RDBMS Workshop: Informix

Moderator:
Luanne Johnson

Recorded: June 12, 2007
Mountain View, California

CHM Reference number: X4069.2007

Table of Contents

STARTING RELATIONAL DATABASE SYSTEMS, INC 4

C-ISAM AND ADDITIONAL COMPONENTS 5

DATABASE MANAGEMENT SYSTEMS ON UNIX 8

RDBMS COMPETITION 11

MARKETING THROUGH VARS 14

CORPORATE END USER SALES 16

SQL AS THE STANDARD 16

IMPLEMENTING SQL FOR INFORMIX 19

PRODUCING HIGH PERFORMANCE INFORMIX PRODUCTS 21

DISTRIBUTED DATABASE PRODUCTS 24

UNIX AS A PORTABLE OPERATING SYSTEM 27

INFORMIX ACQUIRES WINGZ 31

GETTING A NEW CEO 33

THE CHANGING COMPETITIVE ENVIRONMENT 34

GOING PUBLIC 36

RDBMS Workshop: Informix

Conducted by Software Industry SIG – Oral History Project

Abstract: The founders and principal pioneers at Informix described the history of the company and the development of its products and markets. The key focus was to support UNIX platform computers with a high performance, easy to use relational database management systems product. Once they were established in this niche they then faced competition from the other RDBMS vendors who each promoted different aspects of their products. Informix principally marketed through VARs so there was always local support for their clients. Benchmarks became a significant competitive element in responding to RFPs, but a lot of these were just could you check the item as being covered rather than how well that function performed. Although they preferred their own query and reporting language, they were forced to switch to SQL in order to be competitive and they were able to exploit a relationship with SAP which did not want to be locked into only interfacing with Oracle. Informix was successful in the distributed database arena and then acquired a number of companies and products including WingZ.

Participants:

Name	Affiliation
Luanne Johnson	Moderator
Nick Baxter	Informix
Roy Harrington	Informix
Robert MacDonald	Informix
Roger Sippl	Informix
Michael Mahoney	Historian, Princeton University

Luanne Johnson: We will begin, in this "breakout" session, to discuss the history of Informix. Roger Sippl will be here in just a minute. Mike Mahoney from Princeton is with us this afternoon. He has a special interest in the fact that Informix was on the UNIX platform. Bob MacDonald was talking, from a marketing perspective, about how, as UNIX evolved, the product was able to evolve as well. What I'm going to do is recap what Roger Sippl said this morning about how Informix got started. He was with Cromemco and came up with a design spec for a database system on the UNIX platform there at Cromemco. And then since Cromemco wasn't in a position to pursue that, he got permission from them to take his design specs and go off and start Informix. I think, Roy [Harrington], you were actually at Cromemco, too, then, is that correct?

Starting Relational Database Systems, Inc

Roy Harrington: Yes, I was with Cromemco; that's where I met Sippl. I was more in the operating system part of stuff. Cromemco was a small company that was really a hardware company, and they made software in order to sell hardware. I was there for three years —and my main thing was to get off of the CP/M [Control Program for Microcomputers] operating system to something a little bit bigger. I worked on a program called CROMIX which was a UNIX-like operating system written for the Z-80 in assembly language. That was my project pretty much the entire time. I knew Sippl there: he was right down the hall and we talked a lot. We got to know each other. He left in 1980, but I had a few more things I wanted to finish at Cromemco so I didn't leave until 1981. At the time I joined in 1981, the company was called RDS, Relational Database Systems, and there were four of us, four full time people.

Johnson: That was you and Roger and Laura King

Harrington: And Bill Hedge. And they had done a little bit of development over the couple of months beforehand but it pretty much started when I got there. First thing we did was C-ISAM [Index-Sequential Access Method] which sort of bootstrapped us into a company. It was something easy to understand.

Johnson: Let's go into that a little bit. I just want to make one comment that the history of this particular sector of the industry is very challenging, because of all the companies' names. Ingres started as Relational Technology Inc. and Informix started as Relational Database Systems and Oracle even had a different name.

Nick Baxter: Yes, Relational Software. So it was RSI, RTI and RDS.

Robert MacDonald: And we all changed our names.

Harrington: We all changed the names to the names of our products.

Johnson: But wasn't that quite a bit later?

MacDonald: When we went public.

Harrington: 1984.

MacDonald: When we went public, the bankers said, "You're known for your product. You should change the name of the company to reflect the product."

Harrington: Everybody called us Informix.

Johnson: Yes. I remember Roger telling a story about one time when he actually received a response to an RFP [Request for Proposal] that Oracle had sent in -- somehow it came to you at Informix.

MacDonald: It got into our hands.

Johnson: Yes, it got into your hands and Roger's saying, "Well, wait a minute, this isn't us." So it was very confusing. So anyway, Informix was started as Relational Database Systems.

Harrington: That's right.

C-ISAM and Additional Components.

Johnson: When Roger was talking this morning about the C-ISAM product, one of the questions that came up was, "When did you first sell your database product?" The actual first product you had was C-ISAM

Roger Sippl: Yes.

Johnson: Talk about that a little bit again for the record here.

Harrington: Well, at the time we had really no customers and Micro Focus COBOL was around. We were associated with them and Santa Cruz Software Operations - another company. And they needed an ISAM package for their COBOL. In order to sell COBOL on UNIX they needed that because UNIX was sort of unusual in that it had no low level record management system of any kind.

Johnson: Now this was in 1980?

Harrington: 1981. So we saw it as an opportunity to create this low level access method to manage the records to be able to satisfy the COBOL guys and to satisfy our needs and get paid for it and help bootstrap us into making money. So that was the first thing we did, and it was actually me and a guy from Microsoft COBOL. We asked, "Well, what do we need in this package?" and then that's what I wrote it -- actually, that was the first thing I wrote.

Johnson: And so then you were selling that. Then you gradually added on other components until it really became a larger product?

Harrington: Yes, the components were designed. As a concept they were there. But it was a matter of implementing them.

Sippl: We had a prototype in the query language -- we had the scanner and the parser all built. In fact, it actually ran. It just ran without an indexed file system and so it ran only on small data sets. It would also work on large data sets but it took a linear scan of the entire files that it was querying. So if the data sets got big it was going to run very slowly. But for data sets of a few hundred records, actually, you hit the carriage return and there was the answer.

Harrington: It was fine.

Sippl: So that all worked. It was just waiting for an ISAM to be hooked up underneath it.

Johnson: What was the query language called?

Sippl: It was called "Informer" and it was based more on the relational algebra than the relational calculus. SQL is more inclined toward what Chris Date described as the relational calculus.

Johnson: And then there was a report writer.

Sippl: Yes, the query language was really just the retrieval syntax that brought data from the database, selected what subset of the database would be retrieved. And then the report writer had a five or tenfold larger set of syntax that was used to format the data into a printed report, but it was still all done with what we called "non-procedural language principles." So instead of using a lot of if-then-else statements or loops or something like that in order to program how you wanted the report formatted, specification languages or non-procedural languages known as 4GL's worked on the principle that the syntax was designed so that you

would specify the desired result. And you would not write the algorithm. So when you first took a programming class they taught you, "Okay, take your problem and resolve it into an algorithm and do a flowchart for it." And the report writing languages, query languages, were the first widely used specification languages where instead of taking your problem and breaking it into an algorithm, you really took your problem and used a syntax designed to solve that problem. You couldn't write a device driver with a report writer but if you were writing a report, the syntax was very convenient. So you used the syntax designed for that purpose to specify the desired result. And then the runtime engine of our report writer would format the data retrieved by our query language which used the ISAM module to retrieve it quickly rather than doing linear scans of data. It used the indexes supplied by the C-ISAM package.

Johnson: One of the things we learned this morning was that Informix was really very much bootstrapped. There was a $20,000 investment from, as Roger describes it, his ex-girlfriend, who we later learned has been married to him now for quite a long time. But the marketing strategy, then, was that you were selling all these various little components in various places as a way to fund the ongoing development. Were you doing most of the development, Roy?

Harrington: Bill Hedge and I were all the development team for a while.

Sippl: Bill did slowly hire more people.

Harrington: Yes, he did. I worked pretty much all on the lower level parts. I designed the lower level parts of everything and we had more people doing higher level. So we eventually got more people doing query stuff and the report writer was done, I think, by Bill.

Sippl: Bill did the report writer. Betty Chang did our screen oriented data entry package. I did some of the first programming but I came to learn a couple of years later that none of my code still existed.

MacDonald:

MacDonald: It had been retired. I think a thing that's interesting history-wise is when you look at the speed with which companies are founded and they develop, now that we've had decades of the VC era. If we were going to start a company today, how quickly we would staff, how quickly we'd set goals, how quickly we'd think of market opportunities. We'd do all of these things. You look at the time scale here; it was founded mid-1980, after all. By 1981 they're developing things, they're just four people. When did you start selling C-ISAM?

Sippl: 1982, I guess.

Harrington: I think it was right then.

MacDonald: By early 1983 there was a full set of products, because I joined mid-1983 and I was the 13th employee. There wasn't a big infusion of cash so you couldn't staff ahead of the revenue. But there was a slow growth model.

Sippl: It was definitely a bootstrap method, yes.

Nick Baxter: I joined about a year later and one thing that impressed me was that for a 45-person company there was every organizational component you'd see in a 5,000-person company. All the departments were there. There might be a third of a person sharing other tasks, but every role was there. It was a full company. Different groups maybe had to evolve or grow with a little bit of pain because another group wasn't quite catching up. But everything was there at that time.

MacDonald: And I think we benefited from the fact that UNIX was so slow in developing as a market at that time. I think if it had developed faster as a market, if we'd been facing more competitors that had been better funded, who knows what might have been.

Harrington: Yes, our curve was exactly the same.

MacDonald: And I think Sippl said this morning: we basically started before there were hardware products we could sell on.

Michael Mahoney: So you were following customers who were using UNIX as their platform. You were writing for their platform rather than you choosing the platform and then selling that to the customers.

Harrington: We did not sell whole systems. We just sold software. So we only tried bringing software on whatever machines were installed.

Database Management Systems on UNIX

Mahoney: So you were bringing basic database management to UNIX?

Harrington: That's right.

Sippl: Back then each operating system usually had a database management system with it that the manufacturer supplied. So back in the days of proprietary operating systems, in

the 1960s and 1970s, each mini-computer in particular had its own proprietary operating system. So Digital Equipment had a line of mini-computers that ran RSTS [Resource Sharing Time Sharing, pronounced "ristus"]; they had another line that ran VMS [Virtual Memory System]. Hewlett Packard had a line of mini-computers that ran the MPE [Multi-Programming Executive] operating system. So each of these hardware companies, when they developed their operating systems during the 1970s, also developed a database management system to work with that operating system. And quite often it would have been very difficult for someone else to write a database management system to work with that operating system. They needed to be real experts on that operating system and you may have even needed to get the system calls that weren't published. So usually a multi-user piece of hardware had an operating system from the hardware manufacturer. It also had a database management system from that manufacturer. UNIX was an operating system with no particular hardware as its basis. And it was designed to be that way; it originally ran on Digital Equipment hardware because that's what the guys at Bell Labs developed on. But they developed it to be a portable operating system and when it did start getting ported around, it was a perfect market from my point of view because it was an operating system without a database management system.

Mahoney: And it was based on a flat-file system.

Sippl: It was even flatter than a flat-file system. It was a character-at-a-time file system.

Mahoney: Yes, it was a character file system.

Sippl: And most operating systems, commercial operating systems, were record-at-a-time operating systems where, when you created a file, you had to explain how big the file should be initially, so you'd say it should be 100K initially and should grow in 20K chunks. And you would explain, "When you go get another sector, go get 12 sectors at a time and reserve them for me." So file systems were pretty primitive and were pretty close to the metal in terms of the user interface for just dealing with files on most commercial operating systems. UNIX broke that mold and had a very elegant character-at-a-time file system and the whole concept of input streams and output streams. Inherent in the C-programming language was the ability to deal with a character at a time and the whole notion of filters in the UNIX philosophy was all wrapped around this concept that a file should be an ASCII file and you shouldn't have magic characters in files that screwed things up. You should be able to stream things in and out of various filters and do transforms. You should be able to take an ASCII file and put it into a sort filter and out would come a sorted list of those words. So this notion of a structured file type on the UNIX operating system was kind of contrary to the philosophy of the UNIX operating system itself that Thompson and Ritchie put into it. However, without a structured file system you couldn't do indexed searches for fixed-length records. And without the ability to do that you wouldn't have commercial applications running on that operating system. The notion of the database management system managing files of fixed-length records that have indexes within

them (one through ten characters in a field would be indexed but the rest of the fields' characters might not be) was different. That whole notion of the structure in Vtree -- another file that's the Vtree index mechanism indexing into that file was all foreign. So if someone was looking for Sippl, it would look it up within two or three disk accesses instead of maybe thousands or tens of thousands scanning linearly through the file. That was all something that the founders, the framers of the UNIX operating system really didn't want to get into. That wasn't part of their whole revolutionary model.

But particularly as the phone company kept using UNIX, they wanted to do the long-distance bill processing on UNIX and were running on larger and larger machines; it required that there be a database system. And since I'd come from consulting gigs at Bechtel and Wells Fargo Bank building database applications, I knew that UNIX wasn't going to go very far without a database management system. Yet it was the only operating system that was going to be portable among all of these hardware platforms that were going to come out of Silicon Valley. So just like there were dozens of single-user personal computer hardware platforms that all ran CP/M, I felt the same was possible with UNIX from our experience at Cromemco where Roy Harrington was commissioned by the founders of Cromemco to write a UNIX-like operating system because they didn't want to license the real UNIX from AT&T so they wouldn't be encumbered by the licensing agreements of AT&T. So they said, "UNIX, how hard is this to write-- how hard is that?"

Harrington: I said that I'll do it in two weeks.

Sippl: Roy, in my opinion, built the Golden Gate Bridge but it was all under water, because, he built it to run on an eight-bit processor, 64K of memory, memory bank architecture, single-user micro computer. And he built a multi-user UNIX clone.

MacDonald: That could work in those constraints.

Sippl: And it ran CP/M programs under an emulator that he wrote. And the emulator was smaller than CP/M was. So if you ran out of memory on CP/M you switched to CROMIX so that you could get more memory running under his emulator. It was a remarkable operating system he built, but the sad thing was that Cromemco didn't come out with their 16-bit Motorola 68000-based machine.

Harrington: Not until after I'd gone.

Sippl: Well, it got delayed some two years.

Harrington: It was too late, yes.

Sippl: So, that is why you left because we had something going on multi-user computers in this database gig.

Harrington: Yes, and I was interested in software and Cromemco was a hardware company that just looked at software as being a necessary evil.

Sippl: Roger Millen and Harry Garland [the Cromemco founders] were wonderful guys; they treated me like a graduate student; they gave me an opportunity to design all this software that became Informix. But they were hardware guys.

Harrington: Well, it was a great opportunity.

Sippl: Oh, yes, it was a wonderful place to work.

RDBMS Competition

MacDonald: Roger made a comment this morning that when Oracle got into the UNIX market, the program was so fat and big that it had a hard time fitting on these computers, let alone running. One of the benefits that we had was that the early UNIX industry had all these very small computers without much resource on them. And we had the expertise here from Roy's work at Cromemco and also just the design work from the get-go that Roger had done, that we were going to design for these smaller computers. And so we wrote a very lean, mean piece of software that could work within this environment. I think competitively that helped us tremendously from 1982 on when we had the database system through, about, 1986 at least, because during those early years the vast majority of UNIX boxes were small and had very few resources on them. And even though technically some of these companies were competing against us, if anybody really tried to run them versus us, we'd wind up winning. And that's what Roger found out: that all the AT&T people wanted was to use their own boxes and then after the fact they discovered we were the only database that really ran well on their boxes.

Sippl: The real tragic thing was, though, Oracle had all this market share and to a large degree Ingres did also because they ran on the DEC VMS operating system which ran on a 32-bit processor with lots of main memory, lots of disk storage, big expensive machines, quarter-million dollar, million-dollar machines. And they charged six figures, sometimes, for a single end user company of their product. So when one of their sales guys sold a few they got three or four hundred thousand dollars that month from that one sales guy. There were just enormous amounts of money flowing to those companies because they were selling to this commercially established VAX/ VMS marketplace. And so they had a lot more ads going, a lot more advertising and public relations.

MacDonald: They had a bigger business footprint sooner.

Sippl: Right, they were bigger companies. Ingres actually started in 1979, although I think someone said 1980, and they had access to venture capital. Oracle started in 1977, of course. So, by the time we got going in 1980, Oracle was already three years ahead of us and it was a company that was growing. They already had 40 or 50 people when I started and they were doubling in size every year. So the problem was when I went to get these OEM deals even though we were the only database system that ran on UNIX and ran well enough to use, these other companies would get the OEM deals from the bigger hardware manufacturers such as Data General or AT&T or Apollo. And they'd get these half-million dollar, million-dollar, two-million dollar OEM deals. Even Altos, although they invested in us, gave Oracle a $1.2 million OEM deal. And Altos had to do the port. It was mentioned that Oracle had a porting kit. But they would just throw this porting kit over the wall and nobody but a kernel programmer who had been employed at Oracle knew how to use it to do the port to the Altos machine. So Altos gave us a $300,000 OEM deal, bought 19 percent of the company for, I think, 1.2 million or something like that. And they gave Oracle all that money and got nothing from them because they never shipped Oracle to run on those machines even though they bought the port of Oracle.

MacDonald: It's a classic case, I think, of market presence. I mean, if we had started in VMS, too, we would have had an equal shot at that market presence. But we didn't.

Johnson: But you never questioned that, Roger?

Sippl: I did.

Johnson: You seemed to be so dedicated to the UNIX platform.

Sippl: I was, but I spent probably a quarter million dollars on an ill-fated port to the VMS machine.

MacDonald: What year was that?

Sippl: It was later. It was maybe 1985 or 1986 or 1987, somewhere around there.

MacDonald: It was after we had done an MS DOS version, because we had ported to MS DOS so that we could run on single-user boxes in the same environment. We did this because we had some value-added resellers that were buying these low-end UNIX boxes. It became attractive to them once the IBM PC was out to also be able to run the software transparently, if you will, on a single-user implementation.

Sippl: And my impression was that that was worthwhile. Our port to the MS DOS machine was successful. It was a commercial success and it helped us sell our product on UNIX. Oracle also ported to the IBM PC but as you heard it didn't work there either. And so if someone really wanted the same database software on both their UNIX machines and their PCs, they'd come to us; we won a lot of deals on that basis. And then as networking became popular, became doable, as 3Com started shipping products that worked, then people would run our data entry package and our report writer on the PC and it would talk to our database system on the server. So we had that same architecture, broadly speaking, that Sybase had. But we weren't smart enough to call the product "SQL Server."

MacDonald: No, but you did invest in the VMS project.

Sippl: We did. But then we gave it up. We also invested in an MVS port of our report writer to DB2 to make it the report writer for DB2. But that turned out to be a very messy technical project so we gave that up, too. Our cross-platform product across UNIX systems was highly successful. But our cross-platform ports across other operating systems we found more difficult than would be worthwhile for us financially except for the IBM PC. But the ports to all these other proprietary operating systems that Oracle was doing, we let that go to them and we focused back in on UNIX. And that was our secret to success: that we continued to run on UNIX.

Harrington: We could run on any UNIX box. I mean, we had multiple large rooms with machines all over and you could get Informix to run on them in a couple of days.

Sippl: Yes. And the real issue was if we only sold 20 copies on a particular machine should we keep supporting it, should we keep selling copies on that machine? So that was our issue: which ports do we whittle off the low end of the list? But it was a list of 100 machines. We were always considering whether to drop the bottom 20 percent.

Harrington: Sometimes they dropped out anyway. It was easier.

Sippl: Yes, sometimes the hardware manufacturers stopped supporting the operating system or something. But Oracle, by comparison, in the late-1980s they probably ran on five or six UNIX machines in a usable fashion whereas we ran on 80 to 100.

Marketing through VARs

Johnson: Yes. And through all this time you primarily were marketing through VARs [Value Added Resellers]? Is that right?

MacDonald: Marketing is a hard word to use even back then, to be honest. Stu Schuster from Sybase said in this morning's session, and I think it was true, "marketing wasn't marketing the way we think about it today in the software industry." If we took today's experience and went back then we'd do things a lot differently. Marketing back then was providing information. And I think naturally the earliest business consumers of UNIX were value-added resellers because prior to major corporations deciding that UNIX was okay to use for their projects, the people that would spend money and create business systems were value-added resellers, generally smaller entrepreneurial organizations. And this was worldwide. We had customers from Europe to New Zealand because we used to talk to them every morning. And they were developing these little systems to do this, that and the other thing. That kind of drove the early business of UNIX and we were the only database program that worked effectively on all of these boxes. And so we were like the natural kind of kingpin combined with this killer app that they developed, the fourth-generation language. Because when these smaller organizations needed to do businesses quick and dirty, to get them out the door, they needed to be productive. And when we developed a fourth-generation language we used to sell it. This was the message we marketed: in one tenth of the effort, one tenth of the time, you can develop a full application compared to writing in C or C++. That was hugely attractive. They found this language incredibly easy to use, very flexible and then they could adapt it easily for future versions. And that whole package was just very attractive to value-added resellers.

Baxter: You're still selling it now.

MacDonald: I still am. And we were selling to bigger corporations but it was like a slow trickle at first. And I think I did this in my oral history with you, Luanne. I remember one of the first business deals we got, this was in the mid-1980s, about the time when at UniForum more ties showed up at the UNIX conference than sandals. These two Anheuser-Busch employees were doing a project for Anheuser-Busch. But it was a system meant to go to their distributors. It wasn't a centralized system at Anheuser-Busch headquarters. They were using something like an NCR [National Cash Register] box or some early UNIX box and they basically did the numbers and they told their bosses, "We can do this product cheaper, better, faster using this new super-micro computer running UNIX and using this database system from this company, Relational Database Systems, than we could if we start doing this on a small DEC computer or a small IBM computer." And they sold their management on it and we did the project. But there weren't a lot of those projects going on. I mean, it took a renegade inside a large organization at that point to basically up-sell to the management that UNIX was worth a bet. Now you go ahead five years later to 1989, 1990 when we were talking about it, when Sybase was talking

about penetrating the finance industry, then they were getting a beachhead there. We got a beachhead in hotels with Hyatt because somebody at Hyatt decided it was time for a new reservation system. And they played off the multi-processor UNIX boxes versus going with VMS products from DEC or IBM mainframes. And they said, "It's going to be so much cheaper." So they bet on it. We helped make them successful and then within the next 12 to 24 months we signed up another six hotel companies. But that's how different it was in terms of the eras, that 1980 to 1985 versus that 1990 range. That's kind of in five-year chunks. So it was there but VAR's were the vast majority of sales in the early part.

Sippl: Yes, my recollection of it was almost a third, a third, a third. We had this dealer channel also which I always counted as an end-user channel with just a facilitator in the middle, where it wasn't a VAR, where they didn't have an application that they built on top of our databases. They were just like a ComputerLand or a procurement consultant-- a company that helped large corporate customers with procurements of things. And at least 25 percent of our revenue, even in the early days, was these end users that may have found us by themselves miraculously: someone at Northrup or at Rockwell who's doing a government contract that had been working with the university on it. And so they were using UNIX and they needed a database because they knew what a database was because they had other commercial operating systems that they used. So I think it was closer to a third of these sort of end users, government at first and then AT&T itself was a big end user. They were ten percent of our revenue.

In the early 1980s, AT&T would use their own computers internally or they would even buy other people's computers. They'd buy Altos and NCR machines and things like that. And then there were also these corporate deals for the visionary corporate procurement departments which often was sort of an internal VAR. But it was an IT guy at MacDonald's whose job it was to figure out how to find out from every MacDonald's store how many hamburger buns they'd need delivered the next day. So this led to just-in-time inventory control management from a central environment, like a MacDonald's headquarters or a General Motors parts depot to all the General Motors dealers and distributors. So these were corporate deals involving thousands of machines: thousands of small multi-user machines that were designed to handle six to twelve users. And they would go and be in all the MacDonald's. And they would go and be in all of the car dealerships. So on these sort of in-house corporate deals, we made decent revenue. And as the machines that ran UNIX got bigger, like the Plexus machines and then the Sequent machines, our price points on those machines got up there. For example, if you bought everything we had to sell for one of those machines it might be $40,000 or $60,000. So you only had to sell one of those to make up for 40 or 50 Altos machines. Our end user revenue through these various types of procurements, I always earmarked or ball parked at about a third of our revenue. Then the VAR revenue was another third. And the OEM revenue was about a third.

Corporate End User Sales

MacDonald: Yes, by the 1990s, though, I'd say the end-user corporate revenue was the vast majority of our business.

Sippl: That was the vast majority. When it finally switched over to be a decent business, when you got to deal directly with the end user and you got to get a differential-- you would get this list of requirements. And they'd have 112 questions: Do you do this, do you do that, how fast is this, how fast is that? And the benchmark wars started around then in the late 1980s. The DeWitt benchmarks: the professor in Washington was commissioned by AT&T to create a bunch of benchmarks for SQL-based database management systems. And AT&T was going to do a large procurement of, I think, four or five million dollars. And there were 30 tests in this benchmark, if I recall. And us versus Oracle and versus Ingres and we beat both of them. I can't remember the details on Ingres but I believe we outperformed Oracle on 27 out of 30 of these performance benchmark tests. And Larry Ellison, I'm sure, proposed and had published ads by Oracle, showing the three tests we lost, but not mentioning the other 27 that we won. And it was like we were on a scale or something. We were very heavy and they were light.

Johnson: That's funny but you're right, yes.

Mahoney: Now, that's marketing.

Sippl: "Oracle, wins again!" you know, and then this query, "Oracle, wins again!" and this insert, "Oracle, wins again!" So I thought that was remarkable. We won the five million dollars and he won the ad campaign.

SQL as the Standard

Johnson: You just touched on something I want to go back to. This came up this morning in the big meeting about this whole thing about SQL and how all the other companies that I'll call the independents since they weren't IBM, at some point had to grapple with the SQL language as standard. Can you talk about that a little bit?

Sippl: Well, I'm glad we survived that transition because that one scared the heck out of me.

MacDonald: It scared us all -- it was a company-wide concern.

Baxter: It was a make or break situation.

MacDonald: Could we make a transition successfully?

Sippl: Yes, it was do-or-die decision. We had two OEM's, Siemens and Altos, that were both pressuring us to switch over to SQL. We had not designed the internals of the system to support SQL on a number of fronts. But Roy [Harrington] proved to himself that we could still use C-ISAM as our fundamental retrieval method with its locking and everything. But we didn't have transaction logs. We didn't have commitment roll back. Some of the SQL statements were remarkably complex, the nested query statements and such, unnecessarily so in my opinion. So I wasn't really nuts about implementing this language and I knew it was what caused Oracle to not fit on a lot of these UNIX machines. So I was reluctant to do it but I knew the hardware was getting bigger, the memory was getting bigger, CPU's were getting faster. And I could see a standard forming and we were all about standards: the UNIX operating system is a standard, the C-programming language is a standard. We helped create the standards committee that became the POSIX committee. We were on standards committees. So I had to clench my teeth and say, "Okay, we're going to redo the product and it's not going to be that compatible with the old product because the query language is going to be different; it means all the report writing programs are going to be incompatible because it's going to be an SQL statement that retrieves the data. The same formatting syntax will work but it's really going to be a different product."

Johnson: About what timeframe was this? When did you make this decision?

Sippl: 1983 I believe.

Johnson: Oh, very early on.

MacDonald: Yes.

Baxter: You hired me then**.** I joined as product manager.

Sippl: You were the product manager for that?

Baxter: Yes, and that was mid-1984.

MacDonald: I'd been there for a while by then because I remember us grappling with this and it was a classic thing. We were trading off our resources.

Harrington: The discussion went on for a while.

MacDonald: We were very proud of our language, because we felt in many ways it was superior for the environment we were in. But it was as if the writing was on the wall, if we didn't get compatible -- that was what I remember us debating. I mean, long term if we don't adapt we'd lose it. And the whole database industry in the open systems era has been like that. If you don't watch it, at certain points something gets on the master checklist and if you're not there, you get left behind. And this was looming as one of those things: if you didn't have SQL compatibility, you were you going to get left off the list.

Sippl: So I saw that boiling up and it seems like we debated it forever. But I'll bet it was four months or something like that.

Harrington: Yes, well, that was forever back then.

Sippl: And I remember I was going to make a speech as part of a panel. And I know Gary Morgenthaler was on the panel. I can't remember if Ellison was on there or if it was someone else from Oracle. And I don't think Sybase was in the market yet but they might have had someone on the panel because they were coming into the market.

MacDonald: Not yet, no.

Sippl: Yes, they hadn't even started yet. But there were a lot of slides about SQL and its benefits and I thought Gary Morgenthaler was going to try to defend QUEL

MacDonald: His language.

Sippl: Right, claiming that it was a better language, a scientifically better language. And I thought, "I could do that but, boy, am I going to look stupid." What do users care, if it's just incrementally better? Is it noticeably that much better?

Harrington: It's just different.

Sippl: Do you have to get a masters degree in query languages to know whether to agree or disagree with that assertion? Skip it. If we can go to SQL, we'll have to go to SQL and so at that conference I announced that we were going to come up with a next version of our product that was going to be based on SQL. And Gary Morgenthaler almost died because he had all his QUEL slides up there. And he really had to tap dance through his presentation. I didn't mean to put him on the spot or anything like that but he just happened to be up after me and he had to really tap dance around why wasn't Ingres going to go to SQL as well. And he said "Well, we have been talking about it but, QUEL is so much better because of this and that. But it's not an industry standard yet. If it becomes an industry standard that is something else."

Johnson: And then they did support SQL a couple years later.

Sippl: Right. They double-clutched on it quite a bit more than we did.

Johnson: Yes.

Implementing SQL for Informix

Sippl: But, you know, we weren't sure we were going to be able to finish it, whether it would work, whether shipping an SQL product without commit and rollback was going to be satisfactory to the marketplace.

Baxter: Wasn't our first version also for the PC-AT or some such platform?

Harrington: With SQL? No.

Baxter: I thought it was for the IBM PC. Oh, there was a mini-computer version for IBM but then there was also a PC version and we were doing both of those. We had to split the kernel into two halves that would swap in memory.

Sippl: Oh, well, yes, we had a lot of implementation problems on machines that were too small.

Baxter: But I thought we were compelled to do it.

Sippl: It was bigger and so for some of our customers that wanted to run the new product on some hardware like the PC-AT which did not have as good a memory addressing architecture as the 68000-based machines, it didn't fit very well.

MacDonald: Yes.

Sippl: So it was a real challenge. Fortunately, our customers kept buying the old product; they kept buying the non-SQL version of Informix for what - six, seven years?

Harrington: Yes, it's still running, isn't it?

Sippl: Oh, I don't know, but they remained measurable part of our revenue until I left in the early 1990s; it was still bringing in quite a bit of money. So the smaller, leaner, meaner relational database product was better in a lot of ways for a lot of people. But we left them

behind and they didn't really squawk. I thought they were all going to leave and complain and sue us but we never really got a whole lot of heat from our customers.

MacDonald: Probably because industry-wise, it was in the ether. I think it's because industry-wise everybody was realizing that it was the future.

Sippl: Yes, it became obvious. By the time we finished it, it was probably pretty obvious.

But the other thing that was a shortcoming in our non-SQL product was our interface to the C-programming language. It was an API that was very easy to use but it was not really relational. It was more of a data dictionary-based API on top of our ISAM product and it did its calls a record at a time so you couldn't feed our query language into the programmatic API. It was either we were going to SQL or we were going to have to upgrade our API to include our query language syntax. And so we went to SQL. Embedded SQL for C was our first relational programmatic interface into our database engine. We were quite a bit behind these other companies in terms of the real technical transactional features. When the guys in Sybase said that they were transactionally better than we were, it was true for a couple of years there. But then we got the SQL version done and we went on to do a more sophisticated version of the SQL product. And I think Informix Online was the product that ultimately came out that had commit and rollback.

Johnson: And about when was that?

Sippl: That was the late 1980s or early 1990s.

MacDonald: It was early 1990s.

Baxter: No, 1989-- 1988, 1989.

MacDonald: The only reason I know is I named Informix Online.

Sippl: We did have an intermediate name.

Baxter: Turbo, I mean Turbo had it first. And then we changed the name.

Sippl: That's right. We had Turbo SQL which tried to use shared memory and had transactions.

MacDonald: Yes.

Baxter: And rollback.

Harrington: It had more disks and more memory.

Sippl: So we started making some of these things that made our competitors' products hard to use but we put those in for the more sophisticated audience. But it was a separate product.

Harrington: Well, it was trying to get going.

Producing High Performance Informix Products

Sippl: It was a separate product, too. So if you wanted to buy Informix SQL, it was still easy to install. Our files were files of the UNIX operating system, whereas with all our competitors, you basically formatted the disk to your database file system disk. We didn't take over the computer and shove the operating system out of the way. Informix SQL sat nicely on top of the operating system. As we got into our more sophisticated versions, well, then it needed a special version of the operating system.

Baxter: Well, the file-system wouldn't implement the locking we needed in the database engine. So we had to replace it.

Sippl: So we came up with a higher end product with a higher price point; it was more complex, ran on fewer machines, was a little trickier to install and configure and monitor. It was more prone to need a database administrator but it beat those other guys in many ways.

Harrington: It ran a lot faster, though.

MacDonald: Well, we needed it to because again that was part of the transition. By 1990 you started having business demand for higher transaction processing rates.

Sippl: And it wasn't six and twelve-user machines anymore. Now it was several hundred users.

Johnson: Well, when you started to talk about SQL, you said you could see that the hardware itself was going to become more robust. What was happening with UNIX during all this time? Were there things also happening with UNIX that changed the market?

Sippl: It changed a few percent but those places it changed, as far as the database technology was concerned, were important areas. It gave us record locking, it gave us shared memory so that we could have one database process that dealt with interactions from lots of different users. UNIX also had other changes that made no difference for us, that Sun ballyhooed quite a bit because Bill Joy [founder of Sun Microsystems] was getting the inside scoop from the Berkeley tapes that were coming over all the time. But none of those other changes really meant very much for us.

Harrington: Shared memory was the main thing.

Sippl: Shared memory was big. But then we had a product that was competitive with Sybase in terms of transactional processing performance. We also had good query performance. We also had good performance for report writing and such. But when we were also the OLTP [On Line Transaction Processing] leader suddenly, and then Sybase had a problem with their record locking architecture that finally caught up with them where certain applications designed in certain ways ran very slowly. Because we did record-level locking; we worked with the UNIX operating system authors to get a system call that would allow us to do that. Sybase didn't do record-level locking. They locked a much bigger chunk. I should have asked them while they were here. They'll probably wince and then tell me but SAP [an acronym for Systems, Applications & Products, the original meaning for the company SAP AG] could not run worth a darn on top of Sybase. And it was a big problem for them and took them years to re-engineer it.

MacDonald: And it was a pivotal business moment for us because in 1989 when we were probably at our low end marketing perception-wise in the industry, I'd say perception-wise analysts would rate us fourth at that point. And then by 1993 we were ranked as the technology challenger in the marketplace. So we had swapped places with Sybase and a lot of it had to do with our ability to do this.

Sippl: That was a big issue there.

MacDonald: SAP did not want to be beholden to Oracle. So they didn't want to offer just one option. They wanted at least two viable options going for their environment at the time. Maybe they saw a foreshadowing of the future when they were going to be more a competitor than a partner with Oracle.

Sippl: Oracle was already starting to compete with them.

MacDonald: Yes, and so they wanted to make sure that there were other database options. And when Sybase couldn't play they cozied up to us. And we were sending people to Germany.

It was a big project for us and then that was a key component in the perception changing about us between 1989 and 1993.

Sippl: Then we started our symmetric multi-processing project; we invested in a group in Portland and that was a success. And that compounded this issue where all of a sudden we came from being the best and easiest for report writing and 4GL and application building where you didn't have to be a rocket scientist, to having the best performing and highest OLTP-rated transactional database engine on UNIX as well with our high-end, high-priced products. And the SAP thing and the symmetric multi-processing thing, that just really hammered it home. And then Sybase's problems became sort of public. And we were number two all of a sudden. The only thing about Oracle is they started three years before us. All the way through the 1980s when I was running the company, Oracle would double in size. They'd go from 80 million dollars to 160 million dollars or something. Well, we had doubled in size, too. We went from ten million to twenty million or something like that. We were always one eighth Oracle's size. So we'd double in size which was about as fast as I could physically grow a company without going insane. And later in the 1990s, software companies-- when they had their heyday-- would triple every year; every now and then they'd double one year and then triple the next and then double again. But that was as fast as software companies could grow. So we weren't going to catch Oracle.

MacDonald: No, and I would compliment them and say that I think they became more marketing oriented before any of the competitors. Oracle started selling concepts years ahead of the other companies, realizing that by selling concepts, you're selling perception which is selling comfort on the purchaser's part. They were selling multiple versions before any of us were. I remember that when I took over marketing, I dealt with some of the product marketing people. For the first time I did an analyst tour and they were saying, "Well, we only talk about the next version. We don't talk two versions out." And I'm saying, "Well, look, our competitors are talking two and three versions ahead. So they're telling the marketplace, they're giving the marketplace a vision of where things are going." And today it would be like every company sells a vision way ahead of what they're doing or they're not going to be around. And we learned to do that. But, my hat's off to Oracle. They were pushing the envelope on selling the future way ahead of the rest of us.

Sippl: I'm not sure it was marketing visionary or just Larry didn't want to lose a deal.

MacDonald: That's true.

Sippl: He kept saying, "Yes, we're going to have that, too."

MacDonald: No. It wasn't that sophisticated, but he did do it. And then, we took off. At that point we started matching vision for vision. And it really helped change things.

Distributed Database Products

Sippl: There were a few hip fakes along the way where you're talking about changes in the database market and if you didn't go with it you were sunk. The fact that Ingres was late to get to SQL, I think, seriously damaged them. There were these distributed database wars and this was, I don't know, 1986 or 1987 or something like that.

MacDonald: It started then. It went on for years.

Sippl: There were these two-page color ads about who had the best distributed database and who had the best two-phase commit. This meant that you could have Oracle running on one computer and Oracle running on another computer. And you could submit a query that involved tables on both of these machines. And one of the machines would run over to the other machine and get that data for you and bring it back and you could get back a consolidated query result. And you could do a transaction. You could withdraw money from savings on this computer and deposit it in checking on this computer. And if one of them went down in the middle, the logging mechanisms would realize that and back out the transaction. Or if they both succeeded, the two-phase commit mechanism would commit that one and commit the other one and then, after it was done committing to both of them, give you the application back and commit that both of those are done, and they're good, they're solid. And this feature was going to be really hard for us to do and these guys had these two-page color ads going every week in *Computer World.*

Johnson: Just Oracle or all the other companies?

Sippl: It was between Oracle and Ingres. It was the big battle and Sybase got into it to some degree as well. And we were public by then and the industry analysts would call me up and say, "When are you coming out with Informix Star?" because they had Ingres Star and Oracle Star, whatever they called it. And the Oracle marketing spin was that this is why you're going to buy Oracle everywhere. Because then you can do distributed database applications. But if you don't have Oracle everywhere, well, Oracle doesn't do a two-phase commit with Informix. And this was the era where companies like AT&T or Southwestern Bell or someone had a lot of Oracle and had a lot of Informix. So what are you going to buy on your next procurement? Are you going to buy more Oracle or are you going to buy more Informix? So Oracle was trying to make that question self-answering by making a big deal out of distributed database. They wanted customers to say Oracle has distributed database but Informix doesn't, so since we might want to do distributed database operations in the future we'd better stop buying Informix and buy Oracle everywhere so that we can do a lot of distributed database transactions. We never implemented any of that.

MacDonald: Yes, we did.

Sippl: Never had it.

MacDonald: We did build the feature. We never sold it.

Sippl: Really?

MacDonald: Yes.

Sippl: You didn't build it in my time.

MacDonald: No, we did it later.

Sippl: You went ahead and did it anyway? Was that Phil White?

MacDonald: No, no, no. What happened was, and this was classic, because this would often happen. At any given point in time, there were procurement lists of features to do on an RFP; you'd get a request for a proposal and you couldn't play unless you had the features on this list. And what we noticed is after all of that B.S. going on, it started showing up on RFP's. And as soon as it started showing up on RFP's in order to even compete, if it became a critical item you'd have to have it. So we implemented some version of it.

Sippl: Nobody ever used it.

MacDonald: Well, that was the funny thing. So it was a check box feature, we had it. It was never an issue. Two or three years later the lead database guy at the Gartner Group [the leading market research firm at that time] pulls me aside and says, "Hey, Bob. I'd love to talk to you; do you have a customer that's done this?" And I just started cracking up because we've never had a customer implement it. So it became one of these weird features that it was like you had to have it to play. But then nobody ever wanted to risk doing it because I think in real life, nobody wanted to go through this thing of having these transactions maybe breaking across places. So they never implemented it. But it was just that way it went on. We did it. We finally did it after you left.

Sippl: I'm sorry to hear that you did it.

MacDonald: It was a waste of time.

Sippl: For two or three years I just stubbornly said, "No, we're not going to do that."

MacDonald: Yes, and we stayed with that until it started showing up on the list.

Mahoney: What makes these conversations so invaluable to us historians is if one were to go out and read the ads, as an evidence of what was actually being accomplished, you'd think they'd actually done it. I mean, I've remarked about how revolutionary computing is because there're always revolutions in computing. Most of them are called off for technical reasons. And getting evidence of where the hype is out there but there's nothing there is really important to us.

Baxter: Well, you would think that was a hugely important thing that everybody was using.

MacDonald: I used to hire consultants or analysts once in awhile before we would do a product launch. I'd have them come in and I'd show them what our road show was going to be and then I'd have them pull it apart for us. And part of the reason was that we'd usually include one of the analysts we were going to go see anyway. So we're giving them an analyst fee so then they're consulting with us and they're kind of in our rowboat; they're in the lifeboat with us then, wanting us to succeed. But one of these guys was great because we were talking about something and he said, "Well, you know, Oracle's really good at 'stubware.'" And I said, "What do you mean?" And he said, "Well, they are doing just enough of a feature that they can trumpet the fact that it exists. But when you really try to use it, you wind up stubbing your toe because there's not enough of it to really be of use. But they do just enough so that they could declare that they have it or it's on the list or that they have it. So, yes, it's true.

Baxter: Or if they had to finish it, they had the right stuff.

MacDonald: You can finish it later. It's just, like, "We've got it now."

Sippl: I grew up as a user of database systems, though, so I really never wanted to ship anything I wouldn't actually use myself.

MacDonald: Well, I think our culture reflected that.

Mahoney: And it fit with the UNIX culture.

Sippl: Yes.

MacDonald: Very much.

UNIX as a Portable Operating System

Sippl: Well, UNIX represented the portable operating system culture; I believed in standard operating systems. I thought this business of spending two years on a job working on a Hewlett Packard mini-computer running MPE. And then your next job is basically the same problem for a different bank. But they run VAX VMS and you have a six-month learning curve to start writing your first line of code. I thought that was just such an enormous waste of money and energy for the whole world that I was really against proprietary operating systems. So it wasn't that I was that much in love with UNIX per se because that's what I studied in college or anything like that. But UNIX was going to be the portable operating system. It was going to be the multi-user operating system of the future.

Mahoney: They wanted a platform for an operational system within the company, and, then, at a certain point…

MacDonald: This is at Bell Labs at ATT.

Mahoney: And then the decision was made, "Why bother with that extra layer. We'll just use UNIX as our operations platform. So that then became a niche within the company. It was about 1975.

MacDonald: That's interesting. So originally it was a platform for building things.

Mahoney: For building operational systems for the telephone company. And then they decided, "No, if we're going to build it on UNIX, why not just make UNIX our platform within the company?" And that's how it spread through the company out of the research division and then to AT&T. And I would gather, once one made that commitment, then one started to worry about, "Well, if it's going to be the company's operating system, what else does it need? And do we want to try to build our own database or get other people to do it for us?"

Johnson: You mentioned something, Roger, that you were talking to the UNIX guys about adding some features that made it possible for you to do it. Were you always in communication with the UNIX people?

Sippl: No, I never got to talk to Thompson and Ritchie. This was something we had to do initially at the level of the licensees, so Altos, Plexus, Onyx, these guys licensed UNIX from AT&T. What we did was we designed the locking system call. And I wrote it up as if it were a page in the UNIX manual. And some consultant or maybe someone at the Santa Cruz Operation actually wrote a sample implementation of it because we didn't have UNIX source code, but Santa Cruz Operation was a porting house for UNIX so they had UNIX source code.

So, I think they actually wrote up a sample in the implementation of how it would run on the UNIX port of the Onyx machine or something like that. That was given to any hardware manufacturer willing to promise to put it into their port of UNIX.

So that's how we got it put into UNIX. So when these companies would port UNIX to their hardware, they would often add some things to try to differentiate themselves from other companies. And, there were two things I tried to get them to do: one was to add our C-ISAM to their shipment of UNIX so their shipment of UNIX would have an indexed file system; I was just really surprised it did not go better.

Harrington: Yes, I know.

Sippl: The other thing was to add that locking call. And we were quite successful with the locking call. And then when we started the /user/group and its standards committee, which then IEEE asked us to turn over to them as the POSIX committee. We put that locking call into the POSIX standard. So then AT&T, when they saw it coming into the POSIX standard, put it back into the UNIX original source code that they gave out to the licensees. The next version of UNIX that came out to the licensees had the locking call built into already. So that's how we got things into UNIX. We kind of made the licensees do it

Harrington: Backwards.

Sippl: And force it back into. Inevitably we did license C-ISAM to AT&T with full rights for them to bundle it with the UNIX operating system. We licensed them the source code and the right to include the source code with their licensing of the UNIX operating system to the licensees. And they had some management change, after they signed the deal and paid for it, and they never did it, as far as I know. But we did that deal. We got $4 million for it.

Johnson: At that point, you really weren't selling the C-ISAM.

Sippl: We didn't think we were. We decided to do these fire sale deals where you could have all you could eat for $100,000. So if you were a VAR or an OEM and if you were coming out with a new UNIX computer and you wanted to bundle C-ISAM with it, and you wanted to have the source code so you could support it. It was $100,000 to get the source code and ship as many copies as you want on the platform.

MacDonald: It was late stage.

Sippl: We sold a lot of those. Revenue went up when we started doing that. That was our abandonment strategy of this product and it caused the revenue to climb for the next two years. That was our end of life strategy, but it didn't work.

Mahoney: The last of your seed corn is the most expensive.

Sippl: Yes. You know, C still needs an ISAM. I mean if you buy a book on the C programming language it says nothing about how to read and write records using an index; it's sort of criminal.

MacDonald: All I know is we spent a lot of time and effort with our porting lab over the years because since everybody would do their own variations, you never knew what they were doing, so we'd have our porting engineers who are already pouring through it, asking, "What's the new release of their version of UNIX? Does it have everything we need?" And then, once in a while, some numbskull at some company would pull up parts of the operating system. And since we had designed to the operating system, unlike some of our competitors, this all mattered to us. So, if we were going to bypass it it really didn't matter, but we were relying on it.

Mahoney: Yes, my daughter does software integration. She started at Cambridge Technology. She's been with a series of companies. And I've made remarks about portability and UNIX, and she said, "I'll tell you about portability!"

MacDonald: Yes. I mean it was a good theory, but it was sort of like we had to do so much.

Harrington: You had to know what you could use and what you couldn't.

MacDonald: Yes.

Baxter: There were some UNIX wars for a while there, too, because there was UNIX International and then here was X-Open. And they had their proprietary flavors of UNIX. And, we were one of, I think, two companies that were in both of these organizations. And we were asked a number of times, "How can you play both sides against the middle on this?" And our response was very simple: "Two flavors of UNIX are much better than 150 flavors." When I started, maybe a third of our building footprint was the porting lab, a third of our space was the room with all of the computers in it. If we only had to do two ports to UNIX that would be much better.

Harrington: And we wouldn't have all of these issues.

Sippl: There was a lot of UNIX gamesmanship along the way. What was that group? Motorola started a marketing group for a binary standard for UNIX. So you would only port once and you compile it and it was supposed to be binary compatible across 20 manufacturers. I told them that would never work and -- it didn't work.

Harrington: Xenix was trying to do that. Remember, Xenix.

Johnson: So, basically you had a computer from each of these various manufacturers.

Baxter: We had multiple computers.

MacDonald: We had multiple computers from each manufacturer.

Baxter: Because there would be version five of some operating system, version six of the operating system. We need two or three.

Johnson: And you might have a customer that still had version five.

Baxter: And we had to upgrade the operating systems.

MacDonald: Our manufacturing list at our manufacturing facility (which was just basically making tapes), had a long, long list, because you had all of the different boxes, all of the different version of UNIX, all of the different versions our box. And so you had to multiply all of that.

Harrington: It was a severe portability issue.

Baxter: Yes, it was thousands of SKUs. By the late 1980s, we had easily over 1000 SKUs.

Johnson: That's really interesting because I always think of the portability.

Harrington: At the source code level it was not a big deal. It's just that each object code would be different because everybody would do it a little bit different. It wasn't that it was that hard to port from machine to machine, it's just that you actually had to do every single one.

Sippl: And after you ported you had to run to make sure that the port succeeded.

Johnson: Interesting. From the outside of that whole UNIX thing you think that's the solution to everything. But there's still a portability problem. Where I was working we were porting basically COBOL code to a lot of different mini computers. It was a big deal and everybody dealt with the portability question, but it was still there. In UNIX it was a little different.

Sippl: Yes, it wasn't as bad on UNIX because it was fundamentally UNIX you're porting to. And as Roy [Harrington] said, we had certain style rules that his organization followed - we didn't use the broad set of system calls. We didn't use everything that the C programming would to offer. We tried to keep things down.

Harrington: We did things that were portable. There are certainly standards of driving. We used the smaller set, the common set for everything, which is kind of weird because you then got a lot of manufacturers who would say, "What could we add that's special, then we could make it special?" We'd say, "We really don't want anything special."

Sippl: Yes, they would come to us and say we want to specialize our machine. We want to make our port the best. What can we do?

Johnson: So when was it? Was that when you went public? It was 1986, wasn't it?

Sippl: That's right.

Informix Acquires WingZ

Johnson: And then you became Informix and you went through that situation where there was an acquisition of a company back in Kansas.

Sippl: Yes, we did.

MacDonald: Twenty years ago.

Sippl: Twenty years ago?

MacDonald: Yes, wasn't it 1987?

Baxter: 1988-1989.

MacDonald: I came back in 1988 to the company and you had done the deal early that year.

Sippl: Yes. The Wall Street analysts had gotten to me. They had succeeded in scaring me that Oracle had eight times our R&D budget, and eight times our marketing budget. You know, how are we ever going to keep up with them feature for feature and stuff like that. And they convinced me that we needed a differentiated product and a differentiated strategy. I had also started on a theoretical thesis that database management would continue to expand to include the intelligent search of anything, so word processing documents, emails, spreadsheets should all be indexed. This notion that you save a file in the file system and then you can't find it again to save your life, I thought was really a loser idea when we had all of this great technology for finding and storing and retrieving things based on a searchable field. So, I started toying with the idea that if all documents were stored in a database management system, and had tags on them so you could search on the tags but also any text that was in them, we could extract and put into a free text searchable search engine (which databases did not have). That was when I went back to AT&T, again, and said, "We should build this. Informix should build this SQL database engine that has the ability to have a field of a record be from zero to n bytes that could be a large object. And that we will also free text search that blob of data. And therefore, people will be able to search for documents, based on just typing whatever they think might be in the document and we'll search for that document and we will retrieve the document from the file system for you."

I made the pitch to AT&T that the operating system of the next version of UNIX should actually have a database management system as part of the file system. And the file system should use that database management to actually be the storage and retrieval mechanism. But that was too big an idea to get through to AT&T. So I thought, "What would happen if we owned a collection of office automation software?" Microsoft did not quite own the office automation software market back then. There was WordPerfect. There was Lotus. And there was a set of products under UNIX. So, I thought, "Well in this business where we're networking PCs to UNIX machines, if we owned a suite of office automation software, then we could basically have those products modified so that they stored their files into our database system and so we could revolutionize operating systems, as well as office automation, as well as database management all at the same time."

MacDonald: There was another key point back then; it still wasn't universal that there was a PC on every worker's desk in every kind of company across America because we were selling into companies that were using UNIX terminals tied to UNIX systems for the people that needed access to computers.

Sippl: It didn't really matter. We bought a company that had all of this office automation software, and their spreadsheet was about to become a graphical spreadsheet that was really sexy and it was before Excel had come out on Windows. Unfortunately, after I bought this company, I realized the whole gravity of the company we had bought was all about bringing this graphical spreadsheet to market before the Windows version of Excel got to market. And that it

would have monstrous more bells and whistles and greater graphing and look slicker. And, indeed, it did. It was a beautiful product. It had great graphics, and you could spin these 3D bar charts around in the sky, it was all sort of slick.

Baxter: Light sources, 3D light sources, remember?

Sippl: 3D light sources, so you could get the shadows on the bars the way you want. It was just gorgeous. And these guys that we bought had booths at these trade shows that looked like spaceships.

Baxter: With Leonard Nimoy.

Sippl: They had Leonard Nimoy. You would go and you sit in this theater inside this booth and Leonard Nimoy was on the screen telling you about what a great spreadsheet this is going to be. And they gave out WingZ bags to people; that was the name of the product, WingZ. Their WingZ bags were great. They kept telling me this product would ship in three months. And then two months would go by and they'd tell me they need three more months, and then two more months would go by. They spent a million dollars a month in marketing on this. All along it kept slipping. After it slipped about seven to eight months, I flew back to Kansas City and I spent three days with the product manager and the engineers and I started crossing off all of these features that they had added to the list since I bought the company and since they told me it would be three months. Because the feature list just kept growing, every three months, three more months. So I started crossing stuff off and I flew back. About three weeks later, my QA guy comes in and says, "By the way, they added all of those features back. And the programs we're getting to have more bugs than the ones we were getting three months ago. And the bug rate is climbing, not falling. And even when it starts falling, if it falls at the rate it was climbing it will be a year before we ship this." So that led to a large fight and a major problem for the company.

Getting a New CEO

And, that's when I decided I was a little burnt out on the whole thing. I had survived cancer. I started this company when I was dead broke. We had built it up. We had taken it public. And now I had created a mess. And I was just a kid; I was 24, when I started it. I was now 33 or 34. And I told the Board that we needed to bring in someone to run this company who knew how to run bigger companies because it had gotten to be a big company and I'd never even worked at a big company, much less run one. And my Board was telling me I was underestimating my capabilities. And they're saying, "You think you're screwed up, you should see every other company," and things like that. But it didn't comfort me any and I felt very bad about the whole thing. So I tried to hire Bob Finocchio to be President of the company and I almost pulled that off. He almost took the job, but he went to 3Com to be a senior VP. The recruiter convinced us

that I was right: that we needed to hire a CEO and replace me. And my Board was very reluctant about that and I had to convince them one at a time. I finally did and we did a search and that ended up with hiring Phil White. And then Phil ended up not really working well with me as sort of the technical guru or CTO or evangelical lead or anything like that. He was a control freak and little did I know, it took me two or three years to figure this out, but he really didn't want me around at all. So inevitably I went on and did other things. But, that was the saga. And I thought Phil was going to come in and fire all of these guys or sell the company back off or spin it off or shut that all down and just focus on the database. But he didn't. He kept making sort of half decisions. But the database business took off again, for better or for worse. Also, right around there, I guess the whole database industry had lots of bad quarters, but I didn't realize that. I thought it was just me. I made these horrible mistakes. And our revenue was climbing now on the database side, too, for the first time in history but we had lost focus and everything else.

MacDonald: Well, I think it was a pause before it really took off with all of the corporate sales that we saw in the 1990s. In the late 1980s, the database concept was getting digested by bigger companies. And it was that pregnant pause where it's, "Are they going to be successful?" And industry by industry as the examples came up showing how it was successful here, whatever bank you guys were working on with Wall Street was successful, all of these other people started projects and the things took off again. I'm talking about Sybase here across the table.

Sippl: I think if Bob Finocchio had taken that job, we would all still be working there; I really do, because we had a great company. We just needed some more experienced hands-on senior management to come in and help us out, because we were all young. And if we had gotten a guy who was eight whole years older than we were or something like that and had studied under one of the Silicon Valley greats to come in and help us run it and just be more systematic about the numbers and make harder decisions things would have been different. We needed to stop the spending that was out of control on products that weren't going to ship, or even when they did ship they were going to be dead in the water because Excel was going to ship at the same time by then. If we would have just been able to make those hard decisions ourselves, which I don't think we were capable of, or just gotten enough help to do them, I think we would have not only survived that mistake, but we would have also survived the database doldrums financially in those couple of years. And then with the Informix Online product, we would have all enjoyed terrific revenue growth.

The Changing Competitive Environment

Johnson: We talked about this a little earlier. With Sybase and Oracle, when it came to the 1990s things became incredibly competitive. The earlier discussion was that everybody had their own sort of different platforms and so on. At this point, was it that the platform became

irrelevant? Or was it the fact that UNIX simply became suddenly more used? Or was it just that the world became more dependent on database capabilities? Did companies have to be able to retrieve information in that way in order to be competitive in their own industry?

Sippl: All of the above.

MacDonald: I think, in the early 1990s what happened was it became clear that open systems were going to be the next generation systems at more and more corporations. They were going to sunset their proprietary systems and go to open systems. During the 1990s, a guy in investment banking turned to me and he said, "There was a period in there where you couldn't lose investing in relational database management system companies. There was so much wind behind them, a combination of the move to open systems and the need for database technology on those open systems that if you just could have spread your money across all five companies, you would have made a killing." And it was strong, because we were all growing. I mean as competitive as it was between us, we were all growing. We were all adding customers like crazy. The deals were getting bigger and bigger, the customer's enthusiasm was so high. During that period, I think there were thousands of licenses that were sold that are still on shelves at those corporations because they were so convinced the future was UNIX database. It was, "Well gosh, if we can do that kind of a deal with you, we'll buy some extra licenses for our future use." Today you wouldn't even hear of that. Today it's every dollar - you've got to beg for every dollar. But back then, we were doing some deals that were just unbelievable. So, as competitive as it was, it was also the whole segment was just surging for a period.

Sippl: And the 1980s were competitive, it just wasn't among these companies. I mean we had to knock off all of those other companies including Unified. Unified was our number one competitor by three fold for like four years from like 1983 to 1987. We did have to fight tooth and nail for deals versus them. But we, overall, won most of them.

Johnson: That's why you're here today and the guys from Unified aren't.

Sippl: Right. They fought a good fight, but they didn't quite get across the line. Nikos Nuremberg went on to start a different company that became very successful and went public that, what company was that?

MacDonald: Actuate.

Sippl: Actuate, yes. And, in fact, the guy who ran the company that I acquired for the office automation debacle is running a European ATM software company that's worth $1.2 billion now. So these people all did well. It was just a question as to what degree were we right? What tactics did we employ? And who hung there and who flinched and stuff like that.

But, all boats floated for several years there in the 1990s the boats that were left were OK. So, even though it was competitive then among these few companies, Sybase, Informix, Oracle and Ingres. But UNIX was a foregone conclusion. SQL was a foregone conclusion. Relational database was not even discussed any more, relational or non-relational, way old as far as an issue goes. It was really marketing, politics, quality of your sales force, execution issues that determined whether you won more than the other guy won, but everybody was winning.

Going Public

Johnson: Was there a significant difference at the point that you went public? We often hear from a lot of people that that was such a critical turning point in their company because suddenly they had to start managing things differently because they were public. I'll just let you answer the question straight without speculating on the other stuff I've heard today. Did things dramatically change by going public?

Sippl: It didn't change stuff for us, a whole lot, because we had this guy who was my business advisor named Dan Sully who I had put on the Board. He had run larger companies, before, and he was a tough nut. And he made us do things very strictly all along. So we didn't have that huge change, except for this business, which was all new to me, of Wall Street analysts and the stock going up and down, what it meant if you missed your forecast and where did these guys get this forecast from anyway? So the whole game that you play with the analysts who put out forecasts of what your revenue is going to be next quarter, and then calling you up all of the time and trying to find out whether you're going to hit that number or miss that number. And then, when you do miss by a little bit, the huge reaction that you didn't expect. That was all new. I didn't find it particularly intimidating. I found it interesting and a little hectic. But that wasn't nearly as scary as trying to figure out a strategy to differentiate yourself from Oracle or something like that. But, I don't think we did need to differentiate ourselves any more than we already had or were doing. I think we were doing all of the right things for the product. Yes, I think we ended up differentiating ourselves by not going into the accounting business and the applications business on top of the database. We continued to embrace our VARs and focus on the database products. And I think that differentiation in itself would have succeeded and did succeed throughout the 1990s. So it turns out we didn't have to do any of the radical things we did. Going public was very good for the company, because again, we were a small company; we still didn't have very much money. We were cash flow positive. When we went public, I think we had been profitable for 10 or 12 straight quarters; we were cash flow positive all of those quarters. And I mean consecutive quarters; I don't mean the profits went up more the current fourth quarter than the prior fourth quarter. I mean profits went up every quarter. But, we were doing $20 million year when we went public. It wasn't a huge company. Our market cap after the IPO was $54 million. The stock was trading around $7 a share so the whole company was worth $54 million. You guys raised 70 million, right? So, I mean they raised more money than we were worth after the IPO. So, we were kind of the company that

got no respect for a long time. I was surprised Hambrecht and Quist was willing to do that IPO because it was such a small deal for them. It was almost in the round off error in terms of numbers to them. So, we decided to do a smaller deal than we might have done otherwise.

MacDonald: You did a secondary, though, a year later to raise money.

Sippl: We did. When we did the initial public offering, the second public offering was already in the plan. We thought we'd do the first and then get better known. Then we'd put out a couple of more good quarters; because business was going well, we were pretty confident about that. And then, we planned to do a second public offering, which we did at around $18 or $19 a share. And that's when Altos surprised me. I got a phone call from someone at H&Q, I think, informing me that Altos had sold their entire position, 19 percent of the company in one block trade. "Really, that's nice to hear. I guess I don't have to worry about them anymore." But that's what small potatoes we were. And, I think they cashed out $20-some million. And then, when Acer bought Altos, I think Dave Jackson said, "You know, Sippl, we still have that $20 million and I got it out of Acer. I kept it out of the deal." So I think they made more money on their investment in Informix than on their net operations. I mean they were a successful company.

MacDonald: No, it was interesting because we were profitable for 10 to 12 quarters before going public. It was hard to track things back then because you just didn't have pervasive information the way you do today, but at some point a Wall Street analyst had come to us, and said, "You know, you may the only profitable UNIX company." But he had actually been counseled into running a profitable business by Dan.

Sippl: Well, that, and the fact when we were putting payroll on the Visa we didn't have a whole lot of choice.

Johnson: Mahoney, do you have any more questions? Do you guys have anything that you want to talk about that you think was really significant in the evolution of Informix that we haven't covered? Anything that you think that really needs to go down into the record for the history books?

Sippl: Well, I do think, even though we never spent any of the money Altos gave us, I think that was an important investment. There was a VP of engineering there named Kapil Nanda who went on to run a very successful consulting company after that. And he was the one who pressured me to do SQL along with the senior computer scientist from Siemens, Hans Georg Zimmerman. So those two guys are unsung heroes in the UNIX business, really, because they were both major factors making the super micro business successful in the U.S. and Europe but no one ever hears of those two guys. But it was Kapil who decided that Altos

should invest in us and that helped us go public, helped attract attention from the investment bankers and such.

Harrington: What else did we leave out?

Sippl: Laura King was quite a factor. I'm sorry she couldn't come. She was our VP of marketing early on. And she filled a lot of different roles over the years in the marketing area.

Johnson: She came over with you from Cromemco, right?

Sippl: That's right. She was my department. Who else are we leaving out? Are there any other major players? We had one gray haired guy in the company, Dick Murphy. He did help us run the ship.

Mahoney: That was just a very a young company.

Sippl: Yes. We had investors. As I said, it was our lawyer and his buddy, a real estate guy, Al Norup and Jay DeBenedetti. They were both on our Board. And I'm sure Al didn't know a personal computer from a toaster oven most of the years he was on the Board. And Jay wasn't much better, but he tried, at least. But Jay was always a real estate guy, so he was very numbers oriented. And he'd always say, "What are you giving all of these options for? You know, we've got the employee stock ownership plan. We've got this option plan." He always thought the compensation was too big, that we were giving all of this stuff away to the employees. But, we were early in the stock options model. So in 1980, when I started the company, when I hired people, I gave them big chunks of stock options because the company wasn't doing anything.

Harrington: There wasn't any money to give them.

Sippl: I'd say, "If we worked a lot and make something out of this, who knows, the stock might be worth something some day," rolling my eyes, because I thought it was kind of a long shot. But, we worked real hard and we turned that into real money. The problem was the options were written to have a five year life. And 1985 was coming around, and we weren't public yet. And, if these people exercised their stock options, they were going to have to pay taxes and I think it was the alternative minimum tax that caused that problem because they were supposed to be exempt from taxes, but Congress, I think it was after we granted the options, passed the law that said, "Yes, they're exempt from taxes until you sell the stock, except for the alternative minimum tax, but, married people get a $40,000 exemption and single people get a $25,000 exemption." The people that we gave these stock options to were saying, "What, a $40,000 exemption? If we go public next year for $7 a share, my options will be worth

$500,000, so I've got to pay taxes on $500,000 minus $40,000. I have to pay taxes on $460,000. I don't have 40 percent of $460,000." But the options were going to expire. So they had to exercise them or lose them. But if they exercised them this gain of $400-$500,000 was going to put them into the alt-min and they were going to owe $200,000 in taxes that they didn't have. So, we were totally hosed there. Plus, we had another little problem where there was the personal holding company problem. This is in Microsoft's prospectus as well as ours, where if the top five people own more than 50% of the company, it was considered a holding company if your revenue was from passive sources. Selling software was not considered a product, according to this IRS manual. It was considered royalties because we wrote it once and we're publishing it and we're protected by copyright law. So it's like we're selling books; it was considered to be royalty so it was passive income. It's just floating in. We could barely keep the money out of the place. So we were going to be considered a personal holding company and that was going to be a huge tax problem on the IPO. So Harrington and I and Laura, I think…

Harrington: And Dan.

Sippl: And Dan Sully.

Harrington: Dan Sully and I worked a lot figuring out…

Sippl: He came up with the spread sheet…

Harrington: On how to do it.

Sippl: So, we sold stock to an employee stock ownership plan that we created just for the purpose of having another entity that would be a shareholder that we could offload some of our obviously excess stock. And we did this at something like, I don't know, $1 a share, $2 a share, something like that. The company did go public the next year at $7 a share so we got really screwed on this whole deal.

MacDonald: Thank you, Roy.

Harrington: Yes, you're welcome.

Sippl: But we kind of said, well it will be good for the employees, guys like Baxter, guys like Bob. So, at the next Board meeting Jay said, "Well it's nice of you guys to do that. Now, let's get rid some of these options that we're giving away. Now they've got this ESOP thing, we don't need these options." But we had lots of little structural problems involving being one of the pioneers not just of the database industry but of the software industry. In our prospectus, we

had to have risk factors like it's not clear that copyright law is going to prove to be suitable for protecting copies of software, that if large piracy occurs it's not clear we'll succeed in court. It's not clear that we'll be able to enforce our copyrights overseas. All of these risk factors, unfortunately, were actually true. And, we also had another risk factor in the prospectus that was we disclosed the fact that I had had cancer when I was 19 and I was 29 or 30 when the company went public. And we didn't know exactly how to phrase that. So we were going to go back to Henry Kaplan who had invented the cure for Hodgkin's disease and who cured me and tried to get a measure on what the chances were I was going to croak during the next few years because the investors did want to know if the CEO was going to live or not. But I was way out of the risk period, because the curve on it kind of goes - after five years, you've pretty much got a normal life expectancy. Some cancers aren't like that, but Hodgkin's disease was. So what they did was they bought a key man life insurance policy for me, and they got the same rate as a smoker with high blood pressure, so that's what went into the prospectus. It said, "There is some risk Mr. Sippl will actually relapse in cancer. It's vanishingly small according to actuarial results, as evidenced by the fact that we were able to get a key man life insurance policy with the same rate as a smoker with high blood pressure. And you guys invest in companies that have CEOs who smoke with high blood pressure all the time. So you can obviously neglect that little news item." But every now and again on the IPO road-show, someone would ask the last question, "Mr. Sippl, how's your health?" "Pretty good. I just ran a couple miles this morning. How 'bout you?"

Johnson: OK. Thank you very much.