AI: Expert Systems Pioneer Meeting
Session 5: Business Reviews of Later Companies

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
Denny Brown	Teknowledge
Ed Feigenbaum	Intelligenetics/Intellicorp; Teknowledge
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
Alain Rapaport	Neuron Data
Herb Schorr	IBM/ISI
Reid Smith	Schlumberger
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
Paul McJones	CHM, Special projects
Len Shustek	CHM, Chairman

Ed LaHay SI SIG, Meeting Project Manager

AI and Expert Systems Terminology

Burton Grad: We are now starting Session 5, which is to discuss the other companies we didn't get to yesterday. We're basically an hour and a half behind schedule, which I consider no problem at all. We will cut down on the applications time, which I thought was probably the least significant. There are books that have a lot of these applications described.

I do want to thank all of you. I found yesterday absolutely fascinating. You do have a language that's a little different from mine; it's academic. The words you use have a very precise meaning to you, and they mean something totally different to me. For example, I know what a declarative sentence is, but that's not what a declarative thing is here. Imperative, I know what that means, but you have a different meaning than I have.

Not all the historians are here, but David, you can represent the historians.

David Grier: You think that's a good idea?

Grad: Well, you're a well-educated historian like most of them are. You were there for part of yesterday but not all, but I'll also ask Colin who knows the subject area, or David Brock who knows certainly more than I do, whether the terminology and the words were clear to you regarding the meaning of expert systems and things like that and whether it's just me who's a slow learner.

Grier: They're clear to me, but also remember I'm not really the historian; I'm an engineer that has wandered around the world. What I know is happening in the historical community, particularly the one dealing with AI, is a tremendous debate that is not going to be resolved today, tomorrow, or any time soon over what these things are, what categories holds the ideas, and how they relate to each other. That's what interests historians, and that's what they get all excited about arguing about.

Grad: Is what we're doing here? Trying to get these words clarified further, is that useful or is that a waste of time?

Grier: I think there are limits to its value, but what we've done so far has been fine. The best comparison is the Smithsonian in the 1970s when it realized that the first generation of hardware engineers was passing from the scene. It rushed to get ideas from them and set in stone a whole bunch of concepts because they were the only ones doing it. When you use those materials, you translate things out of it. They were responding in many ways to the trial on the patent on the computer in 1972, so they tend to have a bunch of legal categories that are very rigid and don't really illuminate what was going on there.

I think that's what will come of this. People will see that we were consistent; we were trying to understand as best we could, but historians like to make these arguments about what terms mean and how they relate to each other.

Grad: See this is what had worried me. In a different context, a different time, the word has a different meaning. What can we do now to clarify that for someone who looks at it later, that it's the meaning you intended, it's what you meant it to be. For example, when someone sees a simple word like declarative 50 years from now. Denny was trying to explain to me what the difference between declarative and an imperative sentence was. I'm not going to spend a lot of time, but Colin, do you have a thought on that?

Colin Garvey: I think it's definitely worthwhile to ask a room like this what, for instance, expert systems meant, so that conversation yesterday is really important to have especially with this group of people because it was a unique term that was chosen for a good reason.

Grad: Do you think we've accomplished our purpose with that yesterday, or do we need to spend more time on it?

Garvey: I'd like to hear a little more myself, but as David said, I'm a historian, and we like this kind of thing. Some of the low-level terminology about programming may be less relevant to the conversation. But I think the really key terms like expert systems might be worth revisiting. For instance, when terms like expert systems and AI stopped being used in the 1990s, why did they stop being used? When did you stop calling your company an AI company? Those are questions I'm very interested in.

Hansen Hsu: One thing that I noticed that seemed to be sort of assumed and glossed over in yesterday's discussion was maybe the practical or pragmatic term that expert systems may have represented in AI. I think that was for me manifested in the fact that we didn't have a discussion about the meaning of the word intelligence, and there was no discussion over whether or not expert systems constituted a way of understanding human intelligence, which is maybe what earlier AI, general AI, was trying to do. I don't know if that's worth our time today.

Grad: We will have some time this afternoon. After we talk about why all you guys didn't succeed, we can get to those kinds of subjects. I do ask the historians to help guide on this because remember, we're collecting this information for historical purposes, not just to give you a chance to brag about what you did and we'll go from there.

CyCorp

Grad: Okay, we are going to now cover the other six companies we didn't cover. Doug, since you were on a roll yesterday with MCC, why did you go to CyCorp? Why did you leave MCC?

Doug Lenat: As MCC's funding began to decline, they made one of the canonical sorts of mistakes that shrinking companies make, which is they cut technical people but didn't cut nontechnical people. When they had 450 people, it was reasonable to have a 10-person purchasing department, but when they had 100 technical people having a 10-person purchasing department made less sense and so on. As a result, the overhead rate at MCC kept going up and up and up. As they went out to try to get government contracts, but it just became an anchor dragging down that effort.

One by one, the few most successful projects including ours got our sponsors behind us and essentially left the mother ship and formed spinout companies, which I suppose exacerbated MCC's problem.

But now we're on my ship, so let's continue the story from my point of view. So, we formed CyCorp. MCC owned five percent of CyCorp, but we owned the full rights to the IP [intellectual property], the technology that had been developing for the previous decade.

Grad: Who's we?

Lenat: CyCorp. I owned about 91 percent, maybe 90 percent of CyCorp.

Grad: It was your company.

Lenat: Yes, which I still pretty much do. The other employees owned about five percent, Now the other employees own about 10 percent of the company. We haven't gone out of our way to get investment dollars or gone public or anything like that. We supported ourselves through initially the first couple of years. The sponsors who had supported us at MCC like Apple and Microsoft continued to support us, but as we wanted to grow, we had to go out and get government contracts. We went and got a large contract from Dave Gunning, who was at DARPA at the time and is again at DARPA.

Basically, it was doing a high-performance knowledge base project, HPKB. We became part of that project and, thanks to doing well there, got folded into a series of ensuing DARPA projects and intelligence agency projects that eventually became ARDA [*Advanced Research and Development Activity*] and then IARPA [*Intelligence Advanced Research Projects Activity*].

CyCorp Revenue and Contracts

Lenat: From about 1996 or so up to about 2012, maybe 2011, that's where most of our funding came from. We had gotten I suppose a grand total of, depending on how you count, about $40 or $50 million of funding to build the Cyc system, which for this purpose you can think of as a large expert system during the MCC era. Then in the past 24 and a half years, we've gotten about $150 million of additional revenue. We've spent about $200 million to build up Cyc to where it is today.

Grad: Excuse me. You mentioned about $50 million, but you still had 90 percent of the company.

Lenat: Yes.

Grad: How the hell did you get money without giving them some ownership?

Lenat: Because I'm extremely adorable.

Grad: Do all of you agree with that?

Lenat: Or I was back then.

Friedland: I think what he's saying is they didn't get $50 million in investment; they got $50 million to execute government work.

Lenat: Okay, MCC did not do government work until about the time that we left.

Friedland: What I'm saying is they didn't buy stock with the $50 million; they bought work.

Lenat: Yes, not government work, but other services. Yes, all along during those 10 years, 1984 to 1994, we delivered technology. We delivered the growing Cyc knowledge base to all of the member companies, and they sent what were called assignees to MCC who would spend 6 or 12 or 18 months at MCC and then rotate back into their member companies. Part of what the companies were getting was this sort of apprenticeship and training.

Grad: They were paying for services.

Lenat: That's probably how they thought of it. Yes.

Grad: What did you think of it?

Lenat: We thought of it as this is a solid guaranteed funding source that will be a platform for us for the coming decade, that we can securely do things but don't have an immediate application, which the other expert system builders and especially companies who were building expert systems had to face: How can we do something that will be practically saleable?

I felt the expert systems community kept running up against the bottleneck I mentioned yesterday, which was the lack of general knowledge, the lack of common-sense knowledge. I'm not talking about factual knowledge. I'm not talking about what today you would find on Wikipedia or access via Google, but

almost a complement to that, the things that everyone who writes articles assumes the reader already knows about the world. Like since you're wearing that shirt, you probably own that shirt, and since you own that shirt, you probably own that button of that shirt because ownership transfers through physical parts.

Grad: Is that what was meant yesterday about tacit understanding?

Lenat: Yes.

Grad: Yes, there's so much we take for granted.

Lenat: Yes, prior and tacit knowledge. We actually got a group together at Stanford in 1983 to help me estimate how big an effort this would be. People did this estimate in different ways. Allen Newell did an estimate based on how long it takes to burn concepts into long-term memory: how many things could you learn by the time you were about 10 years old? Marvin Minsky did an estimate based on the number of words in a dictionary, and Alan Kay did something based on encyclopedia articles.

Fortunately, they came up with estimates that were all about the same, which was, "Oh, we might need like a million rules." The fortunate thing was that while they were off by an order of magnitude, they were all off in exactly the same direction, and if we understood I think exactly how large this effort would have been, we might not have started it. But we thought that, yes, we can get this done in less than 10 years.

CyCorp Strategic Plan

Grad: Tell me what CyCorp was supposed to do.

Lenat: CyCorp was supposed to carry on that effort to actually continue building the tens of millions of general rules like that which we now have, about 25 hand-entered general rules about a small number, about one and a half million terms in what in the 1980s we dubbed an ontology to try to distinguish that from a taxonomy. It's all well and good to have taxonomized terms with generalizations and specializations on relationships, like sleep, night, bed, person, home, and so on. But in order to say that you actually understand something, you really want to have the rules of thumb that people have about that—like people sleep at night, they sleep with their eyes closed, they sleep lying down, they can be woken up, they don't like being woken up—things that you know about the world and have known since you were a small kid.

Basically, what we did was to tack with the funding winds for that next 15-year period getting projects that had some nontrivial, hopefully 50 percent or more projection in the direction we wanted to go anyways, so we didn't have to take investment dollars, we didn't have to borrow money, and so on, but we were able to make progress.

Then starting about 2012, most of our money and now 100 percent of our money came from large companies that actually were using the technology. They license the technology. There's usually a period of 6 to 18 months at the beginning where we do some of the customization and development that several people have talked about yesterday, but we don't want to make service dollars. We don't want to be a service company, so as rapidly as possible, we want to get an application that they will just license and use. Even better, we want to have a few of their people up to speed to use our technology so that from then on, they can build the applications.

We're modeling the enterprises as we go along. The *n* + 1st application for a particular large company goes much faster than the first application did because we've already modeled a lot of their enterprise. We've essentially semantically unmapped our ontology to their database schemas, for example, and we've modeled on their policies and procedures and so on.

Grad: Can you name some of your clients? Is that appropriate?

Lenat: Yes, but the more successful ones want to keep things quiet.

Grad: Okay. Then don't. That's fine. Do you have some that have failed?

Lenat: There's some customers that are perfectly happy to have us reveal their identity like the Cleveland Clinic, Accenture, and a few others.

Grad: That's impressive.

Lenat: In the last couple of years, maybe two and a half years, we've tried to focus more on three verticals, not because Cyc is inherently oriented toward or should be oriented toward them, but it's more for the Willie Sutton reason: "Willie, why do rob banks?" "It's because that's where the money is." We're focusing on healthcare and financial services and energy customers, but we also have applications that are completely unrelated to that like helping sixth-grade students learn math and science better and things like that.

CyCorp Hiring Strategy

Grad: How many people do you have working with you?

Lenat: We have 65 people, almost all of whom have their PhDs. Most of them are PhDs in philosophy because it turned out to be more economical and frankly easier to teach philosophers what they need to know in order to effectively program in Cyc's logical language than it was to hire and retain computer science graduates. We have extremely low turnover, typically on the order of like one percent or two percent. One person every year or two will leave. Most of our people have been with us for well over a decade.

Grad: Are these employees, or are they loaned to you from the participants?

Lenat: No, they're all employees.

Grad: That's a big operation.

Peter Hart: Well, what we all want to know is how many purchasing agents?

Lenat: We get by with less than 10 full-time purchasing agents, but it's a struggle. We have a small number of what you might call administrative people, but we try to keep that at the level of like two or three such people if we can.

Unidentified Person: Were philosophers also cheaper to hire?

Lenat: That's why I said more economical; I was trying to avoid saying cheaper. But I would say not just to hire, I would say to retain as well. For all intents and purposes, the lessons we learned at MCC and in the early years of Cyc have helped with that.

One lesson was you need an expressive representation, something as expressive as English if you're really going to be serious about capturing what subject-matter experts are telling you. The rules that they want to tell you don't translate well into three-word sentences, don't translate well into associative triples, don't even translate well if you put on a straitjacket at the level say of MYCIN rules and so on. Sooner or later you really do want quantifiers, nested quantifiers. You really do want negation; you really do want modals like believes, intends, expects, dreads, and so on; you want nested modals. You want basically the full expressive power of English. That forces you, and it forced me kicking and screaming, from something very much like the MOLGEN Units Package, which is where we started into something which the philosophers actually get trained on, which is higher-order logic.

Cyc Reasoning and Knowledge Base

Lenat: Now the problem with that is if you're not careful then the average problem, even if you had 1,000 rules let along 25 million, would take until roughly the heat death of the universe to get like 100-step proof, let alone a 400 or 500-step proof. We had to come up with ways to speed things up, and the first big step we took was to give up global consistency.

That may sound scary, but you intellectually know that the surface of the Earth is globally spherically, but you live your life as though it were flat, which works because it's locally flat. In much the same way, the Cyc knowledge base is sort of a 12-dimensional kind of structure of regions being context or micro theories, and things are more or less consistent within one of those. The further away you get, the less and less consistent things are, until finally you're in fictional things or one dimension in time things that were true 5 years ago or 55 years ago but are not true anymore.

In general, Cyc reasons by kind of spreading activation so everything is resource limited. After a while, you come up with answers. When you get a contradiction, you just sort of give up in that direction and go in other directions. You get a series of answers, and you have not really proofs but arguments, so you have pro and con arguments behind the reasonable set of answers to any question that you ask. Obviously if I asked, for example, "Was Bill Clinton a good president?" it's not like there's an objective answer. There are two answers, and there's a set of arguments for each of those two answers and so on.

That's one reason why this was used a lot by intelligence agencies: in intelligence analysis, and in a lot of business analysis as well, people are confused by confirmation bias. A person comes up with an answer, and it's very hard to get off of that answer and a plausible story behind it and look at the second and the third and the fourth most likely answers. One approach would be to have a whole team of analysts independently looking at a problem, but Cyc can do that, and really if you think about it, all expert systems could do that more or less automatically.

Grad: Hold it a second. We have a couple of questions.

Lenat: Let me just close by explaining the main way that we sped the system up by orders of magnitude. By orders of magnitude, I mean by seven or eight orders of magnitude. The main insight was separating the epistemological problem of what the system should know, which is these nice 25 million beautiful high-order logic rules. Separate that from the heuristic problem of how the system reasons efficiently.

To do that, we have this sort of community of agents that are little inference engine experts that do various things, some of which are domain-independent like caching the transitive closure of transitive relations, some of which are domain-specific like balancing chemical equations, and things like that. At any moment, there are about 1,100 of those that are active and paying attention to what you can think of it as a multidimensional blackboard structure: whenever anyone can do anything, they'd raise their hand, and the one with the best track record gets called on and so on.

In the very back of the room is this one guy who is just the general theorem prover who always has his hand up. You never call on him unless you absolutely have to because he's so goddamned slow. In fact, one of the things that we learned a few years ago that startled me and startled us was that if you ever call on him the chances are that you're not going to have enough time to wait for him to give you an answer, so we quietly turned the general theorem prover off. The only inference engine that works in Cyc now is this body of 1,200 specialized inference engines that were supposed to just occasionally come in and do something.

I'll stop at that point, but I have 143 other stories like that.

Accenture's ConnectBot Application

Grad: Paul, you had a question.

Paul Harmon: You started by talking about general knowledge and then you mentioned with I guess Accenture, rules and procedures.

Lenat: Oh, I'm sorry. The Accenture application is something called ConnectBot. For those of you who went to SXSW [South by Southwest] this year, the SXSW opening party was hosted by Accenture. They rented out this large multistory restaurant. The main application that they showcased was something called ConnectBot, which is a Cyc application. So, you go in and you chat with it for a minute or two about where you came from, what you're interested in, who you're traveling with if anybody, and so on. It would make recommendations of which of about 700 things to see or do in Austin that you ought to consider doing, and then you'd see the reasons behind them, the line of reasoning. If you disagreed with any one of them, which sometimes people did, you'd cross that out, and it would quickly do some truth maintenance, as Reid was talking about, and get a new set of justifications.

Very often, if you were traveling in a group the recommendation would be things your group should do this even though it's not at the top of anybody's list and here's why your group will be the happiest if you do that one. That was just a particular Accenture application. There are other applications in the works for Accenture.

Singularity versus Human-Machine Cooperation

Peter Friedland: I'm going to use a word that I usually really, really hate when people like Kurzwell say it: singularity. When you first started Cyc, when you first started the effort, because it originally was part of MCC, I heard you talk and others talk that at some point there would be enough general knowledge that a system would be able to perform at some level much closer to humans because it knew enough about the world at large.

Now I'm not particularly one of those who's scared about our AI overlords, except that Ed's my AI overload. Do you still believe, or did you ever believe that at some point there'd be a tipping point where enough knowledge was encoded that interesting things would happen?

Lenat: That's a good question, but I think the answer is “No.” Long before that would happen, what you'd have, and what we're seeing already everywhere, is a combination of humans using software. Together, they are smarter than they would have been, able to solve harder problems, do more in parallel, misunderstand each other less, and so on. As each individual human gets smarter, humanity gets smarter and then something interesting may happen. It's hard to predict what will happen then, but it's not singularity of the machines taking over. It's just the machines amplifying…

Friedland: I'm not saying taking over. I was talking to Pat Winston about this: from the point of view of storytelling, a system would need to know to start to exhibit at least somewhat dramatically different

behavior. I'm not wondering about taking over; I'm more worried about is there a tipping point where there's enough knowledge that in a sense it knows when to ask for more knowledge the way a human would.

Lenat: Oh, that's actually the stage we're at now. That's why I think we're in the final few years of the Cyc and Cycorp effort in that sense. I believe in less than five years we'll cross the point. We're really at the point that you're talking about where most of our people now are not sitting quietly in their monastic cells illuminating new rules. Most of them are in fact helping Cyc to understand natural language, understand what it's reading out on the web, helping it to contextualize. —I know this will come as a shock, but you don't want to take what you read on the web literally all the time.

Friedland: It's all fair and balanced.

Lenat: A large part of the task is to understand the correct context in which this was set.

Friedland: You're close to that you think now.

Lenat: Yes.

Friedland: I think that's an important point for you guys to get.

Grad: Absolutely. I think Doug could talk to us for the rest of the day. I'm going to be a little selfish and give a few of you other guys a chance to talk. Doug, it's incredible—this is a totally different level I gather than the other projects that are here, and he's lasted for what, 30 years?

Lenat: Thirty-four and a half, but who's counting?

Grad: I had one final question. What happens if something happens to you, you get run over by a bus?

Lenat: As I said, we have a set of really, really competent individuals. I didn't always feel this way, but I believe we're at a stage now where I could retire or I could leave. I'm not going to, but I think that we've passed the point of achieving orbit in that sense. In fact, a lot of the commercial customers we have in the three vertical areas I mentioned, most generally don't really deal with me anyway. They deal with our business people.

Grad: Do they have the vision that you have? That was what I was asking.

Lenat: Yes, they have. I would say they have the momentum that it would take a while to start losing altitude and burn up in the atmosphere.

Introspection and Inference

Brock: I had a question, and then Tom has a question before we move on. Doug, I think you just alluded to it with your joke about the monastic cells. Was it the case that your philosophers were using introspection to build the set of 25 million rules?

Lenat: Absolutely. Look at any piece of text: could be an advertisement, could be a piece of a Shakespearean novel, could be an encyclopedia article, it doesn't matter. Whatever it is, try and introspect on the white space, not the black space. Don't look at what was actually said there, but think about what the writer of this sentence assumed that the reader would already have known.

Nowadays, these so-called Winograd sentence pairs are examples of that, where you have something like, "Fred tried to put the package in the suitcase, but it was too big." What was "it"? Was it the package or the suitcase? You know that it was the package, but why do you know that? It's because you know big things don't fit in small things and so on.

It's that introspection wherever you see an ambiguous word or a pronoun at the end of one sentence and the beginning of the other. The writer assumed you would infer various things happened that would allow you to understand whatever illusion was taking place. Yes, it was introspection based on looking at random fragments of text.

Commercial versus R&D Projects at CyCorp

Thomas Haigh: My question I think builds on what Peter was talking about. About 25 years ago, when I was doing my undergrad AI class, we heard about Cyc. We heard about it at least in the context as this symbolic AI community putting its best foot forward to try and see if these approaches, with sufficient investment, will produce a situation where the knowledge base is sufficient that it can bootstrap itself and start reading the newspaper and encyclopedias. When I heard you talking about Accenture and enterprise applications and business things, I thought, "Oh, it must have gone in a different direction," or at least maybe you're seeing what you were doing differently. But then what you were talking about now it was back to achieving orbit and making that transition.

I guess my questions are first has your personal sense of what the project is about changed over time or remained constant? Second, have we learned anything about intelligence from doing this, and if so, what?

Lenat: That's really sweet, thank you. Nobody ever phrased it that way to me. For the first part I would say, “No, it hasn't changed at all.” This is merely the mechanism to support the effort without putting some kind of existential risk by taking a large investment or by going public or by going into debt or something like that. These are the mechanisms by which we fund, let's say, the 50 percent of the effort, which is dead on track.

Don't think of it really as 50/50; think of it as 80/80. We try to choose application projects for these verticals where a nontrivial fraction of what they want done will in fact forces us to add more knowledge to Cyc. We're doing some application for Cleveland Clinic that involves knowing that when you have fluid flowing in a conduit and it gets pinched then pressure builds up upstream and goes down. You want to represent that in ways that when you do oil well stuff that same piece of knowledge will be used and so on.

We try to pick projects and we try to represent things at a level within those projects where we're cutting as few ontological corners as possible so that the knowledge will be maximally usable in the future. It's really 80/80, not 50/50, in terms of in the direction that we set out for and what we have to do to support ourselves.

In terms of learning about intelligence, I hate to say, “No,” because that sounds sort of sad, but I don’t think we've actually learned that much more. We took as our null hypothesis starting 35 years ago that we knew enough about knowledge representation. It turns out we didn't, but we learned a lot so the high-order logic stuff helps in that regard. We knew enough about reasoning efficiently, so it turns out we were wrong about that, but we made these 150 mistakes that led to 150 engineering breakthroughs, or mini breakthroughs. That sounds like a lot, but if you divide it by 35 years, it's not quite as impressive.

Basically, I think I started out with the right plan, and we just sort of have carried out that plan. I didn't think it would take this long, but I would say we're still on the same trajectory.

Grad: We're going to have an oral history done of you for maybe 10 hours, and that'll be David's responsibility, not mine.

Artificial versus General Intelligence

Ed Feigenbaum: Just one question. Sorry about the voice. A follow up to Tom's second question, though I wasn't even going to bring it up —yesterday, is about was the general conversation about AI. Whenever I get up and talk, and my slides are on whiteboard and I talk about AI, I always cross the A out. I say that it has nothing to do with whether it's a machine or not; this is what intelligence is.

In the first decade of AI's work, there was a great deal of focus on, as I mentioned yesterday, generality and problem solving— for example, Newell and Simon's pioneering work on the general problem solver used a knowledge base. If you go back and look at the individual efforts like the trigonometric identities solutions and all that, they were trivial knowledge bases, like 20 things to know instead of 20 million things to know. That was not convincing at all.

When we got to Dendral, it was clear that the power of performance was in the knowledge base; we were using relatively simple generated test algorithms for reasoning. As we continued to pile up these expert systems, Al Newell and I had this discussion, and I guess other people have picked it up. It's called the

Big Switch Theory. Al Newell said, "Oh, Ed, you're a proponent of the Big Switch Theory of Intelligence," which means there's no general knowledge: you are doing this and then you're doing that and then you're doing that and then you're doing that. That's what Doug learned when he turned off the general problem solver. He learned that he needs 1,100 of these specialized agents, but then he probably needs 11,000 of these specialized agents.

Lenat: Don't say that. How about 1,200?

Feigenbaum: Anyway, when you see the conferences—you have AAAI conferences and neural network conferences and so on—there's enough brand of confidence that you should be very skeptical of. It's something like AAAI but it has a G in it, where the G stands for general, I can't remember exactly what the acronym is.

Unidentified Person: AGI: Artificial General Intelligence.

Feigenbaum: Anyway, don't believe it. That's not where it is. There is no such thing. For example, John McCarthy's whole career aside from inventing time-sharing was in formalizing, giving us the tools to create this general reasoning engine sort of along the lines of his early paper in 1958, "Programs with Common Sense." Yet in the end, to get something done, none of that was used. It's a wonderful legacy, but it wasn't a practical engine.

Grad: I'm going to call time on this. Let's say we could go on with Doug's company forever; most of you would find it exciting. I'm ready to go to the next.

Syntelligence

Grad: Peter Hart, I think you're number two today with Syntelligence.

Hart: I'll begin with the backstory that started in 1980 when I started the artificial intelligence lab for Schlumberger in Palo Alto using their just acquired Fairchild subsidiary as kind of a domicile of convenience. After about two years, Dick Duda who had joined me there and I decided we wanted to start an expert systems company, but we remembered the experience of Machine Intelligence Corp. that we talked about yesterday. This time we found a couple of business guys to join us, and then the four of us started in late 1982 and got it off the ground in 1983.

Grad: The names of those people?

Hart: The money guy was Sheldon Breiner, and the marketing guy was a brilliant computer scientist and Harvard MBA named Steve Weyl. Syntelligence had a distinctive business model and very distinctive technology in a somewhat distinctive kind of arc that it traversed, so I'll briefly describe each.

First the business model: Syntelligence developed end user expert systems for large banks and insurance companies. The systems were used to analyze the financial risks for commercial lending and commercial insurance underwriting. I emphasize the word “commercial” because assessing the risk of lending or insuring say a half a billion dollar a year company is very different from doing the same thing for individuals and a totally different and much more complicated approach. We did not offer an expert system shell or tool or general training or anything like that, so this was for end users. That was unusual in that era. I didn't know anybody else who was actually doing something for that, so that was the business model.

The technology was very distinctive. Initially, Dick and I thought that all we needed to do was re-implement Prospector, and we'd be off and running. Once again, we were very naïve and not close to being able to do what was needed. We wound up with two other brilliant people that we brought over from SRI: René Reboh and Tore Risch. We invented a completely new system based on the ideas of nonprocedural functional programming with some twists.

One twist was that we had functions over random variables. For example, if you wanted to compute profit equals revenue minus expense but you were going to do it for next year, you were making guesses, so you had a probability distribution around that. Then you had to compute a derived distribution for the uncertainty about next year's profit. That was built into the infrastructure.

Another twist was that these were functions over arrays, so it was partially a relational database system. For example, if you wanted to know, "What's the profit for the next five years over these geographical territories?" you had the same idea, but you were array-based. Those were kind of core ideas to be the side-effect-free functional programming language.

Another thing was interesting: the control regime was pure data flows, almost pure feet forward data flow that was governed by a very tightly coupled user interface that we called the Form System because these analysts always looked at big forms, and any change, any update in a data item would trigger a feet forward.

There were quite a number of things like that that were very distinctive. We talked yesterday about transparency and explanations. I developed an explanation subsystem that did a pretty decent job of allowing you to drill down on figuring out the most important factors in coming up with a recommendation. There was a lot of richness there and a lot of expressiveness but very much attuned for financial risk analysis.

The other distinctive part of the technology was the computing environments. We had a very rich knowledge-engineering environment that was implemented on some workstations that included some niceties like an incremental compiler. You were asking yesterday about interpretative versus compiled code, and we were trying to get the best of both worlds by having an incremental knowledge-based compiler.

In a knowledge engine, you make a small change—just that part got compiled and you ran very quickly. What was more unusual was the runtime environment. Our customers were 100 percent true blue owned by IBM, so our runtime environment had to be an absolutely plain vanilla, big iron IBM application running. MVS was the operating system in those days, and CICS was the telecommunication controller, so we had an MVS/CICS system that looked plain vanilla, except we figured out how to get IBM terminals performing as bitmap displays.

When you looked at these IBM boxes, you thought you were looking at an early Mac or a Xerox Alto or something like that. We had squadrons of IBM engineers descending on us, saying, "How the hell did you do this?" The end user had an experience very much like mousing around on a Mac, even though they were using these character oriented PCATs or 32… I forget what they were called.

Grad: 3270s probably at that point.

Hart: Something like that, that generation. There was a lot of very distinctive technology around this.

Syntelligence Licensing and Services

Grad: Your income was from using these tools to perform the services for your clients.

Hart: Not at all.

Grad: Then tell me what it was.

Hart: These were products. We had license fees. Initial license fees ranged typically from $1 to $3 million for initial license fee. As we've been saying several times, we often had services associated with that, but that was mostly to get customers installed, up to speed, and able to be on their own. We were not interested in being a service company primarily.

Grad: Once they bought your software…

Hart: Licensed it.

Grad: Licensed, I'm sorry, you never sold software.

Hart: Never.

Grad: Nobody sold software. Once they licensed it, you trained them sufficiently so they could use that software and do the work themselves.

Hart: And maintain it because the knowledge bases were always evolving. After that, there was a 15 percent or so annual renewal license fee for maintenance and so forth.

Grad: What level of revenues did you reach then?

Hart: I think somewhere between $10 to $20 million a year.

Grad: Isn't that interesting? Everybody when I ask them how much their revenue was, they were all around $10, $15, $20, $25 million. …

Hart: There's more to the story. it was pretty impressive at multiple levels in my opinion, and I still don't know of any similar architecture that's been implemented, but maybe there were some. Doing all this fancy stuff took many years longer than we initially had expected, but by the late 1980s we were both profitable and cash-flow positive. We had happy customers in four countries on three continents. We had many thousands of professional users who were typically credit officers or insurance underwriters. Also, their bosses would occasionally use our explanation system to drill down:why did you come to this decision or make that recommendation? We analyzed in total many tens of billions of dollars of potential financial transactions with this huge group of professional risk analysts in these insurance companies.

Syntelligence Sale and Closure

Hart: Yet, Syntelligence isn't around today.

Grad: How come?

Hart: How come? I'm glad you asked. I well remember us in January of 1990 sitting around and saying, "Man, things are going great. We have so much cash on hand. We could go for a solid year without selling anything." That's exactly what happened. You might well wonder why.

The Great Recession of 2008 more or less obliterated memories of all previous recessions, but in 1990 there was a recession that was one of the most severe of the post-war era up to that time. Unfortunately for us, it was especially severe in the financial services sector that happened to be our market. We were selling big ticket, enterprise-wide mission-critical software. Every purchase decision finally made its way up to the very top of these big banks and insurance companies. It was a CEO or the chairman of the board or somebody very much like that who signed off.

I will tell you that in 1990 those guys were not signing off on a nickel of expense that was not absolutely required to keep their current operations limping along. Our goose was cooked, but not for any reason related to AI winters or hype or over-promising or anything of that sort. In early 1991, I closed the company down and I sought excellence elsewhere.

Grad: Did you sell it?

Hart: Partially, as a matter of fact. Our single largest customer was the largest property and casualty insurer in Great Britain; now it's called the Royal and Sun Alliance. They were very invested in us and over time, they had both licensed a very large version, a site license, and made an equity license. They were so invested in us that they bought the assets and hired many of the technical people and that kept going for quite a few years. After that, I finally lost contact with that. But it wasn't like going public; it was basically a fire sale of the assets.

Grad: How much?

Hart: No, I really don't remember.

Grad: How many zeros?

Hart: At least seven zeroes.

Grad: But not eight.

Hart: No. No. It was basically a fire sale because they had I don't know how many thousand users and they wanted to support them.

Grad: Did they continue to sell it to other people?

Hart: I don't know.

Grad: Did they continue to support all the other customers you had taken care of?

Hart: I don't know. I was not part of that deal. I was the old guard, and I was out, so I don't know really what happened after that.

Syntelligence Application Areas

Brock: Peter, were people using the system for trading to analyze the risk for trades?

Hart: No, not at all. I looked at half a dozen financial applications including trading and including foreign currency exchange at the beginning of the company's life as possible application areas. I will tell you one thing about Wall Street traders: every person you speak to has an even more compelling story than the last person you talked to about why their strategy is great. What I learned from that is you don't survive as a Wall Street trader without an absolute fabulous story.

But nobody believed the expertise, so that was not the kind of thing we wanted to syndicate. We wanted to syndicate the knowledge of senior financial risk analysts—they're called, underwriters in the insurance industry—and credit officers and credit analysts in the banks. The very best people are considered to have much more operational expertise than the higher turnover lower levels. That was really the sweet spot in the market.

Brock: Even risk management for trading I guess is different.

Hart: No. We wanted high volume, so we went after what's called the commercial middle market, which is much bigger than mom and pop companies but smaller than say an IBM or a Google. Those companies self-insure, and they use insurance companies as operating managers. They're not really risk managers; they're financial managers. That middle market is where there's a huge opportunity and there was.

Grad: You say you're using statistics, but in effect when they used your programs and so forth with their reasoning. They came up with an absolute range, an answer. It wasn't to build norms.

Hart: We did not use statistics in the way you're thinking. We relied on industry statistics from providers like Best & Company as just part of the input, part of the data for analysis. We used basically probable decision making based on underwriters' judgments.

Syntelligence's Functional Programming Approach

Haigh: I was struck when you talked about different technologies that were about. You started with the programming language and said that you built that into the arrays and functional programming approach and basically data types that reflect those probabilistic things. Was that idea of producing a programming language that is in that very direct way coupled to what you believed would be the needs in this particular industry, was that a novel approach? Do you know, is that something other people did?

Hart: To the best of my knowledge 23it was unique. People here know what functional programming languages are: side-effect-free, nonprocedural, functional programming languages. You know what expert systems are, you know what relational database systems are, and of course you know what spreadsheets are. This system integrated all of those in a very clean side-effect-free way with this very simple control regime, this very simple data flow regime. I think it was really quite an unusual architecture.

Grad: Are these tools and so forth still in use?

Hart: I served as an expert witness in an intellectual property case, and it turned out that my own prior art was the most relevant art. Twenty-five or 30 years after Syntelligence closed, with an old

Syntelligence guy, we resurrected a system. I did a save and retrieve, and this thing came to life. I hadn't seen it for like 25 years; it was just astonishing. Other than that, no.

Syntelligence Knowledge Acquisition

Colin Garvey: Did you call the system an expert system when talking with your client?

Hart: Yes, we did.

Garvey: How did the knowledge acquisition go? Did you do a lot of interviews?

Hart: Yes, that worked well. It's a good question. That worked great. We did these typical interviews that we refined over a period of time. We got their best experts to do knowledge engineering on their own without an intermediary once things had been basically built, and then it was kind of refining and extending. It was very simple, very simple interactions.

Herb **Schorr:** You seemed to have run a successful business for a product which was useful. How come nobody revived that, and why isn't there a new version of Syntelligence?

Hart: Probably because I was too tired to do it again. I don't really know, Herb. It's a really good question. I don't know how those things are done today. I mean I was up to my eyeballs in that in the 1980s, but I don't know if there are better decision methods today. I'll tell you that some things like personalized insurance are not a right area for this. For example, when I was thinking maybe I should go for that market, the head of Allstate Insurance once said to me, "You know, Peter, once you've seen 40 million drivers, you've pretty much seen them all." So very simple schemes work for that; you need just the right level of complexity where it's too much for a junior person but not so much of the technology is overwhelming. That was really a sweet spot, a huge market, something we could address. I don't know why there's not something there now. Maybe there is, I don't know.

Neuron Data

Grad: Next, Alain Rappaport is going to talk about Neuron Data. Then we're going to take a break, so if any of you are getting anxious.

Alain Rappaport: I'm going to disrupt a little bit what we've been saying here by proposing that instead of a walkthrough a cemetery with businesses, we are actually witnessing an important moment in the development of computer science where everybody, I mean maybe in the range of software companies at large, there are more successes even in this room than there is statistically out there because it's a fascinating technology. Listening to Peter or to Doug here or anyone, it's still extremely topical. I'm just trying to make a case that this is a very exciting group of people and definitely not a cemetery.

Your last comment was, "I don't know if it's getting done or not." Let me use this to make the point a little bit differently now: this a very successful group of people precisely because it became embedded in computer science curriculum. If you decide that an expert system is just, “Let's be able to put some rules together, some kind of control structure, and it is already useful and then you've won.” Today most of the things we're talking about like, even the system you're talking about, is probably doable by putting some R libraries together—R being a statistical language or the Facebook environment for development or Google. Rules are in there. You see what I mean? They're in there. You can actually build an expert system in no time; they just don't know that they're building an expert system necessarily.

Grad: This afternoon I hope we're going to be talking about how this has become an everybody does it kind of thing and find out what your thoughts are. I want you to tell us about Neuron Data, your company.

Rappaport: When Neuron Data started, it was kind of a different story. We did the first implementation of our system called Nexpert on the Lisa. I don't know how many people know about Lisa here.

Hart: Well, sure.

Friedland: I'm sure they have one downstairs.

Rappaport: Yes, exactly. We should go there and actually run it. On the Lisa, until we got invited to chat with the team at Apple, it was building the Mac and that's where we put it. We literally lived there and fed ourselves on the free Odwallas that Steve [Jobs] had declared should be a benefit of people working at Apple.

Grad: Start at the beginning. You started a company when?

Rappaport: We started the company in 1985.

Grad: Who was with you? Who were your partners?

Rappaport: My partners were Patrick Perez and Jean-Marie Chauvet.

Grad: Where was the company located?

Rappaport: Palo Alto.

Grad: You were here at that point in time.

Rappaport: Yes. We just came here and rented an office.

Grad: Why did you start a company? What did you have in mind?

Rappaport: It's a good way if you like to measure the success. It was just a very exciting time.

Grad: No, my question was, what was it you wanted to do when you started the company? Did you have a particular thing you were trying to build, sell?

Rappaport: We wanted to sell the ability for people to put together a knowledge base in a specific area that would be their expertise and run it at a very low cost on a very small machine. It's pretty difficult to describe, but we talked about the LISP machine yesterday. LISP machines were indeed absolutely gorgeous. They would inspire anyone, and we looked at the Mac as a small LISP machine, very tiny. We built a very graphical user interface so that anyone could just play with this thing. That was very, very central to our paradigm.

Grad: This was in 1985?

Rappaport: 1985.

Neuron Data Financing and Customer Base

Grad: Did you need financing to start the company, or did you self-finance?

Rappaport: We needed a little bit of financing. We got some venture capital in pretty early.

Grad: Big money, hundreds of thousands, millions?

Rappaport: $250,000, I think, with which we ran until a mezzanine round before the IPO in 2000.

Grad: Pretty good.

Rappaport: We were very rapidly profitable. I mean, my first sale was in 1985. It was like you sell a tool that's put to use very, very swiftly. The wind, if you like, was really with us.

Grad: Who was your market? Who were you trying to sell to? Did you have a particular industry or area that you were trying to sell to?

Rappaport: All industries, all areas. I'll tell you an anecdote.

Grad: Go ahead.

Rappaport: The first time we showed the system was at the AAAI Los Angeles. LA was 1985, maybe. We didn't have any room so we were able to rent a little space about two feet by three feet, organized our computers as a totem so you could kind of read the different things.

We didn't have any experience building a highly specialized knowledge base, so I built one for fun, for illustrative purposes, that was about diagnosing high blood pressure medicine. That was one of my post-doc efforts. Somebody came, looked at it, and said, "Super interesting, I want to use it." Our first customer was Hughes Aircraft for antisubmarine warfare, which I'm sure some of you had as well. Very quickly it became anything and all topics. I don't know when you want to talk about applications.

Grad: We'll talk later. When someone tells me that everybody's a market, I don't know how you market to them. I was always wondering, how did you get to these people? How did you get the people to know about your product?

Rappaport: Right. Marketing is important, always important. I don't know if our efforts were very different from anyone else in terms of going to conferences and explaining how it works, but the key was also that we wanted to be a tools company so we didn't want to do services because we didn't understand anybody's domain better than they did so we kept it as software.

A lot of our effort was minimizing the services. However, there were companies that were interested in becoming, if you like, service companies around Nexpert, which had become by then maybe Nexpert Object. We quickly moved by the way from the Mac also to the PC with Windows, so we were one of the four companies that Bill Gates invited on Wall Street to present interesting software on Windows. We did the demo, and of course the IPO is history now, but it was a pretty interesting time.

Grad: It's a very different model than most of the others, isn't it?

Expert Systems on PCs

Harmon: In 1985, the PC at the time was a 286 machine, and all of a sudden there was a great interest in expert systems. Up until that time, all the expert systems were running on big workstations, Sun, PDP-11, something like that. Neuron Data came out on the Macintosh. All of a sudden it created a user interface that looked much like what the big guys were doing on Suns and so forth.

I think it was less their marketing than people beating a path to their door. I mean, all of a sudden if you'd been looking around and thinking, "We should play with expert systems. Our company should learn more about it," and you looked at the couple of PC versions that were out there, they looked like old IBM blue screen with type in one line at a time sort of things. They just weren't serious. Then all of a sudden you looked at Neuron Data and here was a graphical environment. You could see where the different rules

were being kept in what folders and so forth. It looked like a Sun machine. It looked like one of the big guys, and it was available at a reasonable price.

Grad: What were you charging?

Rappaport: I don't remember exactly how much we charged, but it was a bit more than the Mac itself so maybe $2,500.

Grad: You ended up with thousands of customers, tens of thousands? You end up with a big market apparently?

Rappaport: Yes, it was a pretty big market. One principle was to be a tool. The other one was to be embedded with a callable interface, which is an API now.

The market for our tool was Fortune 500 companies worldwide. Of course, Palo Alto was our headquarters, but we had offices in Paris, London, Frankfurt, Tokyo. It was very, very interesting. The problems were very different, but embedding was very important for us. It was also written in C, which gave us an advantage over issues with LISP.

Reid Smith: Was this all in English, or did you do it in French, in UK English, in Japanese, and so on?

Rappaport: We did it in several languages. I mean the challenge is of course when you have to go to Kanji and these, but we were working with Sony and a bunch of companies of that nature, so at some point you needed to be in the native language. That's when we developed the first non-AI tool for open interface; that was a way to port your application to a computer that would run Kanji for example. That was a very useful piece of technology because you could get it on any kind of interface you wanted just to use the API.

Neuron Data Runtime Licensing and Applications

Grad: Was your product embedded in other products that were sold? Did someone else take yours, embed it in something they were selling, and did you get a royalty? I'm trying to see how that was used.

Rappaport: People making products with the product you mean.

Grad: Yes.

Rappaport: Yes, I think there were a few. I'll try to remember who did that. I'll just use some very quick examples; we can double click on them later. Tandem Computer was well known for the

redundancy, so they embedded an expert system to make decisions in real time. They had some knowledge about when they would want to actually move to one of the new systems. American Airlines did another version of the algorithms to try to intelligently load airplanes.

Grad: But did they pay you a royalty or just simply license it?

Rappaport: License fees.

Grad: That was it.

Rappaport: Yes.

Grad: They might have sold it 10 times or 100 times in every one of their machines, but you weren't getting a royalty for each sale.

Rappaport: No, we would sell runtime systems, license the runtime. The development environment, the runtimes, that was a pretty clever way of going about it. There were lots of applications of interest. Like academically, the Human Genome Projects used our graphic UI to start kind of breaking down some of the knowledge in there.

Of course, one of the most amazing applications that touches on things we discussed before was FinCEN, the Financial Crime Enforcement Network that Ted Senator built. It was a beautiful application with a blackboard architecture and the expert system making decisions of some sort. The rules were incredibly simple. I don't know if I'm at liberty to tell much about this, but the return on investment on this application was unbelievable.

Grad: Sounds like it.

Rappaport: They said it was over a billion dollars of return in terms of detecting money laundering issues.

Denny Brown: I have a comment on your question about marketing: How do you do the horizontal when you're not focusing on a particular area? In 1984, 2005, 2006, people were coming to us, all of us with, "How do we get into this? How do we do it?" His niche at that is, "Here's an entry point that you can walk in for only $2,500 and get started." The demand was already there; the demand was very high.

Grad: If you built it, they will come?

Brown: it was very much that way. Some of that demand was generated by some of the bigger companies who were selling the same kind of thing for $10,000 and $20,000 on these machines which

were also that level. The comparison was quite dramatic that you could get going, you could get real application done on a PC.

Rappaport: If folks were interested in doing what Mark Fox was talking about yesterday, like constrained-based reasoning or configuring the elevator—when to call it and do it on the queuing system—we just didn't do that. We were not really interested in complex engineering problems. It was less AI-ish in fact for us. We were more interested in, “It's so easy to put the rules in, so put your rules in as a financial manager or whatever and just get it running.”

Grad: This is what we might call bread-and-butter type applications.

Market Crash and Neuron Data Sale

Grad: Tell me about the company. How long did it last? What happened?

Rappaport: Like I said, our market was increasingly Fortune 500, and we went public.

Grad: In what year?

Rappaport: 2000.

Grad: How much money?

Rappaport: It was an interesting collision of issues because we were a profitable enterprise, and we ran into the crash that everybody ran into as well.

Grad: You sold before or after the crash?

Rappaport: During.

Grad: I see.

Rappaport: I don't exactly know when the crash was, but we went public in March 2000.

Grad: It would be about a month before the crash really.

Rappaport: Time to get things organized, and the crash was doing the damage across the board anyway.

Grad: What happened after that?

Rappaport: We got bought actually by a German company called Brokat. They bought the public company, and they themselves sold the technology and the platform to Fair Isaac, where I think today there are still people using it or at least the basic principles to do the FICO scores. It became a very interesting vertical asset

Grad: What year did you sell it to the German company?

Rappaport: I'm not sure of the timing, but I think it was in the months following the IPO.

Grad: IPO? Did you stay with the company at that point?

Rappaport: No, I became an observer on the board in 1998 I think when we did the first financing, the first substantial VC round.

Grad: Fascinating. Any other questions or comments?

Expert Systems Industry Competition and Hubris

Haigh: Building on what Denny said, as I'm listening to you talk about customers and marketing, it seems like in, for example, the relational database management software industry that came up during the 1980s, there's a period at the beginning where new companies like Oracle and Informix and Sybase were not so much competing with each other as persuading the customers that the minicomputer based relational database system is a good thing to have.

Then there's a point later on when people have bought into that. It becomes more of an industry, and they're competing with each other because the customers know they need a relational database management system. The question is which one do they pick.

Do you think the expert system industry ever reached that point where you were competing with each other for customers who knew that they needed a technology like this, or did it always remain that your biggest opponent really was in-house development or just not building a system? Did you ever reach a stage where a lot of the other people around this table were competitors? There were customers who already knew that they wanted to acquire an expert system technology.

Rappaport: Yes, it had reached that point even though all the stories are quite different. There was a healthy competition, just a very healthy competition going on.

Feigenbaum: I'd like to use Alain's story as a jumping off point to pick up on what Paul said. The overarching theme here of these comments is hubris. The early expert system companies were suffering from great hubris. Namely, they thought that their software was a really big deal, and "Boy, we really know a lot," and "Boy, is this hard to duplicate." Well, that was a lot of bologna; it was not. For example,

Teknowledge could have licensed the E-MYCIN software from Stanford and gone to market with it. Instead, Lee Hecht decided to spend a million dollars developing, what was that package called, M1?

Brown: S1 was the big one, and M1 was the little one.

Feigenbaum: Anyway, that was priced according to roughly speaking what kind of hardware it was run on, a Sun workstation or something. It was like $10,000?

Brown: Twenties.

Feigenbaum: Twenty? Then all of a sudden Osborne in Berkeley comes out with a $99 PC program doing the same thing?

Brown: And Neuron Data.

Feigenbaum: And then Neuron Data. The price points were dramatically different, and the expert system companies just got blindsided by thinking that they were big shots, that they had some really big powerful stuff here.

Brian McCune: What they really had was not the tools; it was the expertise to build and tune the knowledge bases on hard problems.

Feigenbaum: That's right.

McCune: He was selling a shrink wrap, and you bought it. Good luck.

Feigenbaum: Let me finish the comment.

McCune: Got a different market.

Feigenbaum: If you look in our book, *The Rise of the Expert Company*, you'll see a chapter there on DuPont. Dr. Ed Mahler from DuPont pioneered the idea that he didn't want all this expensive software. He was going to buy the cheapest stuff on PCs and give it to the chemists and teach them how to be their own knowledge engineers. Then he wouldn't hire knowledge engineers. These chemists were smart; they were all PhD researchers at DuPont doing their stuff. And it worked. There are all kinds of great stories.

Grad: Later on, we're going to get to the point where we talk about what happens next. This seems to be a transition point where, all of a sudden, it's not big hardware and big prices. Now, "Hey, we'll provide the guys with the tools," and we begin the DIYs. It's the do-it-yourselfers is what you seem to be saying.

Feigenbaum: Well the technical reason why this works is that if you believe in the mantra, "In the knowledge lies the power," what we're saying is that the knowledge of the domain reduces the search necessary for a problem solution drastically. The knowledge is the most powerful heuristic you have, just world knowledge and therefore the search spaces were very small. PCs were perfectly able to deal with that.

Smith: I wanted to go back to what Alain said about FinCEN. FinCEN.gov was looking for money laundering activity. They reported $28 billion identified in October of 2015. Suspicious activity related to money laundering. It was a big deal is my point. It's just that this was at a scale that my little mind has trouble with.

Grad: You're saying this little program enabled them to do this?

Smith: That's what I'm saying.

Rappaport: Right.

Smith: Absolutely. That was the first one they did. The second one they did was for identifying securities fraud. Did Ted write that on Nexpert also, do you know?

Rappaport: I only know of the money laundering application, detecting money laundering paths and getting there on time to prevent it.

END OF THE INTERVIEW

