as a federal employee, I was no longer working for them. The company sold expert systems and grew actually slowly after this huge birth until about 1992. Then we hit the downturn in people buying expert system products and the like.

Like most of those companies that lasted a while longer, transitioned from being a product-selling company to mainly a service-providing company. In fact, that whole history is detailed in Jeff Moore's books, *Crossing the Chasm* and *Into the Tornado* and the like, what happened to these companies to make that transition.

Finally, it went out of business.

Feigenbaum: Yes, it was the tech bust in 2002, I think.

Friedland: I think it was like 2001–2002.

Grad: It lasted that long.

Friedland: It lasted that long.

Grad: Okay. I'm going to cut here. I want to do one more before we do a break.

Feigenbaum: Two things I wanted to add to what Peter said. There was an event that Peter didn't mention that was in my view quite significant that basically led to the spinoff of Intelligenetics. Remember Peter said that the NIH through the NLM gave us a whole bunch of money to make this software available to the nation. In 1984, I think it was, they decided to do it themselves. Don Lindberg, who was head of the National Library of Medicine, was also Chairman of the AI Medicine National Group, decided that that libraries needed to change and the National Library of Medicine wouldn't just be a paper library or an electronic library, it would also be a gene-based library. They set up a biotechnology information center.

Friedland: Yes. Although that was actually in 1989. This is actually the final annual report of Bionet at Intelligenetics, and that was through 1989.

Feigenbaum: Anyway, we didn't get that government money anymore.

Friedland: By that time, the spinoff company was managing Bionet. What Ed said is absolutely true. The federal government decided that instead of having a private company manage it, it would do it.

Grad: They would do it themselves.

Feigenbaum: The other thing I wanted to mention is a good segue from Gary and to Peter's talk, and it will also apply to technology. There's something in Gary's story which you didn't hear much in Peter's story, which is that when the company is loaded with cash, like Intelligenetics was from these large public offerings, it's a great time to go off and do something else. When you're a parent who's an expert system specialist and you have other expert system specialists and AI people and biologists around, they're not about to go off and buy a security company or something like that. It just didn't even enter our mind. That's bad because there was a lot of money —and there are other gardens in which you could plant seeds and we didn't.

Grad: But you might not like what the seeds grow up to be. They may grow up to be peppers instead of peas or something.

Feigenbaum: Oh, Red Pepper Software.

Grad: That's what I was thinking about, exactly.

Founding and Staffing Inference

Grad: Anyway, one more and then we take a break. I want to talk about Inference.

Allen: Okay. I'm telling the story of this is as somebody who came into the company some years after it started.

Grad: But you can go back.

Allen: A lot of detail will be missing in the early days. Basically, the company was started by two people: Chuck Williams, who came out of USC [University of Southern California] at the Information Sciences Institute at the time, and Alex Jacobson, who had spent time at Hughes Research. At the time he got involved with Chuck as a cofounder he was consulting for CEC at the time. They saw an opportunity to take the kinds of technologies that Chuck had been exploring in the context of ISI and apply that to building natural-language understanding systems. They did a number of deals which are really kind of interesting in a number of different ways.

Somewhere in there they got hooked up with Steve Wolfram at Caltech [California Institute of Technology] and got the commercial rights for the distribution of a program called SMP for the Symbolic Math Program. It was an early kind of thing like Maximo and so forth that came out of Wolfram's work. That was a sideline business that they had for a period of time, but a lot of that effort was really focused on generating revenue to help birth the system that Chuck was trying to build.

Grad: Which was?

Allen: It ultimately became known as the Automated Reasoning Tool or ART. The work between 1979, 1980, 1982 was Chuck really trying to figure out how to bring that up. One of the things they identified as a huge issue for them was really having an effective computing platform for doing that work in development. Alex at the time managed to get hooked up with Richard Greenblatt at MIT who was taking a look at some of the very early work on building the early LISP machines. He was there at a point where he was able to get involved with Rich and really essentially help found LISP Machines, Inc. at the time to the end of building a platform that Chuck could then use in essence to build the system which became ART. Chuck was working very hard during this time, obviously, as kind of a one-man show.

Grad: He didn't have any outside funding or anything at that point to your knowledge?

Allen: There was some SBIR funding that came through the work on SMP, I believe. You know, they were working off of some funds, some initial funding out of CDC at some point at one point in time. But in 1983, they got hooked up with Venrock and Tony's son, who was the partner there and became an investor in the company. That was kind of the first blush of investment.

All of this was happening concurrently with the evolution of Teknowledge and Intelligenetics and sort of that first blush of real deep venture investment in expert system companies.

Grad: Understand, okay.

Allen: Chuck had this opportunity. They had just raised a round and his challenge was finding people to come in. One of the things that distinguished Inference from the other companies of that era, this first period, was there was not nearly as close a relationship with a university like Stanford, where there was a rich ecosystem of people who could kind of come in and out and kind of get things going. Chuck was trying to figure out how to deal with that. He was kind of at wit's end, but he went out and through agency headhunters was able to find some folks, including myself, who at the time as I mentioned earlier, was working for Mark at the Intelligent Systems Laboratory on some of the knowledge-based scheduling stuff.

Fox: I'm still pissed off that you left.

Allen: That's right.

Grad: If you'd offered him more money, he would have stayed.

Allen: I know. They never think of it. They did something really that I guess has become canonical in this day and age but was kind of a thing of the time. They were starting Carnegie Group at the time, and I actually took an offer form Carnegie Group for about a week until

Chuck came back in an act of utter desperation. He had this horrible flu, and he was at wit's end and was talking to the headhunter about, “What are we going to do?” They said, "Just make him an offer he can't refuse." That combined with the fact that I was looking at the difference between Los Angeles and Pittsburgh, which is a theme I've heard a couple of times here today, I said, "I'll come. I'll come work for you." A couple of other individuals came very, very quickly with me: Mark Wright, who was also at the Intelligent Systems Laboratory, and Paul Haley.

All of us within the CMU ecosystem had become pretty steeped in this body of expertise that grew up out of the work on R1 and Xcon, the work that was being done in Mark's group in terms of expert systems applications and very deeply informed by the work that Lanny Forgy had done on the OPS system and the various kinds of production with these reasoning systems at the time.

Inference’s ART Product

Grad: What is the product? What is the service you were then offering?

Allen: The product was ART.

Grad: What did it do?

Allen: Myself and Mark and Paul Haley and Chuck came together and built the system, starting in the spring of 1984. It was an expert system shell in the parlance of the times.

Grad: Which meant?

Allen: It was an environment. There was a programming environment on the LISP machine with the various kinds of rule-based expert systems engines, with the ability to represent information in that fashion, ability to represent information in terms of what we called schema, so at the time which was frame-based knowledge representation, have that interact. Chuck had done something really brilliant early on which was what back in the day used to be known as an assumption-based truth maintenance system, which allowed us to be able to do hypothetical reasoning and a lot of really interesting stuff.

Friedland: They were our bitter rival. He and I were duking it out for that.

Grad: Oh, was that where there were two that were competing?

Friedland: Well, there were others.

Allen: But it was Teknowledge, IntelliCorp—Intelligenetics at the time—Inference, and Carnegie Group.

Grad: That was the gang of four?

Friedland: Yes, right. Absolutely.

Allen: It was selling that tool, running on LISP machines with services to help people get started, to get the training.

Grad: Where was the bulk of your money coming from?

Allen: From sales, from software license sales.

Grad: License, not from the service.

Allen: I'm not sure how that would break down, but I would say it was probably something on the order of maybe 60/40.

Grad: I haven't asked this question before, but in the other software world that I was in if you broke even on the initial sale you were damn lucky. You made your money on the follow on or the maintenance, whatever you called it, for your follow-on money.

Allen: Yes.

Grad: Did you have the same kind of model?

Allen: No. I think that it was basically a one-time software sale with maintenance.

Grad: But the maintenance wasn't that significant in terms of your revenue stream?

Allen: Yes, it was in the normal course of traditional enterprise software sales.

But I think the thing that was significant was we started building the thing in the spring. We launched it at AAAI in Austin in 1984. That was sort of the commercial launch, and within the next few months, we were doing about $1 million a month in revenue, just on the basis of the sale of the tool.

Grad: I'm sorry, a million a month?

Allen: A million a month. That was something that opened some people's eyes. We'd got an investment from Ford Motor Company as a result of that initial interest and then started building expert systems in this fashion. You know, part of it was sales to the federal government off of the whole body of interest generated out of the fifth-generation computing project.

Grad: How high did the revenue go?

Allen: In the longer run, the company got to between $20 and $30 million.

Inference's RIM and CBR Express

Grad: What was the end result?

Allen: The end result was there were two significant pivots, if you bear with me for a moment. One was about 1988 or so. People had begun to tire of trying to figure out how to get the LISP machines to talk to mainframes and the expense involved with LISP machines and the expense involved in rule-based knowledge engine. They were looking for alternatives. One of our customers, NASA, at the time, made a decision and said, "We don't want to spend any more money on this tool, but we find it every useful, so we will basically reverse engineer it from the point of view of the syntax and the operation of the inference engine and build a thing," which they called CLIPS, which they open sourced and brought out on the market. Other people would have had a different response, but what we did at Inference was basically say, "Okay, well, good for you. That's a great thing."

Grad: You didn't sue them?

Allen: No, we didn't do anything.

Grad: You're not Americans.

Friedland: That wasn't our group at NASA; that was the Johnson Space Center. That wasn't Ames; that was Johnson.

Allen: We then took that code base and brought it back in and built a new product called RIM, which was an expert systems tool but something we'd run on PCs, on Unix workstations—you know, get out of the LISP stations.

Grad: You changed your platform.

Allen: We changed the platform. That was fine for a while, but we started to get into the late 1980s, early 1990s and clearly the business for doing expert systems was softening. Part of what I looked at in the context that we were in there was, what could we do to make the problem of building expert systems simpler? That's when we started to take a look at the notion of case-based reasoning and simply accumulating cases as the knowledge representation and the knowledge base essentially to support decision making in an area which was focused on customer support. The idea was initial telephone support: how do you get somebody to respond effectively?

That turned into another product called CBR Express, which was essentially a specialized expert system shell for building those kinds of case-based reasoning systems.

What happened at that point was all of a sudden that business started to take off. The original expert system shell business lingered. Chuck was devoted to making that happen. Ended up splitting the company into Inference, which followed the CBR Express path and eventually became a customer relationship management company, which exited in the sale of the EGain in 2000. The other half, Brightware, kind of went along and sort of it may even be there in some instantiations still.

Grad: Which one did you stay with?

Allen: The CBR Express was my baby, so I kind of went with that.

Friedland: Did you go public?

Allen: We went public in I want to say 1996.

Friedland: Yes, I thought it was pretty late.

Allen: 1994–1996.

Friedland: Yes.

Allen: That was successful. At that time, I had left, Inference did about $20 to $30 million a year.

Grad: Pretty consistent.

Allen: Basically, as an early part of the customer relationship management.

Hart: I know exactly what your contracts were, but enterprise software sales in that era typically had about 15 percent a year maintenance fee.

Allen: Yes.

Hart: That would be separate from customization, consulting, and so forth.

Grad: Yes, because this was a typical software product. The ones I know you would make your sale, but by the time you paid the sales people and the number of places you bid you didn't get, if you broke even on your initial sale you were very happy. You made your money on that 15 percent. Eventually, it got up to about 20 percent for most of the software companies. Of course, you were raising that every year because it was more complicated or whatever.

END OF THE INTERVIEW