Oral History of Gary Hendrix

Interviewed by:
Dag Spicer

Recorded: November 19, 2004
Mountain View, California

CHM Reference number: X3008.2005

Table of Contents

Gary Hendrix

Conducted by Software History Center—Oral History Project

Abstract: Gary Hendricks discusses in some detail his interest in natural language analysis and machine translations. He covers a number of his Artificial Intelligence related projects at SRI International. He describes how he and 15 other SRI employees formed the Machine Intelligence Corporation and the reasons why it didn't succeed. He then goes into starting Symantec, how it was financed and the problems of having too many Ph.D.'s and the difficulties of trying to bring a more production-oriented approach to its offerings. He discusses Symantec's initial products, its organization and its transformation into a publicly-held systems utilities company. This oral history ends with Gary Hendrix leaving Symantec in 1991.

Dag Spicer: It's Friday, November 19, 2004. I'm Dag Spicer and I'm here with Gary Hendrix at Mountain View, California at the Computer History Museum. Gary was the founder of Symantec in 1982. I just want to talk about two tracks with you. One is your professional disciplinary track which is computational linguistics and how you got interested in computing and the problems of language and language recognition, and then your experience with Symantec which you founded in 1982 and any lessons you might have learned from there. We'll talk about numerous things. Does that sound all right?

Gary Hendrix: It sounds like fun.

Natural Language Understanding

Spicer: Great. I'll just throw out a date, 1970. ARPA came out with a series of funding for six different projects in natural language understanding, I believe. And this is about the time I think that you got into the field. Is that right?

Hendrix: Well, that's about right. I was an undergraduate at the University of Texas and I finished that up in 1970 in May and then I quickly got a Masters in computer science. That ended in December of 1970. And I was beginning to get into artificial intelligence just a little bit and sort of overlapping with operations research, but then beginning in January of 1971, I was fulltime thinking about artificial intelligence and my supervising professor was Robert Simmons who was into computational linguistics and he was the man on the UT campus who was wired into that world. I also took a number of courses in other AI-related areas like robot planning, Laurent Siklossy, if I get his name right (he was always on my case about not saying it right), was another influence because he got me interested in robotics and, to some extent, the representation of semantic knowledge which I then used with Simmons and that's where the name Symantec eventually came from. I did some work in the area of robotics and planning and I wrote a paper that I submitted for publication in one of the journals. And the people at SRI International read this paper and invited me out one summer. And there was a conference at Stanford. I guess it was one of the first joint conferences on artificial intelligence. And so because they had read this paper, they were interested in me and I eventually got hired by them even before I finished my Ph.D.

Spicer: For people who don't know what natural language understanding is, can you just tell us a little bit about what it is? It's not speech recognition, is it?

Hendrix: No, no. It's trying to get computers to understand naturally-occurring languages like English or French or German as opposed to artificial languages like COBOL or C or FORTRAN or SQL, something like that. And there was a time when computers didn't have graphical user interfaces and you had to communicate with them through some kind of language. You could argue that a graphical user interface is a kind of language too, but I'm talking about a more traditional language in which there are sentences of one kind or another. And we had thought that perhaps the use of English would open up computers for a wide range of uses and also, just as a science, it was and still is a fascinating field because we really don't understand much about how the human mind works. We don't really understand very much about how people process language and how they extract the meaning from it, how they think about things other than numerical kinds of things. And how you represent these in computers even thirty years later is still a pretty big mystery.

Spicer: I'm very interested in the disciplinary underpinnings for that, which let's just call it a disciplinary push versus say a contract pull which we'll get to later. The U.S. Army had a great interest in voice-activated systems and so on for which your technology and interest would naturally meld. But in the theoretical foundations of computational linguistics, does that develop around Noam Chomsky and the early attempts in the late 1950s with artificial intelligence and the belief that we know a little bit about linguistics, enough to know that maybe we can extend it with a computer and produce really interesting research results, if not practical products?

Hendrix: Well, certainly Chomsky was the big intellectual contributor to the field and he's an absolute giant. There's no one like Chomsky. And he developed many theories involving language. A lot of them were fairly mathematical in nature, describing how you could represent various kinds of grammar, how different levels of grammar behave mathematically. And people picked up on this and tried doing things with it. In one sense, the compilers are based on some of Chomsky's work in that the way you look at languages that you could use to design languages for machines.

And what the people trying to understand natural languages were doing, well, they were saying, "Well, okay, we're going to try to write a parser or we're going to try to understand the grammar of English." Computer languages are designed in such a way that once the syntax is understood, the meaning of what's being expressed is also readily understood because the whole thing is artificial. It's built up out of little pieces in the way computers are built up so the fundamental parts are all very well understood. I mean, if you get down to the bottom level, it's ones and zeros, offs and ons. And then the components that are built up out of that are well understood because you know how to decompose them all the way to the bottom.

So computer languages are going to be about how you put information into registers, how you move them around, how you perform computations with them mostly of a numeric nature. Natural languages--we can't go in and find out what the equivalent of ones and zeros are in the brain or in the mind if these two things are different, we don't know.

And the kinds of things being talked about are quite different. Ultimately, what a computer is dealing with and what computer languages are dealing with is the moving of the ones and zeroes in and out of registers, in and out of arrays, in and out of vectors. And everything is done that way. What human language is about, well, we can talk about that. We're having a conversation about it here in English and so you can talk about computer-related things in English, but you can also talk about love, you can talk about art, you can talk about the taste of wine, you can talk about the human condition, you can talk about commerce. How are we doing that? I guess the artificial intelligence hypothesis is that there's ultimately some way to reduce all this to the ones and zeros. But how do you get from here to there because instead of having a mechanical process that's ultimately talking about a mechanical process--which is what compilers are--we're trying to think of some kind of mechanical process that's talking about what we assume are non-mechanical processes, namely the entire universe of knowledge.

Spicer: Right. Now, it seems to me that natural language is a kind of a special niche in AI in the sense that language is one of the few things that distinguishes humans from machines or most other animals. And some people feel that language and in fact consciousness itself is really an epiphenomenon of just basic biochemistry. And we don't have to get into that unless you want to, but I'm very curious about a certain stage, maybe an inflection point if you want to call it that, in AI in the 1970s and 1980s in which it was an area of great promise in which the frontiers to AI seemed to still be fairly broad. I know they are still today, but the problems that we thought were easy actually turned out to be difficult and vice-versa. And like you mentioned, I don't know how far you would say we are from a natural language sort of understanding engine for English, let's say, but do you have any thoughts on the *zeitgeist*, if I can use that word--the world view of AI researchers at that time--in natural language. Did they really feel it was a solvable problem or just an interesting research topic?

Hendrix: Well, I think people thought it was solvable then. I don't know that they thought it was going to be easy, but Chomsky, in a theoretical way, had figured out an analysis of language which looked pretty convincing and it was.

Spicer: Generative grammar?

Hendrix: Yes, but even going back to more primitive forms, you could come up with a context-free grammar that could describe certainly not all of English, but a big piece of it. And the very fact that you had a grammar that allowed you to say, "Okay, I can characterize a piece of English this way; I can describe how the language works." You couldn't describe the whole language in a context-free way, but you could describe a piece of it and then there were mathematical tools to talk about what the limitations were of what could be described and what couldn't and where there's this problem of something that can't be captured. There got to be a mathematical basis for it and once you get something into the mathematics, then you've got a whole set of tools for thinking about it, for evolving it. And then along came computers. Well, computers are good at manipulating symbols. Now, we need to make something very clear here. When computers first came out, they were used to manipulate numbers. We were adding things up, we were doing multiplications. We were trying to figure out how to build H-bombs. You know, that's what they were all about. Or, when they went into business, we were trying to figure out how much money everybody had in the bank and keep these very clear records. So, it was about simple rote memory or mathematical calculations.

Spicer: Maybe a very finite knowledge building.

Hendrix: That's right, a very small number of things. So people would write a computer program to do some kind of a simulation or to solve a physics problem, but it was physics that drove computers and physics is driven by mathematical calculations and so it wasn't mathematics with a big M, it was glorified arithmetic, okay? And so they're good calculating machines. They were ciphering machines. Now, when we think about what computers are used for today, word processing is a huge thing. Well, you're using computers just to store the information and let people insert characters and format stuff for printing, but it's just keeping track of blocks of text. The computers don't understand what the text says; it's just a block of text. Or you're doing things like spreadsheets or you've got a database in which you've got blocks of data that are laid out in tables. And the computer doesn't know what the table means, it's just that there's stuff to put in this table and there's stuff to be brought out of it.

So an interesting idea that was becoming current in the 1970s—even the late 1960s--was that a computer could be used to think about symbols rather than numbers. So instead of thinking about arithmetic or instead of thinking about blocks of text, computers could be used to think about meaning or they could think about symbols and the relationship between symbols and meaning or the relationship between one symbol and another which is about as close as you can get to thinking about meaning in a machine. So there's the semantics. Computers can be used to attack the problem of meaning and not just regurgitating things as word processors do or manipulating pieces as is done with arithmetic calculations or insertions into word processors or databases, but actually performing some kind of reasoning, reasoning at the symbolic level. And that was really the basis for everything that happened in artificial intelligence; it was thinking in terms of the machine reasoning about facts.

Spicer: Going back to the idea of language and the word ‘meaning’--I know we don’t want to get too metaphysical here--but it seems to me that language is perhaps the key to consciousness or to our organic carbon-based human inference engines or whatever goes on in our brains and that the ultimate goal of natural language research… I’d like to know if there was a holy grail in the discipline at this time and how would you have measured success? How do you know when you’ve conquered the problem?

Hendrix: Well, I don’t know that people were thinking about conquering the problem.

Spicer: Or just how would you know when you had it solved?

Machine Translation of Language

Hendrix: Most of science isn’t about conquering a problem. I guess if you want to find a cure for cancer, you find a cure for all cancers and you can define that. But that really isn’t the way science works most of the time. What happens is you’re here and you know you want to go out there somewhere. There’s no particular place, but it’s a frontier and it moves. And so all you’ve got to do, your job is not to move from here to over there, your job is just to move a little bit. And so the idea was to take the new tools that were available, namely computers, and see, “Well, can we move a little bit?” And certainly, there were practical problems that people wanted to deal with like--it would have been great to be able to translate large amounts of messages that were written in Russian into English so we could find out what was going on the Soviet Union. We had cracked codes during World War II. So one way to think about Russian text is that they’re encoded English messages and all we have to do is crack the code.

Spicer: Machine translation?

Hendrix: Machine translation was intimately related to cryptography, absolutely, absolutely. And that was one of the original driving forces for the field.

Spicer: Do you know Doctor Peter Toma? Have you ever heard of him?

Hendrix: You know, I haven’t, or if I have, I’ve forgotten. And I forget so much that it’s easy.

Spicer: He prototyped the big translator; it was called Systran. It was a giant mainframe-based Russian to English and back again system. I know General Motors bought it and the DOD. But like a lot of these, I think they get you a certain distance and then you're always needing humans at the end to kind of disambiguate and polish or change inferences that were made by the software and so on. Sorry to interrupt. Are there other problems just like machine translation that might have been pulling this field along in terms of funding? Or maybe you can just speak about the first ARPA grant.

Hendrix: Well, they're two different things that we're talking about here. We're talking about natural language processing and then the whole field of artificial intelligence. Artificial intelligence was also concerned with problem-solving of all types. When you're not just running through a set of rote sequences to make a calculation, but you want to have the system reason to solve a problem. Doing a mathematical proof, for example, was a big thing. If you want to have robots running around in the world, the robot has to have perception. So the whole idea of machine vision--being able to look at the world and decipher what it means. And if you think about it, looking at a visual scene and seeing all the pixels that would come out of that and trying to assemble that and to have it make sense is an awful lot like having a sequence of characters in a written language. So they're both languages. One's a visual language, one's a written language, but they have some very similar problems.

Okay, if a robot can see and it understands where it is, it's going to have some kind of goals and it has to plan how to achieve these goals by going through some sequence of action. Figuring out the course of action is very much like working a proof. It's symbolic reasoning. So that was a big area. One form of symbolic reasoning had to do with figuring out relative probability. So if you're looking at something like diagnosis of diseases of the blood, "Which disease does this patient have?" And so some of that is absolute reasoning and some of it is sort of probabilistic: what's the most likely thing given what we see? You can almost call that like an intuition because it's not hard and fast, but based on probabilities, you can figure out what the best question is to ask next to zero in on the solution. So this was an area that was also hot in the early 1970s and that evolved into what was called "expert systems."

So I just want to say that I think that natural language is going to end up covering the entire artificial intelligence problem because it turns out that although people originally thought, maybe all we need to do is look up words in the dictionary to go from one language to another, or maybe all we need to do is understand the syntax of the language, we eventually came to understand that to extract meaning out of sentences, whatever entity is extracting meaning from sentences has to understand what's being talked about. And that's not in just one sentence, it's in the experience about the world and how the world's put together. And until you can build models of that even more than an encyclopedic knowledge – the kind of knowledge that a child learns by going out and experiencing the world, until you have all that somehow captured inside of a machine, the machine's not going to be able to understand language because the two are interrelated.

Spicer: This is a great point to ask you your thoughts on the Cyc project then, Doug Lenat, and how that might tie in.

Hendrix: I knew when that was being started and it's going on right in my hometown of Austin, Texas. But I haven't followed it so I don't know what its current condition is.

Spicer: They've been adding information, real-world knowledge for twenty years.

Hendrix: They have and I completely agreed with the general outline of the enterprise when it was begun twenty years ago. I think this was the general recognition of the entire AI community that something, not necessarily Doug's particular version, but something like that, and why not? I mean you can't fund everything, so let's fund this one and see what's going on. And so now that's been happening and evolving. And my life's gone off in a different direction and I haven't been following that very closely, but that was the problem and that was the goal that the project in Austin was trying to solve.

Spicer: You know, Marshall McLuhan once said that we shape our tools and they in turn shape us. And you can take anything from a hammer to a computer to a jumbo jet and find ways in which that's true. I don't want to force you into a mental box, but can you see ways in which that is true of natural language programming? I imagine that first people came with a human neuro-anatomic kind of approach. They were mainly medical or biological people who acquired computer skills and then the second wave, I'm just guessing, but the second wave would have been real professional computer scientists who then learned about physiology and such. Did you see any influence of the computer in terms of its architecture or its concepts that, in turn, shaped how people thought about the brain and the problem with natural language?

Artificial Intelligence

Hendrix: Well, I was very much on the computer side of it and thinking in terms of how can we create something practical or how can we just push forward the understanding of language from a mathematical point of view and not from a psychological point of view and certainly not from a physiological point of view. There were people that did that. I don't know how much good interaction there was back in the 1970s on this, or even today. I'm actually sort of skeptical about it, it hasn't gone too far, but I can say that as soon as we built anything mechanical, we could go in and we would find that it broke down almost immediately, that there would be something. You couldn't press them very far but what they'd break and then you could say to yourself, "Well, what are *people* doing that this machine can't do?" And that would give you insight about what to try next. And sometimes the mathematical formulations might give you some idea about what was happening inside the mind. For example, different kinds of parsing algorithms would result in different types of errors. And as you began to get errors that were more similar to the kinds of errors that *people* made, you began to think this is probably more like the algorithm that people are actually using. And in terms of the amount of look-ahead, the depth that you could have in the stack of a machine, you could begin to get some interesting insights into that. But that was not my field, but it was going on and it was pretty interesting.

Spicer: Wow! Let me just ask you one final sort of metaphysical question and it's related to an exhibit on chess software that we're doing here which finishes with Deep Blue, IBM's massively parallel supercomputer that beat the world champion Gary Kasparov in a couple of matches, the latest one in 1997. And, of course, machines of that power suggest something like Hal in "2001: A Space Odyssey," that is, a machine that once you get a certain amount of power, just raw computational power, there's almost a *gestalt* switch in which the machine *appears* to be intelligent. Kasparov commented a lot on this too where he actually literally says *it seems to be thinking*. Do you feel that, in terms of natural language understanding, is there a point at which we're just waiting for enough MIPS [Million Instructions Per Second – a measure of computational power], for example, or powerful enough computers? Is essentially the solution to extension of existing techniques or is there some completely new kind of thing that needs to be understood for natural language understanding?

Hendrix: Yes, I think I understand your question. I think that chess and natural language are fundamentally different and here's the difference. In chess, it's a very finite world. Now, the number of combinations of play is huge, but there are only so many pieces, there are only so many squares on that chessboard. Everything that could be done you can figure out what all the possible moves are. And what is Deep Blue doing? It's doing a very deep game of trying all the possibilities, but it's mathematical and the representation of it, even though there are a large number of individual instances of it, they're not fundamentally different in type. Because I could have a chessboard, a layout, and I can have this little matrix and I can say what is at each position and that's the representation of the whole world. And then I know how I can get from that world to a next set of worlds because of the rules for making moves. And now I've got another thing and it's represented in exactly the same way. So if I can just replicate that enough, if I've got enough memory to hold this, if I've got enough computational power to make all these calculations, I can look forward enough and deeply enough to figure this out. And that's my understanding of the way that the chess programs that have been successful are running, but I'm not an authority on that.

The natural language world is fundamentally different because instead of being able to talk about chessboards, a monolithic thing, and replicate them millions or billions of times in a way where you actually only have one model, in language and in general human cognition, you're not talking about a monolith, but you're talking about having-- if it's done with templates at all, there's got to be thousands or millions of them and they combine with each other in very odd ways. And so instead of having a small number of rules as in chess and a single way to represent the situation, you've got millions or billions of rules and you've got so many different combinations and different kinds of stuff and each one of these different kinds of things has to have its own representation. So more raw horsepower is probably not going to do it, although a great deal of raw horsepower will be needed by somebody and maybe Doug Lenat's project will begin to get us there. It will certainly get us closer if for no other reason by showing us what ought not to be done. Let's hope that he's doing it so we find out what should be done. But there has to be some fundamental breakthrough in figuring out how this stuff is represented. How do you represent all this knowledge about the physical world, about the interpersonal world, about everything that you and I talk about, and how do you get that into a machine? And probably there are a few keys to it that we haven't found and my prediction would be that we'll figure out some way… some magical representation scheme… or some way that spawns these paradigms for holding information. And then it'll be like setting a child loose and letting them learn and the machine will mimic that sort of human experience in building up a huge base of knowledge. And then we can go copy it on a disk because it will be a computer kind of thing and set it up quickly. But it's not just computation and it's not just a bigger machine; there's some scientific breakthrough, there's something missing that we haven't figured out yet. And somebody's going to figure that out one of these days and what a glorious day that'll be.

Spicer: That's exactly what I was asking. So it sounds like this problem might be an emergent kind of phenomenon and we may understand how to build kind of an infrastructure, but ultimately at some point, and I'm trying not to resort to divine intervention and so on, but at some point there are systems that do exhibit intelligent behavior with a very small number of discrete actions like little robot ants for example. You can give them five basic steps and they will replicate a whole range of complex behaviors that real ants do, and the same with birds flying all in their perfect V pattern and so on. People have started to understand those rules and "swarm intelligence" as it's called.

Hendrix: Right.

Spicer: Let's go back just a little bit to your trajectory at SRI and the speech understanding group there. We've talked a bit about theory and now let's link it up with your own personal career. And maybe you can take us from SRI until you founded Symantec.

Natural Language Processing

Hendrix: Okay. So when I went to SRI, it was at first to work on robotic planning because I had written a paper on that. And the paper was a follow-on to some things that had been published by the people at SRI--Stanford Research it was called then. But my interest was more in natural language processing so when I got there I sort of began to switch. And they had two areas. Well, they had many areas, but they had the robot planning people and they also were in the beginning of a speech understanding project that was sponsored by the Department of Defense and this went on for about five years and I think I got in on maybe the last four years of it, three or four years of it. I'm sure that the people at SRI were not particularly devoted to speech. In fact, by the time I got there, the component that had to do with the acoustics was being done-- it was MITRE or Rand or ISI, one of those outfits.

Spicer: BBN maybe?

Hendrix: Well, no, BBN had a rival project to ours. But SRI was working on the grammars, the lexicons, the semantics, some way to produce a response. And I think it was ISI, Information Science Institute down in Los Angeles, which was doing the acoustic part in partnership with SRI. And we had these two systems that we were putting together so they would guess words and then we would put the words together and we'd give feedback to say, "Well, this combination of words doesn't work," or, "This combination of syllables won't work, so try something else."

Basically, the conclusion of that project was that computers were deaf and they were really dumb, so it wasn't a great project in the sense of being successful at being able to understand speech, but we did learn a whole lot about language as a result. And a few years later, people began to be more successful with speech recognition. And today you can call up and instead of being irritated by a menu you can be irritated by a speech recognition system. And they do okay. They're not great, but they're okay. And they're used in very narrow niches like entering a number or talking about the name of an airport that you want to fly to. And it's usually you're picking a word out of some very small range of possibilities. And there are other things out like systems for being able to do dictation. And they're pretty remarkable. If we'd have had one of those machines that had that capability back then, we could have really had some fun with it.

Spicer: You just reminded me of a great scene out of "Star Trek," the original one, when there was an office with a typewriter that you can talk to. But this shows you a very interesting way that people conceived the future. You're not talking to a computer that's using speech recognition that then does something, takes some action. You're actually talking to an IBM Selectric typewriter that as you talk types out the text. So it's really interesting how people sort of view the future. That was their understanding of speech recognition and language understanding. Take an everyday object. You know, they didn't sort of carry forward about the current technology of typewriters that may not be around anymore. They just sort of tacked on this technology to an existing typewriter. It was quite funny. Anyway, so we're up to about 1975 at SRI.

Semantic Grammars

Hendrix: Well, we're up to maybe 1973 or 1974. Projects came and went, but we had a steady flow of support, either from DARPA or from the National Science Foundation, to do work having to do with understanding language in one form or another. At one point, one of the DARPA projects, and the Department of the Navy may have been in on it too, or maybe it was just a Navy project picked out by DARPA, but we created a system to ask questions of a database of naval information. And it was a big relational database, had dozens of tables in it, information about all the ships in the ocean because we were following all of them. And for our ships and the Soviet ships, if they were military ships, we understood the armament, we understood cargo capacities, and we understood speeds. You know, all this kind of information was maintained in those databases. And instead of using a really theoretical approach to language, David Waltz came up with an idea. David Waltz was at the University of Illinois in Urbana-Champaign. He came up with the notion of a semantic grammar. And with just a few simple things, he was able to answer questions. You know, he couldn't do a very sophisticated system, but he could do with just a few parts-- with ten percent of the parts, he got about eighty percent of the way to what we'd been able to do.

So I jumped on this. I thought, "Waltz is onto something here." And I was sort of skeptical about it, so I came in and worked one weekend and on the basis of a sketchy paper that he had written, I created a system that would do what his system did. And I also that weekend had the inspiration for how elliptical sentences could be understood in context. An elliptical sentence is a sentence from which something has been elided. It's flown off, struck out. So you're asked, "What's the speed of the *Enterprise*?" And you come up with an answer. And then you're asked, "Of the *Vinsen*?" Well, it's very much like the previous sentence, but people have left out parts. So it turned out that an algorithm that I came up with to mimic what had been done at Urbana-Champaign also lent itself to being able to do these elided sentences. And so we had this project in place to query the naval databases and had a lot of people working on this. And I sort of came up with a skunk work project to see if by using these kinds of techniques we couldn't make a lot of headway towards the bigger system. And it turned out that very quickly we got excellent results.

These were the so-called semantic grammars. Dave Waltz at the University of Illinois Urbana-Champaign had come up with this idea. And I'm always glad to steal a good idea from someone and Dave was a colleague and a heck of a nice guy so I think he was somewhat flattered that we picked up his stuff and ran with it, but we really, really ran. And we applied that to this database of naval information. And there were a bunch of problems: the database had many tables in it and so we would translate into an intermediate language that acted like all the tables had been joined and so you could talk about anything, relating any database field to any database field. Then we would go through a process to convert whatever the question was into an SQL query to go off to the database and retrieve it and then we would show the information either in natural language form or often by producing a table. And it was a sensation at the time because it was very practical. It would answer questions about this big complicated database.

Now these days, we wouldn't care and the reason is that there are such good tools with graphical user interfaces, people could go in and they could pick the tables and they can make a relation between the two and ask for tables to be generated. But that wasn't the case in the early 1970s. The only way to get a question answered was to fill out a requisition with the kind of information that you wanted, give it to some programmers, and days, weeks, months later, they'd get around to writing some kind of a program that would pull this information out. This new technology made it possible for the brass to get in there and ask questions and get an immediate answer. Well, if you're in a war, you need an immediate answer. So that's why the military wanted it and that's why it was a big deal.

Spicer: I was just going to ask you that. When you say the generals, you're speaking sort of metaphorically, they're still specialists presumably that would ask the questions?

Hendrix: Oh, sure.

Spicer: But the idea is that people don't have to learn SQL or some other arcane interaction language.

Hendrix: That's right.

Spicer: So that was the main logic behind the military funding for that, I suppose.

Hendrix: That's right.

Spicer: And I noticed in a paper that you published in ACM, I think in about 1980, where it talks about databases and natural language interrogation of databases, data abstraction and so on. Am I jumping too far ahead to touch on that?

Hendrix: No. No, the LADDER system--that's what that first one was called--we handcrafted the grammar. It was a semantic grammar. That means that instead of talking about syntactic classes like noun and verb, article, preposition, we talked about things like ship type or ocean. And these then would expand into particular kinds of things or there might be some kind of a phrase. So instead of having a general prepositional phrase, there might be some syntactic type that was called *ship restriction* that would be a restriction based on some kind of attribute of ships that would help you narrow it down. And so by taking these things that were semantic concepts and moving them into the syntax, we eliminated so much ambiguity and we made it very specific, but the whole grammar was written with that one particular database in mind.

Generalizing Semantic Grammars

So the next step then was to say, "Okay, we can't afford to do this for every database. Let's come up with some kind of way to get this information very quickly." And there were a couple of possibilities. One was to have some kind of a chart that an expert would fill out to do it and the other was to go in and examine the schema of the database and use the schema of the database itself to populate the initial vocabulary and then ask questions. So let me give you just one example of this kind of thing. You've got a database and you discover that the database has a numeric field in it (which you know there because the field has a specific data type of *number* or you go in and you look and every value that's in that field is a number).

So let's take something like speed. So you've got a ship and it's got speed and it's got numbers in it. So those numbers are probably represented in some kind of unit, but you don't know what the unit is because it's not in the database schema. But you can ask the question, when a ship has a speed of ten, the unit is ten what is it? It's *knots*, right? So you've got to get the word *knots* out. And so that's a way to prompt people to tell you something about it. If one ship has a speed that is greater than the speed of another ship, then the first ship is blank than the other? *Faster*, okay.

Spicer: That's like elementary school, almost.

Hendrix: It is. And so we would get these adjectives that were associated with the numeric fields. And you'd get synonyms for things. Most of the time, when you have values in a field, you can use those as modifiers, so if you have a list of employees and you've got the states that they're from, you can talk about the California employees, so it's like an adjective, but it's not really an adjective. It's a modifier. So you get those. But let's suppose you had sex and so you've got *male* and *female* in there. So you could talk about the male employees and the female employees, but you also want to talk about the men and the women. So a male employee is called a, you know [gestures for 'blank'], a *man*. And you can begin to pull those out. And so we had a project following LADDER called TEAM. I can't remember what TEAM stood for, but it had to do with being able to teach it. So I'm sure that the T stood for "teachable." And we could give it a new schema and it would begin to apply the semantic grammars then to the new area. It wasn't as good as LADDER because in LADDER the whole semantic grammar was designed for the particular set of data. But with this other approach, we could get sixty percent of the way there.

Spicer: LADDER was for the military?

Hendrix: Yes. It was the one for the naval ships.

Spicer: Okay, thank you. Go ahead.

Hendrix: Well, I was trying to remember what LADDER stood for.

Spicer: Okay. We're almost getting to the start of Symantec.

Hendrix: Right.

Spicer: Is there anything in between there that you need to talk about?

Hendrix: Well, the semantic grammars made it possible to begin to think about commercialization and the teachability of it meant that instead of having to have experts set it up, it could be done more or less automatically, at least I thought so. So we very quickly went from having nothing to having something that a group of AI experts could put together that would really work with a particular database, to then having something that people with less expertise could set up. And then with the teachability, why you didn't need much expertise. So it began to look commercializable.

Advances In Artificial Intelligence

Now, there were other things going on at the same time. There were two other currents that I was aware of. One was that other parts of AI were beginning to work also. This system called MYCIN that could diagnose diseases of the blood was the first of the expert systems and it was- it just made shockwaves when it came out. And other expert systems followed. There was one built in our laboratory and I contributed some of the code that helped build the semantics of what was going on, the representations of it -- and it was called PROSPECTOR. I think Peter Hart was in charge of that project. And it discovered a molybdenum mine up in Canada that was worth a billion dollars or more so it paid for itself pretty well. I mean it was a big deal.

Spicer: Just by someone asking certain questions it came up with this? It sounds like data mining almost.

Hendrix: Well, it was sort of mining-related alright. What it did, if you talked about what you found at a particular area, it could ask you questions to help talk about the rocks that you saw at the surface. I wasn't intimately involved in this at all. I just provided a little piece of technology that these other guys used with it. But someone over there had the idea that okay, we've got information about this stuff over a whole grid of area, so they'd run the expert system over and over again in all the positions in the grid and then they'd get sort of a view -- and I think that was the first time that an expert system had ever been run in grid form. And then they were able to find things. At least that's my recollection of it. I'm not the person to ask about that system, but I can tell you that it was a big influence on me because I could see that it was working and it was making money and that it discovered this mine. Also, the robot vision work at SRI and other places was beginning to pay off.

Spicer: Shakey?

Hendrix: Yes, Shakey the robot had been built and had come and gone by the time I arrived, but Shakey was a big deal. And it was Shakey's planner that got written up and I wrote a paper while I was at Texas about an extension because Shakey's view of the world was that time moved in increments. There was this point and then this point and then this point and this point, but there was no real concept of time. And a paper that I wrote at the University of Texas talked about time being continuous. I had training in mathematics as an undergraduate and it was a real differential equation with this kind of guy. And so I wanted to talk about being able to turn on a water tap and its filling up a bucket and you can tell the depth of the bucket and when the next critical point in time was going to be when something interesting would happen like the water would overflow. So Shakey happened and then I wrote this little paper and that's how I got my foot in the door.

But all the wonderful work on Shakey had been done by the time I got there. But that was the pioneering work to have the vision and the robot planning. And then that began to be more practical as people began to think about what can we do with this in a factory? It turns out that when stuff is forged, in an awful lot of manufacturing processes, you're making a bunch of parts and there's a mold that's built and all the different parts that are going to go into an engine are all in there together and then they get knocked out of the mold and then they're just thrown in there at random. Well, if you could put them in a jig, you can clean them up and turn them into a useful part. And robot arms were, by that time, good enough so you could instruct them and they were flexible enough to reach out and pick something up. But well, where's the part and which part is which? So it was a big deal to be able to work on a vision system that would look at parts coming down an assembly line, figure out which part was which, where it was, its orientation, and give the instructions to the robot arm to put it in a jig. And SRI International had a group that was working on that. Marty Tenenbaum had the general vision group and I think Jerry Gleason was working on a particular one for industrial automation and the whole thing was under the direction of Charlie Rosen who was the guy who started the Artificial Intelligence Center in the first place. So it was in the wind that these things were going to happen. We'll come back to this theme in a minute, but AI was looking like it was about to take off.

Microcomputers

The other thread here was that homebrew computers were happening where we were seeing the beginning of the microcomputer industry. One of the guys brought in a machine that he had purchased. He'd spent a bunch of money on it and it was a little box. And he was going to have this machine at home. And it struck me as sort of silly. I wasn't going to spend my money on this. I mean it was big money. It was like maybe ten, fifteen thousand dollars he'd spent on this thing, about six months' salary at the time, and it had only a few hundred bytes of memory.

Spicer:	Was it a minicomputer or a microcomputer?

Hendrix:	It might have been a mini, I don't remember. But this was pretty early, so it could have been. I don't recall. What I do recall is how happy this fellow was and how crazy I thought he was to spend his money on this. But he was going to be able to do computation at home. And I said, "Well how big is the memory on this?" And it was some tiny thing, but he said that he was going to have some fun playing with it. And then we started seeing more of them. And in that era, it dawned on me that we were working on this natural language stuff for people at the Pentagon basically, but they had experts who could get the information out. There seemed to be a wave of these little machines happening and if anybody needed natural language, it was the people that were going to be having the smaller pieces of equipment because they weren't going to be able to afford a hundred thousand dollar a year DBA [Data Base Administrator] (or whatever the equivalent was then -- I'm sure it was a lot less) to put questions into it.

All right, so those chords were going on. And I wasn't the only one at SRI to observe that AI looked like it was about to get hot. Charlie Rosen and Earl Sacerdoti also saw this. Charlie had started the center and he was head of the industrial automation projects. Peter Hart was at that point doing the expert systems and he was either the director or the associate director of the center at that point. Earl Sacerdoti was probably the associate under Peter, associate director of the whole center. There were about sixty scientists. And Nils Nilsson who had done the robot planning, he was a big part of Shakey's system. Dick Duda who'd worked on robotics and expert systems, Jerry Gleason who was the vision person. Marty Tenenbaum and others. I'm sure I'm leaving somebody out. Those people were interested in figuring out how AI could be exploited and it was basically the management group inside of the Artificial Intelligence Center and I was the manager of the natural language group by that point.

<u>Starting an AI-Based Business</u>

So there were a couple of hall conversations about, "We really ought to talk about what we could do. Wouldn't it be nice if we could increase our income by ten percent or something just to have a little extra money?" So we started meeting over at my house on Tuesday nights and this went on for a long time. We were very worried about how we could produce something without being in conflict with SRI or with the Government. I guess we were real egotists because we wanted to make sure that if we departed, we wouldn't destroy the community at Stanford Research Institute, or maybe the whole of the AI community. What egotists we were. So we talked for a long time about those kinds of issues and we started keeping notebooks about possible ideas. Naturally there was the natural language interface to databases, there was the machine vision system, but we also had crazy ideas like the Roball.

The Roball was a ball with a little electrically-controlled car inside of it, but it would have heat sensors in it and vision systems and so you could play with it. It was just like a smart ball. So we thought that this would be a fun thing to have. And between Roball and the vision systems there were all kinds of other crazy ideas. And doing expert systems of course was a very real and viable thing to try to do. So we talked and talked and talked about this. We started talking in 1977. My wife kicked us out of the house sometime before my daughter was born in 1978 because she couldn't stand Charlie Rosen's, for fear it was going to harm our child. So we moved over to Charlie's house since he had the only wife who would let us come. And Charlie was a wonderful entrepreneur. He had started the Artificial Intelligence Center. He was a real business entrepreneur. He had started Ridge Vineyards. I don't know if you've had Ridge wines, but they're really, really great. And so he and four other guys from SRI bought some raw land out here above Cupertino and it ended up that it had grapes on it and one of the scientists he bought it with was a chemist, so they figured out they could make wine and they sort of got into it as a result.

So he had that enterprise experience and he also had a really good wine cellar. So once we got over to Charlie's place, things began to move along a little more rapidly. And Charlie was the oldest guy by quite a bit and he was close to retirement age. And when it looked like he would actually retire, he figured he could afford to quit and be president of a new enterprise. And so we formed a company called Machine Intelligence Corporation there in Charlie's living room. And we didn't want to offend anybody and we didn't want competition from the Stanford AI people, so we ended up inviting fifteen people to be cofounders with us so we had basically the whole management group out of SRI International plus many of the key players on the faculty in AI at Stanford University, fifteen of us. We each put up five thousand dollars. We had a whopping seventy-five thousand dollars. And we needed a first project so the thing that looked like it was going to be doable in the least amount of time was robot vision, or as I call it, the robot eyeball business. And this was Jerry Gleason's project. And we hired someone to work in Charlie Rosen's garage. I've forgotten what this fellow's name was. He was a delightful young man. His father was a Nobel Prize winner and worked over at SLAC.

Spicer: Ted Panofsky?

Hendrix: Ted Panofsky is the name of this young man. And Ted Panofsky is a smart fellow and he sat there in Charlie's garage with a soldering iron and worked on the robot vision system. Meanwhile, we were out trying to interview people to get ourselves a real CEO and we were talking to anybody who would visit with us about how venture capital was raised, how companies were started because we all had academic backgrounds and academia is not the place to learn business. Maybe in a business school, but certainly not in a computer science department or a physics department or the electrical engineering kind of places where we had come from. And we didn't know anything about business. We didn't know about balance sheets, we didn't know about P&L, we didn't know about sales or marketing, we didn't know squat. But we did know that there was a lot that we didn't know. But one of the big things that we didn't know was how to interview people or how to find the expertise that we needed and it was just all very difficult.

The First Project

One night after we had incorporated and we had a capitalization there of seventy-five thousand dollars, it was in February. What year would this have been? It must have been February of 1979. We went to one of our meetings at Charlie's house and this was right after we had incorporated. And Charlie was all excited because one of his friends at the National Science

Foundation had sent him a solicitation for a new program that they had called the Small Business Innovative Research Program. And NSF had tried this the year before with some success and then I think this was the second year of the program, but it was just getting off the ground. A guy named Roland Tibbetts was the fellow that had really pushed this. Roland is a heck of a wonderful guy and a big, big backer of doing fundamental research in small companies with the idea of fundamentally helping the country.

So, the solicitation was for research from small companies, and we certainly qualified for that, and you could get a whopping twenty-five thousand dollars in phase one money if you could do something interesting. And then their carrot was that if you did well and you won the competition, in the next round you could get a quarter of a million dollars. So this was Tuesday night, because all our meetings were on Tuesday night, and the proposals were due in Washington by ten a.m. on Friday. So Earl Sacerdoti and I were ill the next two days, couldn't make it to work, and we set up shop in my living room and we wrote a very, very nice proposal and we got it into the FedEx. These were in the days when you had to get to FedEx by noon or something to get it out that day. So it was very difficult to get this thing to Washington on time. But we had a completely filled out proposal. We certainly knew a lot more about writing proposals for the Government than the average guy on the street because this is what we did for a living at SRI. And by golly, we got funded. And so that twenty-five thousand dollars was for using natural language processing to help in the manufacturing process because the request for proposals was for things to help in manufacturing. And as you know, you write to what the customer wants. But NSF is really good. If you've got a good idea and you put it in under one thing and they see that it's a good idea, they really try to fund it. I mean National Science Foundation is peer-reviewed by scientists and scientists in general want to advance the state of the art. And when they see something good, well I think they're pretty good about it. I'm a big fan of government-backed research and I think it's done well for the country and I know it's done well for me.

And the projects that got funded at SRI I think were all very good and useful projects and good things happened. At any rate, we had our twenty-five thousand dollars. And so we worked on that and wrote a nice little system that ran on the Apple II and by golly it would parse sentences and it would come out with the types of structures that we would need to go to a database. We didn't have the database with it at that phase, but we were able to parse sentences and we could do them fairly rapidly. And this was sort of a miracle because we had been requiring a PDP 10 with a megabyte of RAM to do this and we were able to parse sentences in I think it was a 48K [] Apple II with 48KB of RAM memory. And the thing was written in Apple's Pascal and it wasn't very big, but by golly, it worked.

Spicer: I should ask you what some of your previous efforts were written in. I meant to ask that. What programming language generally was it?

Hendrix: This was all in LISP. LISP was the language for artificial intelligence right from when John McCarthy invented it. And I was negligent when we talked about what was going on in the 1970s in natural language processing that the big thing was Terry Winograd's SHRDLU system and that was the one to beat. And it was just an excellent system. Of course, Terry then came out to Stanford and I don't know if he's still a professor here or not, but he was for many years and probably still is a great asset to the University. So that was a really great program. And Bill Woods also had a system and the name of it escapes me at the moment, but it was an excellent system too. So it wasn't like SRI International was the only place doing this. Other people were doing really, really great things with Winograd's system. It was head and shoulders above everybody else in terms of what had been achieved to date. So what thread were we on?

Spicer: The Apple II running under Pascal.

Hendrix: Right. So this thing worked and we wrote up our final report and we sent it in and we put in our proposal for the other quarter of a million dollars. So this was around the end of 1979, in the fall, that we were working on it. We put in the initial proposal in February and we probably got funded six months later. We did the work, we sent it in, and we asked for money. But the government grinds very, very slowly and I think about a year went by before we got the follow on, so it was a good thing I kept my day job. I kept working at SRI and I think during that period was when we did the most on this thing called the TEAM project when we were trying to figure out how to acquire the schemas. And I was a little bit devious with the government I must admit because while this was a great thing for the Department of Defense, at least we thought it was going to be--I don't know if they ever made much use of this technology--but if other things hadn't happened then, this would have been a lot more significant for them. But I also had it in the back of my mind that not only was this good for them, but this was the missing piece that was required to make this sort of thing commercializable. So we were sort of serving two ends by working on that project.

Machine Intelligence Corporation

Meanwhile, Machine Intelligence Corporation hired a real CEO, an industry person who had started companies, a guy named Arthur Lash. And Lash went out and raised money for Machine Intelligence Corporation and hired marketing and sales people and got a big building and got all the staff and put the people together to run a real company. And the rest of us, the academic types, we were all just sitting there with our mouths open watching Lash do this.

Spicer: In a good way or a bad way?

Hendrix: Well, in a good way. We were sort of astonished and appalled at the same time. And what he told us was if you want to be a hundred million dollar company you've got to look like a hundred million dollar company and to look like a hundred million dollar company you've got to spend like a hundred million dollar company. And I guess there's some truth to that. And there were a lot of outlays and he talked to his venture capitalists. They knew what he was doing. And the pieces got put in place and it was really neat. And I think sometimes people do that and it works out and sometimes it doesn't. It just sort of depends on-- there's so much random about this and people who are successful have the mistaken idea that they knew what they were doing. Now, it turns out that to be successful you've got to get an awful lot of things right. And if you lack skills, you're going to screw those up. But you can have the skills and there's still all this randomness about it. So I think that Machine Intelligence Corporation had a very good chance at being big, big time. It had great people in it and all, but the bottom line is that it went through thirty million dollars and crashed and burned. And when we talk about a hundred million dollar company, that may not sound like all that much these days, although a hundred million here and there, it adds up. But that was sort of the goal in the1970s. If you can get to a hundred million dollars, you're a huge success. So that was the talk at the time. And Lash was a good fellow. He was knowledgeable, he knew how to do things, and he knew how to handle people well. And he put in all these missing pieces. Getting that one guy in there brought in all these types of expertise which the rest of us didn't have. And he made a business out of what was just an academic exercise. The business ultimately failed, but I certainly don't blame Lash for that. And what I do is I give Lash the credit for giving us a chance at it. It was great.

So it was one of those things. Okay, so Lash is in and the money comes in from National Science Foundation saying, "We'll give you a quarter of a million dollars to go make this happen." So I get the news about this in late spring or early summer of 1981 and at that point, it's time for me to leave SRI. So I resigned at the end of July 1981 and went to work full time at Machine Intelligence Corporation and began to work in earnest to get all this running with the database and another more powerful type of grammar running on the Apples.

And within a few months, Machine Intelligence decided that it really needed a focus because our original idea as a bunch of academics coming out of Stanford University and Stanford Research Institute was that we were going to cover all of artificial intelligence. Well, Lash had the good sense to know that that wasn't going to happen. And so he said, "Look, guys, we need to focus on something. We're going to focus on the vision part of it and we're going to partner with somebody who already has robots." And there was a company in Japan that had a really good start on robots. So they partnered with the Japanese company and they put the vision system with it and they had some very, very nice demonstrations and technology. I think it ultimately failed because it cost more to put scientists into an assembly line to set up the equipment than the equipment cost and that was the problem. It was too hard to deploy the technology so that was the end of Machine Intelligence.

Spicer: Do you remember the name of the Japanese company?

Hendrix: It began with a K.

Spicer: Kawasaki?

Hendrix: It could have been Kawasaki. I don't know, but Earl Sacerdoti would know. There are a bunch of people and I can find that out. So Art Lash had a little problem with me because there was this natural language work going on inside of Machine Intelligence Corporation and he was trying to focus and our stuff wasn't any good for vision. So he told me that he was planning to sell me to HP.

Spicer: Sell you? Your group?

Starting Symantec

Hendrix: Yes, this group, this whole project. And I told him well he could sell something to HP, but I was just going to go back to SRI International and try again to get out on my own because I was bitten by the entrepreneurial bug in one way or another. In his company or in my own company, I was going to do this. And Art came through. He was a pragmatic guy. He wanted to make things happen and he had enough venture capital money to do it, but he wasn't greedy. He was an enabling kind of person. And so when he heard that he wasn't mad at me or anything. He said, "You know, let's find a way to make this work." And he came up with the idea to make it work which was a tremendous gift to me. I mean it was my project and it was my technology. I knew how to do it, but I needed a start. And he said, "Look, we'll make a deal with you, Gary. We're going to spin you out as a separate company and here's what we'll do. We'll keep forty percent of the stock and we'll give you sixty percent. And we'll give you the project. That is, we'll transfer it. And we'll give you a hundred thousand dollars in cash to spend and we'll give you free rent in our building and we'll provide services like accounting and Xerox machine and we'll give you use of our computer for a year."

The condition was that they got forty percent and the other condition was that we would dilute ourselves mutually to provide stock options to get the rest of the people. And he certainly didn't want me running the place. We needed to have him and his group pick someone to be the CEO of the company. Since I didn't know how to run a company, I was delighted to agree to that. And of course I knew because I knew Art Lash and trusted him that he wouldn't inflict someone who wasn't going to work well with us. And so we then went on the enterprise of trying to interview folks who would be suitable leaders for this enterprise. I had that going on at the same time. But it turned out it's a big deal starting a company even if you've got all that help. I mean you want to write a letter. Well, where's your letterhead? Somebody's got to go design it and get it printed. And Machine Intelligence was of some help to me in doing those things, but mostly I was a nuisance to them because they had their own things that they needed to focus on. So while they gave me the space in their building that they might need anyway eventually so they had to overbuy. Most of the things that Art gave me were things that weren't going to cost him. But they were of real value to me.

Spicer: What happened next?

Hendrix: Steve Shanck, who had run Apple's sales and marketing in Japan, became CEO of Symantec. He was lots of fun to take to Japanese restaurants because he could speak a little Japanese and he could do origami on dollar bills and was generally a nice guy, easy to get along with. And he had some interest in changing the culture because basically I had gone out and I hired Ph.D. types to staff my organization. And I should mention a few of those. Norman Haas was our first employee. Norm didn't have a Ph.D., but he had a Masters degree from Stanford and had been active in this area for quite a while and had been one of my right-hand men in terms of implementation at SRI. And Ann Robinson came over soon after and she was another SRI International person who had been in the natural language group since before I got there. And she ultimately was put in charge of the natural language work for the project that we were working on. And Francisco Corella did the database things. I think he had a brand new Ph.D. out of Stanford and had worked in the database area with Gio Wiederhold. And Gio was a good pal of ours. Gio is the only person who ever invited me to a house cooling party.

Spicer: What's that?

Hendrix: Well, if you're moving out of a house then you invite people over for a cooling party and you drink up all the booze so that you don't have to move it to the new house. Violetta Cavalli-Sforza was also a database person and she came out of Stanford and Dan Lynch out of SRI International and Jonathan King and David Levy. We had some really, really good, good people. And most of these guys went on to start companies of their own or do other wonderful things so it was a good group of people. It probably wasn't the right collection of people to create a product, but it was one of the best research groups that I'd ever seen assembled then. We'll get to the problems with that in a minute.

Well, I guess the basic problem was that we overstretched. Instead of just doing the natural language, we decided to build a new kind of database. And we could have done a flat file system, but that would have been way too easy. We could have done a relational database, but a lot of people had relational databases so we stretched and we were going to do some thing that was like an object-oriented database about two decades too early. And we tried to do too many new things simultaneously and it all slipped. So the Dictaphone thing basically didn't happen.

Oh, I know one player who was important that I want to mention and someone else who went on to do some really wonderful things is Jerrold Kaplan. Jerry Kaplan never worked for Symantec, but early on-- I had told you that with some of this NSF work we had built interfaces, but we didn't build the database part. We hired Jerry as a consultant and he put together for us an in-RAM database in a design that he and I worked out. Well, we sketched it on the back of an envelope and then he went and worked out all the details so it was more his than mine, but the concept had been to my specification. The idea was that instead of having records where you had all the information, it would have pointers in it like things in LISP. If you had the same string repeated over and over again, you'd just refer to that string with a single pointer. So if you had a bunch of people and their state was California, you'd enter California once and then there was a number that represented California and that's what actually appeared in the record. And similarly with everything else. If you had a title of a book it would get stored once. If you misspelled California, you could change the spelling in one place and they would all change. If you only wanted to change it in one place then you'd have to say, "I didn't get it right and so in this one particular instance I'm creating a new number." And of course you didn't tell people that. There was an interface for it. So this database kept an awful lot of information in RAM and it was so efficient because of its pointer structures that we could put as much in RAM as a lot of people could put on a floppy disk. So we really didn't need to use floppies, we could put it all into memory and run it that way. And this had all kinds of advantages in terms of speed and efficiency. That, combined with some of our earlier attempts at a natural language interface, formed the basis of a great demo. The whole thing ran on the Apple.

Spicer: Did this product have a name yet?

Hendrix: Well, at this point, I am confused about what these things were called. We either had a prototype or a spec for a thing called "The Wiz," which people called "The Whiz" after a while, of course. And then there was "Straight Talk," which was the Dictaphone version. So those are the two names that I remember being used. And then there was whatever got built for the National Science Foundation and I don't know that that ever had a name. At any rate, we had Steve Shanck as CEO and we had Arthur Lash in the background helping us out along with his venture capital backers including the Hillmans and Hambrecht and Quist and several other people that backed him. But I think Henry Hillman and his crew were by far the biggest backers.

I remember how impressed I was one day when Mr. Hillman came to visit MIC and I had never heard that anyone had a billion dollars before. And I was told that he had a billion dollars and that I was to go in to meet him. And I was rather in awe of this figure. And he was immaculately dressed and had wonderful manners and was just exactly what you would picture a billionaire to be like. You're probably too young to remember, but when I was a boy growing up there was a TV show called "The Millionaire." Now there was a man, who gave away a million dollars to people, but he himself was a billionaire and they showed pictures of him. And so that was the image that I had in my mind and Henry Hillman seemed to fit that pretty well.

Financing the Company

But he was very gracious to me and I was glad to make his acquaintance and very pleased that someone had the guts to fund a company like Machine Intelligence Corporation. So with Art Lash and his experience, he was helping Steve Shanck and me try to get this new company moving along. And he suggested that what we needed to do next was go to the American Electronics Association's financial conference in Monterey. This would have been the conference in May of 1983. So we went to the financial conference and we had our little spiel about the company. And there was a signup sheet for different sessions. And not too many people signed up for our first session, but we gave this demo of our product running on the Apple. And the demonstration was rather convincing because we would do something very similar to what I had just told you about. You know, you'd tell it about some objects and then you would ask some questions and suddenly reports started coming out. And people would sit there with their mouths open. They couldn't believe it. And so they rushed out and they told their friends and then the thing was completely booked. One of the artifacts I have is the posted signup sheet. Each company put up a little sheet and the venture capitalists had a set of peel-off stickers. And the way you would enroll to go to one of the sessions was you'd peel off your sticker and you put it in a time slot. And I've still got that because it was a huge turning point for us.

There ended up being an absolute feeding frenzy over this technology and I wasn't prepared for this at all. I thought we would go and we would try and maybe somebody might be interested and we might be able to get a half million dollars to keep going and add that to the money that Dictaphone had talked about giving us. But what happened was that the venture capitalists all got to talking to each other and they talked it up and then we started believing the hype that they had about us. And we ended up taking – I can't remember whether it was three or three and a half million dollars, but a lot more than what I had been thinking about going in.

And I think it was too bad that we took as much money as we did because if you have a lot of money then you're expected to do a whole lot and you're expected to do it very quickly. And there are types of endeavors, and science is one of them, and this was still a science project as well as a development project where if you try to do science too quickly, all you do is spend money. And so that was a problem. The person who was most enthusiastic about the whole deal was John Doerr out of Kleiner Perkins. And Kleiner Perkins decided that they would like to take the whole placement. And we had some debates about whether we should take their money or whether we should divide up the stock among multiple groups.

But Art Lash told us that if we took it all from them then they would have this unusually large stake in our company and that we would receive a great deal of attention from them and that in the long term their attention could be very beneficial to us because they knew people and they had a lot of experience and in his opinion they were the best venture capital group in the country and it wasn't because of all the money that they had. They certainly had money, but they had good people, they had expertise, and they were a genuine help to the companies that they invested with. So Art had never given us any bad advice before so we were happy to follow that. And, sure enough, John Doerr joined our board and Tom Perkins did. And so we were really pleased to have that happen. And John stayed on the board for years and years and years and was always making good contributions. And he saved the company on a couple of occasions and we'll get to those.

And Kleiner Perkins, when they came on board, also looked at what we were doing and they thought we were weak in some areas and that we needed to have some more marketing folks. And they had just backed Lotus and when Vern Rayburn became available they wanted us to recruit Vern, which we did. And Vern was fresh off of his win at Lotus and he had also had a great deal of experience with other companies. And this interested me and with him we were suddenly industry insiders instead of a bunch of academics on the outside. So now we had an Apple connection through Steve Shanck and we had an IBM and spreadsheet connection and just general broad software connection through Vern.

<u>Delivering a Product</u>

After working a little while longer, it became clear though that our team of Ph.D.s was not going to deliver a product. It took a while to figure this out. I guess we were at it for another six months or so. And what happened was that we would keep saying we were going to deliver something on a particular date and then we wouldn't make it. And our new estimate for how long it was going to take to deliver it was actually then more time than it was from our original one. And after we had done this a couple of times, John Doerr picked up on it and said that he had seen instances before when there were slips and the slips got bigger and that it was indicative of a research project instead of a development effort and they were going to have to do something about this. So indeed they did. And the upshot of it was that there was a reorganization inside of the company with the idea of bringing in people who were going to be more focused on getting product out and less research-oriented.

I was rather sad about all this because I had handpicked all these developers and I had great confidence in their ability. And I felt like we were making a lot of progress, but we were working in an area that I knew nothing about. Everything that we had done had been in LISP and now we were working in Pascal and we had a sorting algorithm. And I knew that our sort algorithm, from a theoretical point of view, was the fastest kind of sort that we could do. So I ran a sort and it was taking a minute or two to sort records. John brought in Lotus 1-2-3 which had just been released, and did a sort. Want to see it again? It was almost instantaneous. Well, of course they were doing things in RAM and we were doing stuff out on the disk, but still it was a very dramatic difference. And a big part of it was that they had done all their work in machine language and had spent a bunch of time optimizing it. And their developers were people who thought about that. They thought about "How are we going to make this work on the small stuff?" Well, we thought we were doing a great job working in a small area and thought we had made a tremendous concession by using Pascal instead of Lisp.

<u>Reorganizing Symantec</u>

But what was really needed on the PC was to use assembly language. And for my people-- you think about this pyramid. My guys are at the top of the pyramid and are used to standing on all these other layers of systems and software. And what was really needed on these little machines where the pyramid is much smaller is to look at this little pyramid as part of the big one. Well, everybody's got to be on the bottom and it wasn't the right crew to get this done. So there was a great deal of gnashing of teeth and the company could have died at that point. I didn't want it to die. I really, really wanted Symantec to work. I wanted to be the guy who brought this technology to market and I didn't want to leave being the big failure. So we decided we were going to do whatever it took to make this happen. So Vern took over as CEO. Steve Shanck left. I think Steve took the blame for what was really much more my fault than his, but he was out and Vern was in. And most of my Ph.D. types had to go as well.

And at that point, Kleiner Perkins and John Doerr came in again with the idea of bringing in a development team of folks who had been very successful on small computers before on other projects and who were very practical people and who had worked on databases. And so a new team came in and joined up with ours. Their leader was very impressed with the natural language system and also with the database pieces that we had put together and for a time tried very hard to work with my group. But it became clear after a while that this wasn't going to work; he wanted us to get out of Pascal for sure and either go into C, or better yet into assembly language.

Spicer: What was his name?

Hendrix: His name was Robert Rosenthal. And he had been an Intel Fellow. He had been a driving force behind System 2000, which is a big database management system. I mean he wasn't the only person behind System 2000, but he was certainly a major contributor over there. And he had a little team of people who worked with him. And he came and made a very concerted effort to try to straighten out my folks and get us on a different path. But we had a tremendous clash of cultures at that point and personalities; but it was more than personality. It was philosophy and world view. And I think we liked each other well enough as people, but the world views were not going to work. And it wasn't just me. If it had been only me I think I could have gotten along with him perfectly well. But with the academics that I had brought in, it was oil and water and it simply wasn't going to work. And what basically happened was that my academic types left and the kinds of folks that he had couldn't produce the same kind of systems. They could produce systems that ran on small computers which my folks couldn't do, but they couldn't produce the magic. So in time, it became clear that that wasn't going to work either. And it was sort of a sad thing. He and his group decided to leave and go on and do something else. And when they departed I was pretty well convinced that Symantec was over.

And it was interesting. The day that this happened, I had been invited to give a keynote speech for one of the linguistics groups. It was probably ACL, Association for Computational Linguistics, I'm not sure that was it, but it was one of these organizations. And they were meeting at Stanford University and I was supposed to be over there. And instead of being over there giving my talk, I was in the middle of dealing with the dissolution of my company when the folks over at the university were expecting me to come over and tell them uplifting news about how wonderful things were and how we were just about to crack this problem. So it was desperate times and what were we going to do?

Well, I felt like we had eliminated one obstacle. Somebody had to go. I mean I could have gone and they could have probably produced some kind of software that could have sold okay in the microcomputer market. And they might have been just as successful as Symantec. I don't know. But they were the ones to go instead of me. And so if I had left, they would have been sitting there with just the ability to do engineering stuff. As it was, they left and I was just sitting there with the ability to do sizzle on big machines, but no hands to make the thing work. And by this point, I had lost all my Ph.D. types. Because I always had in the back of my mind that if we couldn't make it in the PC market, well we'd just go and we'd work on mainframes which were still a big deal and we'd do natural language interfaces on those. That was a viable option. But by this point, we had run through too much money and now we'd basically run through two sets of engineers.

Restarting Symantec

And John Doerr and the rest of the Kleiner Perkins people were pretty aggravated with me because we'd burned up their money and we'd burned up their set of engineering talent that they had provided. I think they were a little surprised that I didn't want to give up, but I was clear that I wanted to continue if we possibly could and that if we could find folks that could actually work together, we would very much like to do that. And I had some darn good people. It wasn't just that we knew about the technological sizzle of natural language processing, but we also had some very fine product marketing folks who had joined the company by then and had a vision about how the technology could be used. Brett Walter in particular made a huge impact on the company, but a man named Jim Chandlen did the best job of recruiting people that I've ever seen. On a single day, he had three product marketing/product management type people start: Brett Walter, Tom Banks and Ken Hess. And they all showed up on the same day and were sort of in a little learning group together about what was going on and then they all went on to make major contributions to the company.

My idea had been to do this funny type of database management system, to have a report generator with it, a graphics generator with it, a forms system that would go with it. Brett brought along the idea of doing a word processor also because if you were running on a small machine what you want to do with a database mostly is mailing lists and things like that so a word processor integrated with it would be good. And I think we were beginning to think in the direction of having a word processor and he had designed a word processor, not the inside of it, but the UI for it, for the Bank of America. And so having his expertise in the design of a word processor was a big deal. And we had seen what he had done in the user interface design for this and thought it was brilliantly done and thought that he could probably apply this to other parts of the product as well.

So we may have had problems with the technologists being too academic, but the product marketing people that were there were extremely good and they could design; they could figure out the right feature sets, the right messages, the right kinds of mix of things to produce a winning product in this market. We had Vern Rayburn who had all this experience, not so much in product marketing, but in marketing and in PR. Dottie Hall was there who had done marketing communications for Lotus and had been also at Microsoft back in the early days; she was also very well connected to people and knew how to get the message out. So we were very strong in that area. We were very strong in the sizzle area and now we had no engineering.

But it turned out that at about the same time, other people were trying to get going and they had sort of the opposite problem that we had and that is C&E software. The C is Denis Coleman and the E is Gordon Eubanks. And they had left DRI (Digital Research Inc.), or used to be Intergalactic Digital Research, and this was Gary Kildall's thing. And then Gary Kildall and Gordon were very much two potentates over there for a long time. Gordon had a lot to do with that company. And I think Coleman had a Ph.D. in business, but he was also interested in low-level software development and was one heck of an assembly language coder and had worked on tools. And so C&E had some really good engineers in Gordon and in Denis Coleman. They had Paul Lancaster, who knew about databases and was a good assembly language and C coder. Tim Binson was over there. I can't remember if Barry Greenstein was with them or not, but those other folks were.

And they had some tools for PC development. They really didn't have a good vision of what they were going to do except that they were thinking of going in and trying to take away the low-end market of the PFS line. So at this point, we were recalibrating, thinking well gee, we've lost all these guys to build the special kind of database. Maybe what we ought to be doing is a flat file as the basis which would be really, really simple and put our natural language on top of that and put a word processor on it. Well, these guys were going to do the PFS series so they had a file system, a word processor, etcetera. And they hadn't started to implement this. They didn't really have a product design in terms of the user interface and stuff. So it was the right kind of mix of people, but we were in a position where we might have had exactly the same kind of problem that we had had with Rosenthal and probably would have except we'd already been through the Rosenthal experience. And we didn't want to do that again. And plus a number of the Ph.D.s were gone so there was less resistance in place, both in terms of just body count, and in terms of the couple of us who were left who wanted to do natural language stuff and were also ready to be very pragmatic by that time. We really wanted to get something done even if we had to eat some crow to do it. And we were ready to learn from some different folks. And on a personal basis, we found it a lot easier to get along with this particular group of people. We seemed to be singing more the same tune. The particular set of products that they were looking at, because they matched up with what our product marketing folks were thinking about, it made for a good match. So the companies were merged in September of 1984.

Spicer: Symantec and…?

Reinventing Symantec

Hendrix: And C&E. So the C&E people and Symantec people were each ready to accept the other in order to make something happen. And this was a marriage that was put together by the venture capitalists. Kleiner Perkins owned a third of the Master's fund. The Master's fund had funded C&E. C&E had stumbled in its own way and we had stumbled in ours and we had the right set of complementary skills. We put those together and we begin to have quite a little bit of success in defining a product. Basically, the product marketing people and I had sketched out something that was a variation on what we had been thinking about before and the whole thing got focused and the pieces got divided up and folks started working. And about a year later, we were ready to ship. Meanwhile, we started to get quite a buzz going because Kleiner Perkins had been involved in this and we had a number of heavy hitters. We had Gordon Eubanks, who had been a major player at DRI, and we had Vern Rayburn. Rod Turner joined us and he had been at Ashton-Tate and had a lot to do with the promotion and sales of dBase II. And we had other luminaries. It was a very nice lineup of people. And so there began to be a little buzz about this. Even though people didn't know what we were doing, they knew that we were doing an AI product and so we began to get some media attention. Not so much attention from people who would be potential customers, but folks in the media; the media were beginning to think about us. And Vern did a wonderful thing in the launch of the product in doing an exclusive so that we would get a cover on the day that we actually announced the product.

Spicer: With which journal?

Hendrix: I think it was "Personal Computing." You'd think I'd know because it was such an important deal to us. One little side story on that, we went over to give them a demonstration of the system and there were various people in the office, but one of them was named was Lee The, spelled T-h-e. And we were putting in some information and he wanted to put himself into the database so we put Lee The into the database. Well, "The" looks an awful lot like "the" and we just weren't prepared for a noun to be confused with an article because there's A and The and that's it for articles. And they're such fundamental things we weren't ready for "The" to be a noun. So we had to change the spelling of his name to T-h-a, which he graciously allowed us to do and we continued with the demo successfully. And then we went back to the laboratory and fixed that problem.

So then there was a launch at the Varsity Theater in downtown Palo Alto which was a wonderful thing and we got a projection TV like you see all the time now, but it was a big deal back then, especially to get one that was in color. So we put on the show and it did pretty well. It was well received.

Q&A

Spicer: And the product was?

Hendrix: It was called Q&A like “Questions and Answers,” because it had this natural language component in it. So the product was built around a file system, a flat file system, with a form system, and a report system. What we called the “intelligent assistant” was the natural language pizzazz that was added to it. Plus there was a word processor built in that could do mail merges. And it had a little programming language in it to enhance the database capabilities. It was extremely easy to use. I mean the whole idea of using English was to make it easy to use, but we carried that concept further, and this was Brent Walters’ design for UI--to try to make everything extremely simple. And I think we were very successful. The natural language sold it as being simple and easy to use and accessible. The interfaces that Brett had designed really were what delivered that ease of use. The natural language helped make it easy to ask complicated questions, but this ease of use was everywhere throughout the product and that’s what made it work. We got a good sell-in initially and again Vern had called up everybody. He did an exclusive distribution deal with Softsell. And the result of the exclusive distribution with Softsell was that all the sales went through there. So the thing that people were watching was the Softsell chart. Well, we had an exclusive with them, so we didn’t dilute our ability to rise on the chart by having any other channel and consequently, we shot up. We released this product in November and by December we were number three on the charts. So it looked great and we got a lot of sell-in. And then we began to look at the sales late in December and in January and they had dried up. So out of the chute we got a lot of sell-in, but it wasn’t really selling through. And we had spent all our money trying to get the thing built.

Spicer: What do you think happened? Who were these initial buyers?

Hendrix: I think the sales were actually fairly good; it’s just that we stuffed the channel and so it was going to take a while for them to sell them. And the word really hadn’t gotten out. We were a new company and we did have a bit of a buzz because several articles came out about us. And in fact, “Software Digest” came out and sent us a letter on January 7th saying that they had by chance just done an evaluation of file management systems and that we had come out with the number one rating. So this was great news, but we still didn’t have many sales and we were virtually out of money. So we were thinking what are we going to have to do? And there were serious decisions in the boardroom about what was to be done. And we were thinking well, we could lay off the development team. They’ve done their job now and we’ll let the sales folks take over. But if you lay off the development team, you can’t come out with version two. If we keep the development team, we can’t cut the sales force because then how are you going to sell it? What are you going to do?

So there was an idea that we came up with called the Six-Pack Program. But before I tell you about that, one thing I forgot was that Rod Turner came up with the idea of sending everybody in the chain that he could think of a little letter that says, “You know, you’re going to start making money with Q&A.” And this was given to the rank and file folks inside of Softsell who were filling orders, it was sent to every computer dealer that we could think of. And it had a dollar in it. “This is your first dollar that you’re going to get,” but it was a way to get people’s attention. And that one dollar bought us a lot of mindshare to get going. So there were a number of things like that that happened; that part of our organization performed very well.

So the Six-Pack Program was to take all the engineers, who don’t need to produce the next version right now, and make them into salespeople. And they’re going to try to go out and cover six stores a day, and be gone for at least six days, try to live on six dollars a meal, and I can’t remember what the other parts of the Six-Pack were, but the idea was to do it on the cheap. And people were supposed to try to go to their hometown if we could cover the country that way and stay at some friend’s house so we could reduce our cost, or sleep on a park bench so that we could keep the expenses under control and we did that. And so we sent our entire development force out, plus accountants, secretaries and other support people.

Now you would think that developers would be really lousy salespeople. And indeed, if they had been doing completely cold calls I think that would be true. But our inside sales force made it their business to line up people to see at stores in these different areas. And so when someone would arrive, they knew what stores that they were supposed to be at and when. And generally people, if they saw a poor developer coming in and he said “I wrote this product. Won’t you let me show it to you on your machine? And by the way, here is “Software Digest” showing us to be the number one product in this area and this just came out. Don’t you want to hear about this?” they would usually melt these people’s hearts and let us show it. And then once we showed it, they were really fascinated by it, especially by the natural language part of it. And they thought that they could use it to help sell machines and if they could sell a machine, which was a lot more valuable than a piece of software, then they really liked that.

So that experience got Q&A off the ground with this big sales force of nerds. There was another very interesting side effect. When we were getting the product ready to go out, there was a constant stream of things that the marketing guys would come up with because of focus groups or just playing with it. They’d say, “Well, we need this feature, we need this feature, we need this feature.” And I’d be trying to put in things that would have to do with better demos, because I was always a very demo-oriented kind of person, to give it a little sizzle. And the developers of course always want to say no to everything because they’ve got to ship it and they’re under big pressure to do so.

Once they went out and they were in the salesperson’s shoes and they saw what a little bit of sizzle did and what the lack of a feature did, they understood that we needed to do another round and these things that looked like fluff, but were actually extremely important in pleasing customers and in pleasing our real customer. Our direct customer was the guy who owned the little software store, the mom and pop software store. Ultimately they’re selling through to someone else, but we've got to sell that first guy first. And if we can sell to him, he's going to go ahead and sell to the next person. So they had had the experience of interacting with these people and consequently in the next go around, instead of not wanting to do these things, there was a different attitude on the part of all the developers to try to get those things to happen. And the next version of the product then was extremely friendly for that audience.

Other Symantec Products

Okay, while we were trying to get Q&A built, we experienced various time delays and already the management team was thinking about that we've got the sales force, we've got these marketing people, we've got contracts with people to build this stuff, we ought to be doing this with more than one product in order to reach economies of scale. So Tom Byers was hired to run a publishing company within Symantec. This was called Turner Hall, Turner for Rod Turner and Hall for Dottie Hall, but really all of it was done by Byers, so go figure. And it just sounded like a good name something that you'd heard before. So, the first product that Symantec shipped was not Q&A at all. It was a product called "Note It" that was written by David Whitney who later played a big role at Symantec in other ways and who I've worked with after Symantec on some projects and is just a wonderful guy.

"Note It" was a utility program that ran with Lotus 1-2-3 that let you put a note on a cell in a spreadsheet so you'd know what the formula was that was associated with it so you'd have some kind of a clue as to what was going on. That product shipped probably in midsummer of 1984. And while we were working on Q&A, we just could not get it to fit in to the 256K [256KM of RAM memory] that came on a standard IBM PC and we were wringing our hands about that. But it turned out that we had this great guy, I think his name was Jim Peterson, who was going to be in charge of doing the product production, hiring that out and making sure we got all the parts right. He was sort of an operations guy. And he says, "Oh, I've got friends who can make a memory board." And there was a new generation of chips out so that we were able to sell a 256K memory board for ninety-nine dollars. Actually, we could produce it for about thirty-five dollars.

And so even before Q&A shipped, we had a hardware product on the market and advertised it as *the best deal in memor*y. And you could buy this by mail-order for ninety-nine dollars and we had very, very good response on that. And then when Q&A shipped, you could either buy it for two hundred and ninety-nine dollars for just the software or for three forty-nine you could get this memory board. And it was a heck of a deal. If you didn't already have 512KB of RAM memory then for fifty dollars you could get the extra memory at one-fourth the price of the then going rate; so it was a real deal. And we sold a lot of those.

And so I'm the software guy trying to do this big product along with a number of other folks. Denis Coleman was in charge of getting the whole thing put together and shipped and Brett was the designer of the entire interface and I'm by this point concentrating entirely on the artificial intelligence piece of it. So there were a whole lot of people working on it. But my point is that while we were doing that, the marketing folks and the ops folks were able to come in and supplement and get these other things going for us that produced money and augmented the business. And they kept at it. Over time, additional products got added that worked with Lotus 1-2-3. You could think of Lotus 1-2-3 as like a completely different operating system. There was the Apple, there was DOS, and then there was 1-2-3. And it was the home for a bunch of stuff. We found out that a lot of people were taking the A1 cell and making it great big and writing letters in there. So we came up with a product that made that easier. It was some third party that came up with this, but we went and sought it out and it got published by Symantec and was called “4 Word.” And there was one called “Squeeze” to take the spreadsheets and squeeze them down to a smaller size. There was the “Cambridge Spreadsheet Analyst” that would look for various kinds of mistakes that you can make in your spreadsheet. And there were a whole series of these.

Spicer: Foreword was like the foreword to a book, f-o-r-e?

Hendrix: “4 Word.” It was 4, the numeral 4 and then “word.” It was like Lotus 1-2-3 and then here’s 4, okay? And the 4 was also f-o-r word processing. Right, so that was that piece. And then other things came along and we sort of got into the Mac business. We had Squeeze and then there was a Mac version of Squeeze which was the first thing that we did on the Macintosh. And that was very successful so then we got SUM--Symantec Utilities for the Mac--and then SAM, Symantec Antivirus for Mac. Macs were really bad about viruses early on because you’d stick a disk in and they would start up and ruin something. So if you put a virus on their disk, they would start it up and they would run it and you were infected. And so we had mechanisms to deal with that. These products were beginning to get us into the utility business. And while that was going on a new version of Q&A was coming along. This was the one that had these features in it that I was talking about that we got from feedback and the cooperation of the developers who had been out trying to be salesmen. The first one shipped in 1985 and the next one shipped in 1986 and we had these utilities starting to come out, or what became the utilities. They were little add-ins.

Acquisitions

1987 was a big year for us because that was the year of massive acquisitions. And this was with the idea of trying to get some growth. There were these behemoths out there in the industry and you would get so much more valuation if you were a public company. To be a public company, you had to have achieved a certain size. So the strategy was to try to get the size up and also to get economies of scale by using our sales force and our accounting department and our HR, but using development groups that were out looking for some way to go public or where the owners wanted some outlet better than what they currently had available to them to get some liquidity out of what they had produced.

So in January of 1987, there was the acquisition of Breakthrough Software in Novato which had *Timeline* project planning software. Later in the year, LVT, Living Videotext, which had outline processing, sort of "thinking" software, that stuff to help people come up with ideas, I love that stuff. They had "GrandView" and they had "*More 2*." And *More 2* was my personal favorite tool for years. And *More 2* was a Mac-based product and we needed expertise on Mac and on graphical user interfaces. And the LVT acquisition was a big thing in helping us get into that arena and gain expertise. Meanwhile, Byers, who had been working on the GEM graphical user interface at DRI tried to get some GEM-type things going and we also tried to get stuff going on Microsoft's primitive Windows systems. And there were various failed attempts, but there was a lot of energy going into trying to get on these other platforms, trying to gain graphical user interface expertise.

And then very late in the year 1987, we acquired Think Technologies, again with the idea of getting closer to Apple. Think had the Think C and Think Pascal and Macintosh Pascal which were the big-time development platforms for the Macintosh. So this gave us entrée into Apple and respect in the community of people who were Mac developers and opened up a lot of doors for us.

The acquisitions were very hard. They were especially hard on the people who were acquired. Again, things like clashes of culture and more just long distance communications and problems like that. We discovered that one of the better things that we could do to sort of help out with this was to send one of our top people to go live with whoever it was. So Rod Turner went to Novato. I didn't have anything to do with trying to get the LVT products out, but I went over and lived in their building just so that there would be a liaison person when problems came up, there would be somebody local to gripe to instead of just "Those guys in Cupertino don't know what they're doing." They could come and dump on me and I could usually solve a problem really, really quickly because it was always something simple. It was just that they didn't know whom to talk to or how to go about getting things done. And that helped. I'm sure if you talked to any of them, it was a terrible time and I was a counterproductive spy or something, but we were trying very hard to get something productive going and it was just difficult when you have one of these shotgun marriages. Tom Byers went out to Massachusetts with the Think group. So we got those mergers under our belt. We spent 1988 trying to really get them digested. And another version of Q&A came out; there was lots of work going on, lots of people coming and going.

Going Public

1989, we had the combined forces of these four companies plus the things that were being published, with our entire line of utilities on top of it. We had enough products and enough revenue to justify a public offering. So it was in the middle of 1989 that the company went public.

And probably what changed the company the most was the next year, on the basis of a high valuation and the fact that we had gone public, we provided a mechanism for Peter Norton's group to get equity without going through the IPO stage themselves since they weren't quite big enough to do it. And that's exactly what happened. We did a merger with Peter Norton that put us very squarely in the utilities business.

Symantec's team, especially the marketing and sales folks were able to almost triple Peter Norton's sales within a year. So it was a big win all around. And it turned out that the creation of the utilities was much more efficient than the big applications and it was just a more efficient economic engine. And so over time, more and more of the company's energies and attention went into the utilities end of things. And that gets us up to about 1991 and I went off to Texas at that point and I don't really have the inside scoop on anything after that.

Spicer: I think we'll stop right there. Gary, thank you very much for being with us today.

END OF INTERVIEW