An Interview with

CHARLES W. BACHMAN

Conducted by Andrew L. Russell[1]

on

April 9, 2011

Boston, Massachusetts

IEEE Computer Society History Committee

Abstract:

In this interview Charles Bachman, best known for his work as a database engineer and theorist, discusses another aspect of his career: his experiences with computer-communication systems. He describes his work with torn tape systems at Dow Chemical in the 1950s, his development of a manufacturing control system at General Electric in the 1960s, and his work on databases and distributed systems while working for Honeywell in the 1970s. Bachman also describes his involvement with industry standards committees organized under the auspices of the American National Standards Institute (ANSI) and the International Organization for Standardization (ISO), including his Chairmanship of ANSI and ISO committees working on standards for Open System Interconnection between 1977 and 1982. He explains his resignation from the Chairmanship after he joined Cullinane Database Systems in 1982 and his subsequent work developing data structure diagrams to clarify relationships between different types of computer architectures.

---

[1] John Day also was present for the interview. See page 19 for further explanation. Connie Bachman helped to edit the interview transcript.

ANDREW RUSSELL: This is Andy Russell, it's April 9, 2011, and I'm here with Charlie Bachman, who has graciously agreed to be interviewed. We have an extensive interview that you did in 2004 with the ACM, with Tom Haigh, so we don't need to do a detailed life history.[2] Instead, I hope we can focus on your activities in computer networking and computer communications, particularly that culminated in the OSI [Open Systems Interconnection] effort in ISO [International Organization for Standardization]. Can you begin by describing some of your early experiences with computers and communications?

CHARLES BACHMAN: Andy, thank you very much for the invitation to come and tell some of my story. A long time before the center of our discussion today, which is the ISO work on communication, there were places where I dipped my toe into communications world, not as a primary focus, but as just part of my work experience.

In a short summary, I spent ten years working for Dow Chemical, from 1950 to 1960; I spent ten years with General Electric, from '60 to '70; and ten years with Honeywell Information Systems, from '70 to '80, two years with Cullinane Database Systems, from 1981-1983,and fourteen years with Bachman Information Systems.

It wasn't until we got to the Honeywell part, in the 1970's that the ISO work on Open System Interconnection came into play. While I am best known for my work on databases and data structure diagrams, my fundamental skills are those of an architect: analyzing requirements and constructing simple, but elegant, solutions. The computer and communication specialists from the various participating countries could fill in for my lack of detailed experience in the communication world.

In my 10 years working for Dow Chemical, I started out working in engineering, and successively moved through the finance and production departments. Eventually, I was asked to become the manager of a new group called Central Data Processing. That was a corporate level organization. It was while managing the CDP group that I finally launched Dow's attempt to install a large computer for their business operations and was asked to organize it and manage it.

I became familiar with the data side of the corporation. One of the very obvious things was the increasingly important interrelationship between computing and communications. Dow had a teletype communications system

---

[2] Charles Bachman interview with Thomas Haigh, September 24-25, 2004, Tucson, Arizona. Available from http://portal.acm.org/citation.cfm?id=1141882. See also Thomas Haigh, "Charles W. Bachman: Database Software Pioneer," *IEEE Annals of the History of Computing* 33 (2011): 70-80.

tying together their factories, their administrative offices, and also the sales offices. Their teletype machines generated a paper tape as output, or you could type on a roll of paper. The output is punched into a tape. Each piece of tape was a transaction spelling out a new purchase order, or a request for authorized overtime, whatever it might be.

The little message requires some response to it. The systems used in the 1950s had teletype machines for the communications links but there were no computers used for routing messages through the system or responding to a message. Instead, the system actually relied on people to tear a piece of tape associated with a particular message from one teletype communication link and then reinsert it into another link to complete the communications circuit. This is what we called a "torn tape" system.

In other words, with a torn tape system, such as the one we used at Dow Chemical, all of the teletype messages from all over the country came into the Dow teletype room in Midland, Michigan. As the messages came into the teletype room, the operators would read the header on each messages as it was being punched out on tape. They would look at it and say, "Okay, here's the beginning of a message," and when the end of the message came out of the punch they would separate the piece of tape for that message. Each message had a header with the destination code followed by its text. They would feed the torn tape back into a second teletype machine that was wired directly to the communication line to which the final destination teletype machine was attached. That communication line would take the message to its final destination.

In some cases the data traffic would need to be a multiple drop link with the communication line from Midland routed to the East coast with drops at several sales offices along the way. And so you could connect with the Boston, the New York, and the Philadelphia sales office, and maybe the Baltimore sales offices, all getting the same torn tape messages routed to them by ever more people in their local teletype rooms.

I don't know exactly what that routing was but when that message came across the electronic system, clickity-click, clickity-clack, that's what it sounded like anyway, each teletype machine along the line would listen, "Is this mine? Is this mine? Is this mine?" And then if not, they'd just ignore it and wait for the next message to arrive.

If it was theirs, they would say, "I'm going to punch out this tape and also print it on a piece of paper to give to someone to process," Everything went directly from the source to the Midland torn tape switching center and was forwarded from there to the final destination.

I was knowledgeable about that torn-tape system and knew what improvements to the system could do towards helping the company prosper and getting things done sooner.

RUSSELL: So the system you described was geared toward improving efficiency within the company?

BACHMAN: Yes, they were able to provide better customer service. And better customer service is continually a driving force for every company. The company has to process the orders faster. If someone wants something now, can I get the answer back to you in an hour? The messages went only to specific offices that were prepared to respond to the information supplied. This was my first introduction to computerized or electronic telecommunications. The goal was to use computers to automate what had been an inefficient and error prone human process.

The ability to define processes in ways that computers could be used to automate the process and creating resulting efficiencies and improvements in customer service was entering a new era.

In 1960 I was hired by General Electric to work on an automated manufacturing control system project. The purpose at that time was to see if we could design a generic manufacturing control system that could be taught to control a number of specific processes by reusing the core program code base each time, but with some specific modifications as appropriate to the task at hand.

This was an important program to GE since it had a hundred distinct product manufacturing departments, making different products. Each one of those departments was an autonomous business unit. It was autonomous, because it was management by exception. As long as they made a profit, as planned, they didn't get much attention. If they didn't make a profit, they got a lot of attention. So they all operated within carefully planned and reviewed financial plans. The department general managers had a lot of authority. At that time it seemed every one of those hundred departments wanted to install their own customized, manufacturing control system. Their experience revealed that this was a slow, expensive and error prone process.

GE had gone through one system development project to try to move in the direction of more efficient system development. That project developed some interesting things but they were not relevant to the primary goal of getting a generic manufacturing control system. I was hired from Dow Chemical to join that project team as their primary system architect.

RUSSELL: So GE knew of your work at Dow, and decided you were exactly the sort of person that they needed?

BACHMAN: Yes, and more importantly they specifically knew of my work at Dow with respect to the SHARE organization. SHARE was an IBM organization of people who had IBM 701 or 704 series computers, IBM's engineering scientific computers. Based on the work we were doing for Dow Chemical,

we determined that we needed a new machine and were trying to decide which one we should order. I thought that we should order an IBM 709, which was the replacement machine for the IBM 704. Dow supported this recommendation and with the subsequent purchase order, we became entitled to join the SHARE organization. I became the designated Dow representative.

The SHARE organization was supported by IBM but was an independent, volunteer run group formed in 1955 to bring together computing professionals to address engineering problems. The SHARE organization still exists today. At that time, it met twice a year at various U. S. locations.

When I got to my first SHARE meeting, I ran into people mainly from engineering organizations that were operating IBM 701 or 704 machines. I looked around and started talking to people. Some people said they wanted to get into data processing as well as engineering.

At this first meeting, I met Harry Tellier from the GE Hanford Atomic Products Operation. He had just made a presentation about a "report generation system" that he and his people had developed that enabled them to use an IBM 702 to do their business accounting and record keeping. They were there, even though they were IBM 702 users, because they had ordered a IBM 709.

RUSSELL: So anyone who had ordered the 709 computer was eligible to join SHARE? They didn't need to have installed the machine?

BACHMAN: That's right, because everyone had to get ready for their new computer. A lot of work had to be done, and therefore many would be involved with the people who had application experience. The first IBM 709 had not been shipped.

RUSSELL: And IBM funded this?

BACHMAN: Well, IBM <u>hosted</u> it. I guess that in the sense that they helped, they provided a couple marketing people who helped on the organization side. All personal expenses were paid for by the members of SHARE. For example, if SHARE was going to have a meeting in Seattle, IBM would arrange for it. We would pay our own individual travel expenses. If you have enough people come to the conference you can have some conference rooms for free since the hotel makes money from all the room nights and food & beverage service during the meetings. So they supported it, but they tried very carefully not to be too controlling. I think they didn't want to be responsible. If members made mistakes, IBM didn't want to be responsible for the member's mistakes, so IBM didn't have final decision-making power on the things.

One example of a novel approach we took in the SHARE group was led by Harry Tellier from GE Hanford. Tellier had been an experienced IBM punch card installation supervisor, working for the state of Washington, before he joined GE. He thought of his IBM 702 as just a great big IBM tabulator. It

had data files, but they were on magnetic tape instead in punch cards. The IBM 702 could sort, and collate, and tabulate magnetic tapes, just as with punched cards.

Tellier observed that they were dealing with various different reports and that for each new report they had to write an entirely new program. Tellier's analyses revealed that every report was almost like every other report in a generic design sense and differed only in a few specific details. The programs to generate reports all performed a similar set of general tasks. They read tapes and wrote tapes and did this and that.

Tellier and his team finally decided to build a report generator program that would combine a generic, reusable program with other more specific programs for particular jobs. To develop a template for a generic report, they let one of their customers draw a picture on a big piece of paper that looked like the 120 print columns on the IBM printer. The customer wrote on this big form the data he wanted printed and where he wanted it printed. Then for each report line described on the 120 column paper, the customer indicated the circumstance under which that line should be printed. For example, one rule might be that a particular line should be printed only when the first ***Employee*** record, or next ***Employee*** record, appeared on the magnetic tape.

This new approach let the customers, with assistance of the data processing people fill out their forms and specify rules under which certain things should happen. These rules and procedures were keypunched into cards. All of the cards for a report were gathered together into what was called a "packet." Each of these packets, a quarter-inch or half-inch thick representing an independent report were read into the card reader on the IBM 702, all at one time, and the report generator would compile and run them together. Two, ten or more packets, each representing different reports, could be processed while reading the reels of a single master file through the IBM 702.

The program to generate different, individual reports was not written by people. The program was actually written by another master computer program, the 702 report generator. The actual, multi-report program would be executed only once, and then be discarded. Discarded, because the next time that master file was to be processed again, there would be a different set of report packets. Some packets were the same, some modified, and some completely new.

In addition to the cards describing the rules and layout of reports, the 702 Report Generator System would read in a second type of card deck. This second type of deck held the description of a particular master file records. It contained meta data. It provided a name for each field, the size of that data field and the type of data.

It is important to note that the 702 master files should be characterized as "flattened files." A flat file contains information about a basic entity and

supplementary data that provided information about that basic entity and all of its hierarchically superior entities, and their relationships. Flat files are simpler to create and use but less powerful when dealing with complicated data structures and their relationships. Most of the information found in these files was of a hierarchical, or a network nature. Since the meta data describing the data and the relationships was not retained in the flattened file, the persons who created the report specifications must identify primary keys and sorting and report structures for the Report Generator. To repeat, the information structures in the data file had to be flattened so that each master file contained only one type (format) of record, the report generator program must be provided, by the report packet, all of the information required so that a report could be generated and subsequently sorted to recreate all of the hierarchically structured reports that were inherent in the underlying information.

This approach to programming a report led to a great deal of redundant data within each file. It meant that each time a file was passed that a lot of redundant information was passed. Thus, while they were able to successful automate the process, they had not yet figured out how to optimize that automation.

We would do well to remind the reader this was during 1957-58 and that the IBM 702 was IBM's first "data processing" computer. The 702 was designed to replace a lot of punched card equipment that spent most of the time sorting "detail" cards into various sequences and collating them with "master" cards to prepare for the printing of just one report. However, it also meant that a lot of reports with different sequences could be produced in one pass of a 702 master file. This was a huge leap forward in the contribution computing made to business operations. It was the flattened file architecture that made this possible.

For the first time, computer programmers had a powerful new ability for generating reports. An employee file could be processed and generate reports in the same pass, with several different primary sort sequences such as a)***employee-number*** sequence, b) ***department-number*** sequence, c) ***social-security-number*** sequence, d) ***date-of-birth*** sequence, e) ***GE-date-of-employment*** sequence, and a lot of other sequences. So we make one pass across the magnetic tape file, with one, two, or more reels of tape. Today, this might seem trivial, but at the time it was major breakthrough.

As the first chairman of the SHARE Data Processing committee, I had started a new SHARE project in conjunction with GE Hanford. It was called the "9PAC" project, for the "709 Data Processing Package." We originally got people from half a dozen different companies to join this group including Union Carbide, Northwest Power Company, Philips Petroleum, Dow Chemical, and Chrysler. I forget all the other companies who were involved

over time. Essentially, on a cooperative basis, our goal was to develop and share in the effort to build this report generator program.

Later, Harry Tellier tried to hire me to come to GE Hanford. That did not work out, so Tellier recommended me to Ford Dickie, the manager of GE's Production Control Service as a candidate for a new Integrated System project (ISP2). Dickie hesitated over Tellier's recommendation to hire me as he thought I was too theoretical. Tellier pushed back saying I was right for the job.

RUSSELL: GE Manufacturing Services wanted to use the results of this project for their internal purposes at GE. To what extent were they interested in sharing their manufacturing control products with other companies, as you described with 9PAC?

BACHMAN: Well in this case, the "other company" was the same one—it was the GE Computer Department in Phoenix, Arizona. The GE Computer Department looked at this new project as one that might lead to a standard product that they could sell to many companies and get further use out of the research. The driving force came from Hal Miller, VP of GE Manufacturing Services, in Schenectady, New York. We actually had one man, Homer Carney, who was assigned to the project from the GE Computer Department. He came to work with us in New York City. So they had one foot in the door and Carney was very helpful. In fact, Homer and I were the only programmers responsible for developing the first database management system, the Integrated Data Store (IDS), during the first two years.

The system that we created turned out to be a uni-programming, transaction-oriented processing system for GE's Low Voltage Switchgear Department in Philadelphia. We could see what the requirements were; they needed a system that could continuously process a string of transactions: new orders, new factory feedback, new queries, etc. coming in all day and into the night.

One of the successes we had was to program in a priority function setting a rule by which groups of one or more transactions would be processed, as a batch. Once a day, the data processing department processed feedback coming from the factory, as one or more groups, and new orders, as one or more groups, etc. Other transactions were fed in as they became available. When all of the transactions that had been received were queued for processing, an "execute" transaction was submitted and the queued transactions were processed according to their priority. We completed one transaction, and another transaction, and another transaction, in order of their priority and their sequence of arrival, within the priority level. So in this way we had a system that would allow us to run and respond to short turnaround transactions and long turnaround transactions and keep almost everyone happy.

The Problem Controller became a fully fledged, online transaction processing (OLTP) system a little later, but in this first instance the only input device we had was a card reader. So any message you wanted to put into this OLTP system had to be prepared in punched card form, whether it was a new order you were going to process from some other company or a receipt from the factory saying they had completed a particular scheduled operation. The only output devices we had were a card punch and an online printer. This was our communications system. Our online transactions were only online in the sense that the card readers and printers were connected to the computer.

An interesting and useful facility of the Problem Controller was that an application program that was actively processing one transaction could generate one or more new transactions, including their data, for subsequent processing. This was particularly useful when it came time to process new orders and shop feedback. After having initially processed the new orders and shop feedback, it was necessary to execute an incremental shop re-planning activity that included a partial, new parts explosion, for one level. This re-planning had to progress, level by level, looking for the new requirements placed on one level and re-planning that level with the subsequent requirement to re-plan one or more lower levels.

This all worked because the Problem Controller stored the new transactions that it read in through the card reader as transaction oriented data that could be read directly by the transaction-oriented programs from the IDS database. Transaction-oriented programs could originate new transactions because they could store new transactions in the transaction queues into the IDS database.

The Weyerhaeuser Lumber Company in Tacoma, Washington, heard about our manufacturing control project. They had been working with a GE consulting organization, called Internal Automation Organization (IAO) that was based in Schenectady. The IAO people had supported our MIACS/IDS [Manufacturing Information And Control System/Integrated Data Store] work because they believed that they could build manufacturing control systems for people faster, with a really well organized database management system, than they could if they had to program it all by hand.

So the Weyerhaeuser people, in conjunction with the GE people from IAO, selected IDS and its Problem Controller, operating system. The Problem Controller got its name, because the GE slogan for the computer department people was “the problem solvers.” They modified the Problem Controller to resolve a Weyerhaeuser issue.

The Weyerhaeuser/IAO team planned to take their existing torn tape system, and replaced the operators who were routing the messages with a GE Datanet-30 computer. The Datanet 30 was a computer designed to terminate communication lines and handle their analog interface. It was also connected it to a GE-235 computer with IDS and the Problem Controller. The Datanet 30 and its communication lines replaced the card reader, the card punch and the

printer. The Datanet 30 handled all the input and output for the IDS transaction processing center.

This GE235/Datanet 30 combination sat together in the Tacoma computer room. Weyerhaeuser had replaced all of the manual processing that had been done on their order processing system. Take the task to process customer orders, for example. They could run it with an IDS direct access database system. They could have an online product inventory for every lumber mill and warehouse. They did not have to go out and ask the people in the mill what was in their warehouse or mill inventory, they could check their inventory by querying the IDS database. They could check both the physical inventories and the net inventories by considering the shipping commitments already made. The centralized computer in Tacoma, Washington was serving all the mills on the west coast and some in the southeast, because they also had paper mills down there. At the same time it was serving sales offices across the country and the Tacoma headquarters' offices.

RUSSELL: How long did it take to switch out their old system with the new one?

BACHMAN: I think it was done in less than a year.

RUSSELL: Wow.

BACHMAN: Let me go back and be a little careful. Let's see, they started on the project before IDS was originally shipped in 1964. In late 1965 they were up and operating a new business system. But it was the same old IDS and Problem Controller, except that the transaction data records were processed as paper tape images rather than as punched card images.

Please let me jump ahead a couple years. I was told the following story by the Weyerhaeuser VP responsible for IT. The Weyerhaeuser VP said that, at one point in time, they had so many orders coming in daily that if they started Monday morning with empty transaction queues, and the orders started coming in, the GE 235 would start processing them in a priority sequence. The whole day the application programs were processing orders, and continued into the night while all of the sales offices, and the warehouses, and the administrative offices were shut down. When the next morning came, the Problem Controller might be still processing some of the transactions that had come in the day before. This might go on all week long. When Saturday morning came there might still be a backlog of work to be done. All day Saturday, all Saturday night, all day Sunday the computer sat there and worked off these orders and other transactions. Fortunately, when Monday morning came, the transaction queues were empty. So they said, "Wow, we're making it—let's just keep surviving here."

RUSSELL: Barely making it.

BACHMAN: But the VP said, at the same time the computer was receiving and promptly processing higher priority transactions. The most critical

transactions being received were requests to establish a new customer record. That's the most important part of the business, because if you have a brand new customer, you cannot enter an order for that customer until you have a customer code assigned to it. You must first send a transaction saying, "Please give me a customer code for a new customer, and here's his name, address and his other information," so you could build a customer record.

And zip, here was a return message stating that a new customer was recorded and there was a customer code you could now use. That could be done in ten or fifteen seconds, because that transaction was the highest priority new transaction. New customer registration would begin as soon as processing of the current transaction was completed. The problem controller that was controlling the flow of transactions knew when to jump and when to say "Peace, peace, you'll get your turn." But, that actual order for the new customer might not get processed until the next day because the computer was not keeping up with the current flow of orders. However, the field salesman who entered the order had completed his immediate task and could start thinking about his next task.

RUSSELL: So the high priority stuff was fine but the low priority stuff…

BACHMAN: Well, that's why they had high and low priority transactions. So that they could give some transactions a chance to bypass lower priority transactions, like an express train could go past on the main line, while the slow freight waited on a siding.

One of the things Weyerhaeuser changed in the Problem Controller was the means for controlling the priorities. They wanted to be able to manage the priority of various transaction types, so that the priorities at one time in the day might be different than priorities at another time.

The original Problem Controller had been designed and built ignoring a basic systems design rule. "Never use a single data value as both a primary key and as an application meaningful value, because you will never be able to modify the value of the meaningful value." I had made that classic programming error. The ***problem-type-code*** was serving as both the primary key for finding the ***Problem-Type*** records and also as the sequence number controlling the processing priority for that transaction types. And one reason you might want to run something as a higher priority at night is because there was a lot of set-up and tear-down to run that type of transaction, and if you have a bunch of transactions of the same transaction type, the Problem Controller was smart enough to observe whether the next transaction was of the same transaction type, and therefore keep the transaction processing program in memory, and also keep in memory the content of the virtual memory buffers. Do not flush them out at the end of each transaction. You did not have to purge the machine and refresh it. It would just keep on going to the next transaction of the same type until its input queue of transactions was emptied. The original Problem Controller (1964) maintained a single list of ***Problem-Type*** records

that were sorted in ***problem-type-code*** sequence, which was also used as the dispatch priority sequence number. Weyerhaeuser said, "We're going to add a ***priority-sequence-number*** field to the ***Problem-Type*** records. We will define a second list containing the same set of ***Problem-Type r***ecords and sort that list on the new ***priority-sequence-number*** field. The new list is going to be the dispatch priority sequence list. The original list is going to be the problem type sequence list."

When it is desired to alter the ***priority-sequence-number*** of a ***Problem-Type*** record, a system control transaction is submitted that identifies the ***Problem-type*** record(s)***,*** whose ***problem-sequence-number*** field is to be changed and supplies a new value for subject ***problem-sequence-number*** field. That system control transaction is no different than any other transaction. It is selected for processing in its specified ***problem-sequence-number*** sequence and is dispatched for processing like any other transaction.

If the value of a ***sequence-control-number*** field for a list is modified, IDS will remove impacted ***Problem-Type*** record from its position in the new sorted list and reinsert it again in its correct new position.

It is worth pointing out, that the ability of the IDS DBMS to maintain all of the records of one record type, simultaneously, as members in two different, sorted sequences, is extremely important. Important, dealing with complex business situations. This where the network-oriented DBMSs are without rivals. In this example, it was all of the instances of the ***Problem-Type*** record.

So Weyerhaeuser corrected the mistake in my design. That's what they teach you the first week in system design school; never assign the data field of a meaningful code to also be a unique key. Primary keys are just unique keys. They shouldn't be used for any other purpose. Please!

RUSSELL: And so you learned.

BACHMAN: I learned, yes. And they achieved the ability to dynamically make priority dispatching changes very easily. Even an application program could make the same type of change if it became necessary. But anyway, they had their online transaction processing (OLTP) system running from the East coast to the West coast, from North to South, at mills, administrative offices, sales offices, running transactions and running completely "hands off."

RUSSELL: It took the people right out of the loop.

BACHMAN: One of the stories they tell about that the WEYCOS system is that it drove the operators up the wall, in the Weyerhaeuser computer room in Tacoma. It drove them crazy, because there were no input or output devices on that computer. There was nothing to feed or unload. All they could do was look at the computer console or the disk file controller and see whether

the lights were blinking, and question "Is it in a loop, or is it working? What's going on?" And they just felt very insecure.

RUSSELL: I can imagine.

BACHMAN: So at one point Weyerhaeuser management said, we're going to attach an electric typewriter to the system and once every hour we're going to print out the transaction queues. If the lists are different than they were the prior hour, then don't worry about it. If it hadn't changed you got a problem. So they quieted the operator problem with this hands-off automatic system. It was really terrorizing the computer room operators, because they were responsible for it, to make sure it was running right.

RUSSELL: But there was nothing they could do about it!

BACHMAN: Well, they could stop it and restart it, but, they didn't know whether to stop it or where to restart it.

RUSSELL: Right.

BACHMAN: So if you're being a good manager that condition shouldn't be allowed to continue.

Weyerhaeuser and WEYCOS laid the foundation for part of my background in communications. The pseudo OLTP system was created for MIACS (1964) and that was followed by the real OLTP for Weyerhaeuser (1965). Now from that point, one of the significant things that happened is that GE in its wisdom decided that they still did not want to be in the computer business. In fact, the truth is corporate headquarters never wanted to be in the computer business. But they were quietly pushed in by a group of GE engineers, in California, almost without telling the GE corporate headquarters. The GE people in California bid on a new banking system for the Bank of America, and they won the contract over IBM, and Burroughs, and Univac, and that was an embarrassment to corporate headquarters who was not aware of the bid. They had to sign off on the Bank of America contract. This came in 1959 with a multi-million dollar contract that they did not know how to handle. At that time Ralph Cordiner was the chairman of GE. Cordiner was well known for a business policy that says, "If we can't be number one or number two in a particular industry, we don't want to be in that industry. Manufactures number three, number four and five, etc. can't make a profit."

That policy was based on a sound business model. Of course GE was not the second or first in computer industry. At that time GE was not anywhere in the computer industry. IBM was first and it was IBM, as Snow White with the six dwarves (Burroughs, Univac, NCR, etc.). This contract was awarded to GE, making it the seventh dwarf. Cordiner was stuck with this new business, and GE's Computer Department was stuck with GE's standard bookkeeping that was showing yearly losses. From a long term point of view it was printing money, it had contracts on computers leasing them for a five year period. The

standard GE accounting rules did not evaluate long-term leases appropriately. Cordiner and his successor, Fred Borch just did not feel right about how the computer business fit in with the rest of the GE portfolio.

Nonetheless, in 1969 the General Electric Department, now a Division, started a major project identifying products that would be produced in France, Italy and the United States. This was to be the "New Product Line" (NPL). The projected engineering and manufacturing costs and delayed income streams were huge and unacceptable to corporate headquarters. The projected profits, using standard GE accounting practices, were minimal. So Fred Borch, CEO and Chairman of GE said "Let's see if we can't find somebody to whom we can sell this thing."

RUSSELL: The whole division?

BACHMAN: The whole division. They understood that Honeywell was also struggling with being one of the seven dwarves. Borch said something like this to Honeywell, "If you were twice as big, you might have a better chance of surviving." So, GE sold the computer business. They sold the business for Honeywell's stock, which GE couldn't sell for a period of time, maybe it was five years. So that it wasn't a cash flow problem for Honeywell and they had a chance to either prove themselves or scuttle themselves.

RUSSELL: But either way GE was out.

BACHMAN: GE was out with no further losses, except they held all of the Honeywell stock. So they still were on the hook, and could not recover any past losses.

RUSSELL: Right.

BACHMAN: They couldn't sell the stock, and they couldn't liquidate the stock until the time period was over.

Incidentally, at the end of the five years lock-up period, Honeywell stock was way up and GE came out smelling like a rose.

RUSSELL: So then when you moved to Honeywell, did you have a new group of colleagues?

BACHMAN: Some new and some old, because some GE people came East from Phoenix. I had been in Phoenix working with the computer division. The corporate headquarters for Honeywell Information Systems was in Waltham, Massachusetts. In Phoenix, I had an advanced development group and the people were offered opportunities to move to Waltham. Some people chose to move and some people didn't. But that's where the action was going to be. So I moved to the Honeywell Information Systems office in Waltham. Our family moved to Boston in the late summer of 1970 and bought a house in

Lexington. At that point in history, our four children were aged ten to eighteen.

I had been working on the GE NPL project before the merger. The GE project was merged with a similar project that Honeywell had been working on.

Let's go back to the communications theme. Three years later (1973), when my NPL responsibilities were completed, Honeywell was still willing to consider the communication add-on to IDS, so I started on that project and called it "Distributed System Interconnection." And so I took on the project, which I had started in Phoenix, under the name of "Inter-Communicator". I had moved to Honeywell and had a chance to work on this project full time, with an advanced engineering group in Waltham, near Boston.

I started looking around for useful solutions to similar problems and became familiar with the IBM project called SNA or System Network Architecture. It had some very attractive features. It looked like it was well-architected. It had six architectural layers. It had a principle that each layer only talked to the layer immediately above and to the layer immediately below it.

A layer might be modified, or replaced, and its protocol changed, as long as it maintained both its upper interface and its lower interface. And so it was possible to swap out a layer with an improved protocol and a new implementation that would support different things without disrupting how the other layers functioned.

IBM's system network architecture had six layers with a physical layer at the bottom with one protocol supporting a pair of twisted wires. The top layer is where application programs operate. It is the layer where someone operating at a teletype machine or a computer terminal is located. It is where an I/O device would exist with a monitor and a keyboard so you could enter or display data.

I'm going to stop here for a moment and say I'd been involved earlier with a project to standardize the IDS architecture. I'd been involved earlier in that with the CODASYL Data Base Task Group (DBTG) whose purpose was to document the IDS architecture and its Data Description Language (DDL) and its Data Manipulation Language (DML), which might be called "IDS2." The CODASYL DBTG bought into the network data model idea, but had not been able to move it toward, as a candidate for acceptance as a national standard.

After IDS became operational at a number of GE departments, it came to the attention of a man named Warren Simmons from US Steel in Pittsburgh. Also, we had been using computer time on the GE 225 computer installed at one of the US Steel plants. We were using it because it was one of the earliest GE 225 installations that had a disk file instead of the usual tape files. We were debugging IDS on it. Simmons became familiar with IDS, and being a forward-planner, said, "This is something that CODASYL should look into."

Simmons was the chairman of the CODASYL's Planning committee. Simmons was widely known in the computer community and had connections with ANSI/SPARC and the X3 committee.

RUSSELL: And X3 is the Computers and Information Processing committee for ANSI at the time?

BACHMAN: Yes. It was the ANSI/X3/Standards Planning And Requirements Committee (SPARC) field of operation.

Now a preamble to this, we tried to talk GE management into patenting IDS. At that time software had never been patented, indeed it was not patentable by any of the US Patent Department rules. We talked to the GE patent department and the patent department said, "Yes, this obviously is an invention," but there would be a lot of work to get the first software patent, and this and that, and we don't want to be bothered. So they just turned their hands down on patenting it.

RUSSELL: And so did the GE Computer Department decide they could profit from it promoting the IDS system as a standard and getting everyone to adopt it?

BACHMAN: We decided that the best way to use it was as a marketing tool. And you can't be a marketing tool, really, if it's a private, secret concept. So we publicized it everywhere we could. And so when Warren Simmons came along and said "Let's just organize a CODASYL activity to standardize IDS," that was exactly what we wanted, because if it could be a standard the GE version would be the product implementation out first in the marketplace. It would have real advantage being the first product based on the national standard. And we would have field support that was already experienced in installing it. So Warren got us kicked off and helped build what became called IDS2. IDS or IDS1 would be a reference to the original GE product. Simmons was the founding chairman of the CODASYL DBTG.

We expected IDS2 to be very much like IDS1, but different, having gone through the standardization process. There would be enough things entered or modified in it to make it so it wasn't exactly IDS1. And it turns out that committees work like that. Everyone now has to have their "hump on the camel's back" (the specification). They wanted their share of ownership of the new specifications. And so, sometimes the names of the verbs got changed. Sometimes something else got changed, and so it was very similar but not the same.

RUSSELL: But from your standpoint, having them interject different words, wasn't that big a deal?

BACHMAN: Well, it wasn't that big a deal because the IDS that was in the process of being implemented for the GE-400 line of computers and the GE-600 line of computers was not yet complete. Those computers had not been delivered.

So if they wanted to change the names of verbs, it's no big deal at that point. The basic integrity of the IDS concepts was intact.

It is far more important that those verbs are not cooked into your customers' application programs that are in production. And there were a few other tactical things that had changed. GE had to make those changes before IDS—GE was compatible with the CODASYL DBTG specifications, but those were never standardized, because IBM pushed back against it and IBM was the biggest force in the industry, because of their dominance of the computer market. IBM was always aware of the IBM customers that were on the various committees, along with the IBM employees. IBM had a predominance of the voting power in the committees. They couldn't destroy the IDS standardization effort, but they could slow it down in various ways.

While no national standard was produced by our efforts, it did produce a carefully structured specification for IDS2 (although it was not called that) that was spread throughout the industry and which was used to create IDS2-like products that were introduced into the US market.

That happened back in the last half of the 1960s. And so by the time I got to Honeywell in 1970, I went back to try to get IDS2 standardized again. We actually formed an ANSI-SPARC subcommittee (Study Group on DataBase Management Systems) to reexamine the standardization of database management systems. It turns out we had the same problem as the earlier CODASYL DBTG. The people who represented IBM and the people, who represented IBM customers, were not interested in standardizing the IDS concept, but they were willing to talk about other interesting problems.

Collectively we agreed on establishing a global architecture over all the functional pieces of the database system. There is a big report I wrote on this subject. Then we settled on a second problem, the problem called, "data independence." The underlying problem is how to exchange information between different business systems that have data stored, concerning the related information, when the data is stored in two incompatible formats. It might be ASCII data versus EBCDIC data. It might be IDS data versus COBOL flat file data. It might be GE data versus UNISYS data. It might be Metric data versus English Units data. And of course it might be differently named data.

Here's the problem at hand. I may have a database formatted in one way with a certain number of data fields in it, and somebody else has some other system that uses data that may have come from some other place, it had different formats or different names for things, the question is how do I match up the things where the information content is consistent, how is the formatting done consistently? And how do we match them up?

We ended up with what was called the "Three Schema Approach." It consisted of an external schema, defining the way users look at the

information. There was an internal schema, defining the way that the database people look at the same information. Then we created a third thing called a conceptual schema. The conceptual schema was the new thing. It was semantically a richer data model after all of the implementation and optimization meta data had been stripped away. It was and is an all-knowing creature.

The conceptual schema had a consistent means of understanding all of the data modeling concepts used in all of the internal and external schema. It knows that all the kinds of information the schema represent and it has a defined map for translating internal and external metadata into conceptual metadata.

To make this concept work, it was necessary to have an internal schema for every database and data file that was to be brought under the control of the conceptual schema. In addition, it was necessary to create a map between each element of these internal schemas and its corresponding element within the conceptual schema.

Furthermore, it was necessary to have an external schema for all of the input/output data structures that were to be brought under the control of the conceptual schema. Finally, it was necessary to create a map for each element of these external schemas that identified its corresponding element within the conceptual schema.

And so, now that there was: a) an indirect path between every external schema and every internal schema, by way of the conceptual schema, or b) a direct path between every external schema and every internal schema. The direct path could be generated using the same information used by the indirect path. The three schema approach supplied enough meta information that you could compute a program that would translate internal data directly into external data, or external data into internal data, using the internal schema, the conceptual schema, the internal schema, and maps at program generation time. The translation programs could be generated for each internal schema versus external schema pair, as needed.

RUSSELL: And so IBM took part in this, or they were supportive of it?

BACHMAN: IBM supported that. That was good stuff.

RUSSELL: And wouldn't undermine their position.

BACHMAN: It wouldn't undermine IBM's dominate position. The ANSI-SPARC item I pulled out of the Wikipedia last night describes what it started out to be and what it ended up being.[3] It didn't end being a generic overall view of all

[3] Wikipedia, "ANSI-SPARC Architecture," http://en.wikipedia.org/wiki/ANSI-SPARC_Architecture (accessed February 20, 2012).

database systems. That was set aside. And we became the Three Schema Approach people.

RUSSELL: So is it fair to say that, in different ways, both the SPARC work and the CODASYL work were important for the industry rather than any single company?

BACHMAN: Yes, both projects were important for their impact on the industry. The first CODASYL DBTG project launched IDS2 and IDMS, the version of IDS implemented, by B. F. Goodrich, for the IBM 360 mainframe. They and other DBTG products have had an important long term impact on the industry. Today, there are more than a thousand IBM mainframes running IDMS OLTP systems.

The later SPARC DBMS project launched the Three Schema Approach that has been picked up and applied in situations where the issue of data independence is critical. We will talk at length later on about how the data independence problem appears in the ISO/TC97/SC16 Open Systems Interconnection world. Solving the data independence problem was the reason for the introduction of the Presentation Layer.

RUSSELL: How long did the SPARC DBMS Three Schema project take?

BACHMAN: That was a two year project. I think that it would have been completed more quickly, except for competing sub-rosa projects. Some people on that ANSI-SPARC SGDBMS committee, who were very competent people, had other interests that sometimes, seemed to be more about visiting interesting cities and eating good food. I think it lasted longer than it needed to have. The work was finally written up by a professor at the University of Montreal, Dennis Tsichritzis**.**

RUSSELL: I should say for the record that John Day, our host today, is here with us in the room. He's been popping in and out, and so if there's another voice on the transcript that's who you're hearing. Day was an active member of the ANSI/X3/DISY committee supporting the ISO/TC97/SC16 Open Systems Interconnection activities.

BACHMAN: When the SPARC SGDBMS three schema project came to an end, I was able to turn my attention, full time, to the idea that Honeywell should develop a software subsystem to focus on the communication side of the application programs rather than the database side. As I mentioned earlier, in looking around I came across the product that IBM had publicized. They called it, "SNA," or System Network Architecture. It was a multi-level architecture. It looked to be well-conceived, well-documented, but it had two serious flaws, from my point of view that meant it could not be readily adaptable to Honeywell's requirements. First of all, it was designed to support, a hierarchical structure of commuters and terminals, with an IBM mainframe at its root; nevertheless, it was called a system network. An IBM

mainframe would be at the center of a corporate or divisional data processing system. That mainframe was connected to subordinate computers, terminals and printers. Secondly, IBM's SNA did not solve the problem of data independence and incompatible data formats within communications.

The requirement to have an IBM mainframe at the root was not good from a Honeywell point of view, because Honeywell's customers may have Honeywell computers installed along with IBM computers, Univac computers and Burroughs computers, and other computers and peripherals.

Honeywell needed an open architecture, where no one computer was king. We're all princes in this world together and we can pass information around and be buddies, as long as we all play by a standard set of rules and protocols. Further, communication between computers operated by different companies and other organizations was essential. Supporting a true network of computers, processes, and people was essential. Support for his kind of open configuration did not appear in the IBM SNA solution.

Having just come off the three schema, data independence project, the ISO communications project we talk about here is closely allied with the ANSI SPARC SGDBMS solution. Various "sources" and "sinks" were generating messages and receiving messages, transferring the same information, but using different data formats. So we have the issue of data independence problems within communications. If you have different companies and different computers, you have different formats everywhere. So we must address the data independence problem again. The mechanism to solve that data independence problem did not appear in the IBM SNA solution.

The six layer high SNA graphic representation looks like a six story building. It is more realistically depicted as two six story buildings spaced apart with six levels of communication protocols joining them: the first layer of the first building is communicating with the first layer of the second building, the second layer of the first building is communicating with the second layer of the second building, etc. until the sixth layer of the first building is communicating with the sixth layer of the second building. Only the first (lowest) layer processes (Physical Layer) directly communicates with its first layer peers. All of the other layers communicate with their peers by passing communication objects down to the layer directly below. The receiving layers unpack their controlling information and passes it's text up to the next higher level.

In the work for Honeywell (Honeywell Information Systems), we raised the Application Layer up one layer, to layer seven. Then we inserted a new layer six, the Presentation Layer, just above the Session Level. That Presentation Layer is where the data independence problem is resolved.

The new sixth layer functionality would be null or empty, if application process A and application process B, are speaking the same language, with the

same formats. If the two applications were not directly compatible, then required translation software would be installed in the Presentation Layer. There is a choice of where to convert from format A to format B: 1) on the sending end, or 2) on the receiving end, or 3) convert format A to the canonical conceptual schema-based format at the sending end, and then convert the data, in its conceptual schema format, to the format B at the receiving end. If two way communications were involved translation would be required in both directions.

I did not know why IBM had not addressed the data independence problem within SNA, and I was not going to ask them. Possibly they were thinking about cases where all of the programs and computers were exclusively IBM products.

With these changes and additions, the resulting Honeywell communication concept was labeled, "Honeywell Distributed Network Architecture (HDNA)."

RUSSELL: It's interesting that you were so deeply aware both of the technical implications and the strategic implications of your work.

BACHMAN: Yes. Today we talk about the Internet being worldwide, and that's the power of it. Anybody who plays by the rules can connect. Well to this point Honeywell started out with basically the same rules. The projected new Honeywell communication system had to be an open system for anyone who played by the rules, layer by layer, for both the interfaces and the protocols. Having an open system was a critical to the Honeywell business plan.

RUSSELL: At what point did you become aware of the ISO work on open systems? I would like to know what role the ANSI-SPARC played in the OSI work at ISO, and how quickly you remember the work progressing.

BACHMAN: The International Organization for Standardization (ISO), Technical Committee 97 (TC97), meeting in Sydney, Australia (1978), voted to establish a new committee to solve the problem of open systems interconnection. It is in my mind that the U.S. delegation to TC97 voted against the issue at that meeting, because IBM was against it.

RUSSELL: And there was an existing committee, SC6.

BACHMAN: That was the data communication committee with scope over the lower half of the contemplated architecture. It was the domain of the telecommunication industry, and we became Sub Committee 16 (SC16), with scope over the full range of the architecture.

DAY: Yeah, but there was an earlier meeting of SC6 where the proposal for an open systems interconnection (OSI) project had been rejected.

BACHMAN: Oh, SC6 rejected it?

DAY: That's the story I've always heard.

BACHMAN: Well I think the proposal went to TC97 at its next meeting.

DAY: Then it went to a subsequent TC97 meeting, which reversed SC6 and authorized SC16.

BACHMAN: Yes, well I think that TC97 meeting of the reversal was the one that was in Sydney. And it turns out that the U.S. delegation, having voted against creating a new Sub Committee 16 in Sydney, then said that the U.S. wanted to be the SC16 secretariat. The secretariat is the nation which is responsible administratively for keeping the records and has the right to establish the chairmanship of the committee. That was very important in terms of the politics of how things go forward.

Honeywell became aware of this action, and knew that ANSI/SPARC was going to set up its own committee to handle the U.S. (ANSI) side of SC16.

The Honeywell corporate staff people in Minneapolis, who covered Honeywell Information Systems, didn't think Honeywell should participate in SC16. They had supported the Honeywell Distributed System Architecture program that I had been working on. I do not know the reasons that caused them to reject the idea of participating in SC16. I thought it was very important for Honeywell to participate, because the HDNA project that I'd been working on for Honeywell was very close to what I thought ISO needed.

I called Warren Simmons at US Steel. He was the chairman of CODASYL planning committee, whom we had talked about earlier. He had been very instrumental in the standards effort on COBOL and later on database management systems. He had been frustrated about the fact that we never got a national standard out of the CODASYL DBTG work. Warren had many contacts and I thought he might have some influence on ANSI/SPARC decisions. I said, "Warren, I want to participate in the SC16 work, and, if it makes sense to you, I'll volunteer to be chairman of the American SC16-oriented committee." He said, "Let me see."

SPARC was the administrative group setting up the U.S. counterpart of the SC16 committee. It would be inviting each of the companies who might want participate on this new committee. I knew that if I were offered the job as chairman of the ANSI/X3/SPARC - Distributed Systems, Honeywell would have a hard time saying, "No." So, I was asked to be the chairman. It wasn't an IBM person. The IBM people who did join the SC16 work were lower-level communication people, SC6 people.

RUSSELL: So then was it ever any trouble after that to get Honeywell to pay your expenses?

BACHMAN: No, Honeywell was fine with my appointment. I wrote first draft of the ANSI committee proposal that we circulated. It was very close to the Honeywell HDNA document.

I'm going to jump ahead for a moment. When I left Honeywell to join Cullinane Database Systems, three years later, I wanted to take all my personal files. Honeywell impounded all the communication-oriented files. My manager, Mike Canepa, said, "You can have them later, but you can't have them now. There's too much company information in them." And by the time came around to pick them up, Honeywell couldn't find them. I had the most complete set of the ISO SC16 files anybody had, with all the intermediate drafts and all the working papers. They all had disappeared. I don't know, whether it was malicious, or careless, or, what. They were not to be found.

RUSSELL: That's very painful to hear since historians depend on those sorts of documents.

BACHMAN: There were two file cabinets full of my papers, concerning Honeywell Distributed Network Architecture, ANSI/X3/SPARC–DISY, and ISO/TC97/SC16–Open Systems Interconnection, including lots of proposals that didn't fly. But, they were all part of the history: what worked, and what did not work.

Some time when we get into the subject I want talk about one of the things that was suggested by the Japanese delegation in the SC16 committee that I delighted to accept and it turned out to be very productive. We'll come back to that later. They suggested a very simple improvement in the graphic presentation of the block representing an entity type. It was an important improvement.

Okay, you want to look at the files you have there and see if they're any help?

[Pause for Bachman and Day to examine copies of OSI SC16 documents from Bachman's papers at the Charles Babbage Institute.]

RUSSELL: One reason why I brought this[4] out and thought it interesting is that it speaks to what you've been talking about, namely, the relationship between your work at Honeywell and Honeywell's corporate strategy and their position in the industry—including national standards through ANSI and international or global standards. And so, as you said, IBM's SNA model was out there before you articulated your own version of a layered model.

BACHMAN: Basically, we added one layer to the IBM SNA, we inserted a new layer.

---

[4] Charles W. Bachman to Honeywell Distribution List, "ANSI Reference Model – Distributed Systems," February 6, 1978. Box 19, Folder 6, Charles W. Bachman Papers, 1951 – 2007, CBI 125, Charles Babbage Institute, University of Minnesota.

RUSSELL: Right, and there were a number of other layered models out there as well, or a number of other groups working on models. ECMA [European Computer Manufacturers Association] is here. They seem to have proposed a similar layered model. And then there were other communication architectures out there, such as the Arpanet. I wondered how much these were on your mind—if at all—when you made your proposal to ANSI?

BACHMAN: Well, I guess the answer I can best give you is that the only two that were figured strongly in my mind, are just the Honeywell distributed network architecture (HDNA) and the IBM SNA. And I think the way the existing military communications were running, their architecture was very similar at the lower levels where SC6 had been working over the years.

[…]

RUSSELL: From your standpoint then, when you submitted the model first to DISY and then eventually to ISO, the SNA and the Honeywell layered models were what you were working from.

BACHMAN: That was the driving force I would say.

RUSSELL: Okay. It seemed like a consensus around your model developed very quickly in late 1977 and early 1978 among the folks in DISY, the American group, and then the seven-layer model was very quickly forwarded to SC16 in ISO. Was there much dispute amongst the American delegation before sending it on?

BACHMAN: We didn't have much time, and in the first order it made sense. You know it certainly was very close to the IBM model except for the changes that were obvious in their need.

RUSSELL: Right. Then you were very quickly appointed Chairman of both the American delegation and of this ISO committee, SC16.

BACHMAN: Yes.

RUSSELL: I wonder if you could talk about that.

BACHMAN: All I can say is that my offer to be the chairman was made to Warren Simmons and the request to me to accept the chairmanship came back from ANSI/SPARC DISY.

RUSSELL: And then what about at the international level?

BACHMAN: The secretariat nation has the right to appoint the chairman of subcommittee. The U.S. had been appointed as the secretariat of ISO/TC97/SC16. ANSI/X3/Study Group Distributed Systems had appointed Charlie Bachman as its chairman and consequently, as the chairman of ISO/TC97/SC16.

RUSSELL: Did you get the sense that the British also were interested to lead the process in ISO?

BACHMAN: Well, they were interested in being the SC16 secretariat also. They felt they were well prepared to do that.

RUSSELL: Did you sense any frustration from them once they didn't get secretariat and chairmanship?

BACHMAN: In my memory it wasn't a big deal when it finally was decided by TC97, which is the section of ISO that deals with computers and communication. I think that was IBM politics or I think it was U.S politics. You know anything different than that, John?

DAY: No, the U.S. held the chair of TC97, as well.

BACHMAN: Okay. It was probably for the same reason.

RUSSELL: Let's see. I wanted to ask about CODASYL (Conference on Data Systems Languages),[5] because it looked like a lot of people involved in the ANSI/SPARC distributed systems group had also been involved in CODASYL. Were there any lessons you learned in CODASYL that you could apply to SPARC committees?

BACHMAN: Well, CODASYL made its greatest contribution through the creation of COBOL (Common Business-Oriented Language).

In 1965 it created the CODASYL List Processing Task Force (LPTF) that was renamed the Data Base Task Group (DBTG) in 1967. Warren Simmons was the founding chairman of the LPTG. The DBTG was a group that worked to produce a specification of an IDS-like DBMS that would be a candidate for a new standard. We have discussed their successes and failures earlier.

The communications group tended to come out of the SC6 background, you know, because there was a long established need for standards for telegraph and telephone. So that—I don't know whether it's just my brashness or what—I thought I could work that field well. I'd been working the field well, so I already re-branded myself, or made myself into two brands, database and communications, before this issue came on the table, because it was a missing piece.

DAY: Yeah, SC6 had not been in existence all that long either.

BACHMAN: I don't know how long that—

DAY: I don't either, but it's not as long as you might think.

---

[5] Conference on Data Systems Languages, which worked to create standard versions of the COBOL programming language and databases in the 1960s and early 1970s.

BACHMAN: Was there an IEEE group doing that version?

DAY: No, no.

RUSSELL: Did you take part in IEEE standards committees?

BACHMAN: No, I was never involved with the IEEE, except that I wrote an article, "The Origin of the Integrated Data Store, the First Direct-Access DBMS," for the *IEEE Annals of the History of Computing*, two years ago.[6]

RUSSELL: What about the Association for Computing Machinery (ACM)? Did they have standards committees at the time, in the 1970s, that you were involved with?

BACHMAN: Not in the sense of a national or international standard. In 1973, I was given the ACM's highest honor, The "ACM Turing Award."

RUSSELL: Okay, so it was mostly these ANSI and ISO committees?

BACHMAN: Well, there were national and international organizations country by country which handled internal things and external things, respectively. There was a lot of cross pollination between the internal and external efforts, because the internal people represented themselves externally.

RUSSELL: There was another organization that I saw in your files, IFIP, the International Federation for Information Processing.

BACHMAN: Well, they ran conferences and they tended to be educational, to help people, and conferencing was one mechanism that they used. They published books.

RUSSELL: I see, but quite separate again from the work involved in setting standards.

BACHMAN: Yes. Maybe some of the same people involved.

RUSSELL: Sure. My next set of questions all have to do with what happened in SC16 once things got started. You were the Chairman, and the first meeting was February 28 to March 2, 1978, in Washington, DC. You've said that things were already moving quite quickly. What's your recollection of how the first meetings went and how quickly the Reference Model progressed from there?

BACHMAN: I don't remember much detail about those meetings, because there did not seem to be any great disagreement in the direction I thought was appropriate to go. Everyone seemed to get on the bandwagon. We always had trouble deciding details of text specifications, and we did some of the same

---

[6] Charles W. Bachman, "The Origin of the Integrated Data Store (IDS): The First Direct-Access DBMS," *IEEE Annals of the History of Computing* 31 (2009): 42-54.

things we did with the IDS work, where people added their "hump on the camel," but typically not to the detriment of the general direction.

RUSSELL: One of those discussions was the term that the ANSI-SPARC group used, "distributed systems." But the British came in with another term, "open systems." It seemed like two different ways to refer to the same thing, but were there any significant differences?

BACHMAN: Well, I would guess that was one of those "humps on the camel," and it wasn't an important—it wasn't worth a lot of argument if it's keeping you from going forward on the real job.

RUSSELL: So then very quickly there's broad agreement that the Reference Model should come first and not any specific standards.

BACHMAN: Yes. The reference model had to come first, so that you would know how to plug and play. John Day certainly has expressed strong thoughts about this point, whether or how relevant the Reference Model is, or should have been, or ever was.

RUSSELL: ANSI and ISO usually standardize technologies that already exist, right? And so this project was a little bit unusual, because it was trying to standardize into the future to make so-called "anticipatory standards." Do you remember that causing any tension at all in the operation of the committee?

BACHMAN: Not to any great degree. It sounds like a good standard argument, but, from an architectural point of view, you might say we built the foundations before the buildings and we've built the walls before, but no one had ever built a roof like this before. Well, sometimes the architect's job is to envision a buildable roof, not just a roof, but a buildable roof. That doesn't prove it's been built before. And so this, just putting the Presentation Layer in this was something that had been worked over and had not become a standard, but had become an accepted architectural feature. It was needed and you drive it up and plug it in.

DAY: Well, and of course the other thing was that if you were you going to be doing these things in parallel you had to have something that coordinated the work, so that there wasn't overlap amongst the work. And so the Reference Model was that. It was the control document.

BACHMAN: Well, the lower levels standards were in place, as they were existing practice.

RUSSELL: Right. So in a sense that was standardizing what was already in operation.

BACHMAN: But it's amazing how many times you can rewrite those sections about the same things that everybody knew about. The trouble was getting them accepted by everybody.

DAY: Well there was a controversy over defining interfaces, and also the layers.

BACHMAN: Yes, there has to be.

DAY: Well, no, I mean the reason we had service definitions as opposed to APIs [application programming interfaces] was because there was a lot of that standard APIs at every layer would require the vendors to expose those APIs to users.

BACHMAN: You mean the protocol?

DAY: No, I mean the interfaces.

BACHMAN: The interface specification. One of the things that I spent some of my time on was trying to introduce data structure diagrams to define the objects of discourse across the interface, as a step toward defining what the interface specification was all about. And we still were not as clear as I would like them to be.

RUSSELL: Speaking of interfaces, I wanted to ask you about the numerous liaison relationships that SC16 set up once it started meeting in 1978. For example, SC16 documents mention CCITT…

DAY: CCITT[7] was a separate parallel organization working on all the layers.

RUSSELL: But there were liaisons eventually created between the groups?

BACHMAN: Not that I'm aware of, but then that—you know anything about that? Liaisons between SC16 and who?

DAY: Yeah there were CCITT liaisons to the meetings.

BACHMAN: Okay, I don't remember that.

DAY: You have to check the timing, but CCITT Study Group 7 established a reference model group fairly—at some point along—not too long after all this starts, 1979, you know, because the merger comes in 1980.

BACHMAN: Now which merger are you talking about?

DAY: Making it a joint project.

BACHMAN: You know I don't remember that at all. And I was still there.

---

[7] International Telegraph and Telephone Consultative Committee, a branch of the International Telecommunications Union that created standards for telephone networks and was beginning to create standards for computer networks.

DAY: Well, before the reference model went to DP [Draft Proposal], it was a joint project with CCITT.

BACHMAN: I don't remember that. It must have been so easy or so difficult or one way or the other that I don't—

[…]

RUSSELL: We're looking at your paper, "Domestic and International Standards Activities for Distributed Systems."[8] On the second page, you mentioned that there's a lot of possible and obvious enormous benefits, but the technical problems and the political problems are substantial as well. I wonder if you wanted to comment on that any more than what's in the piece.

BACHMAN: Well I don't—there's nothing much more that can be said, it's just a big scramble between competing compatibility notions and competing technical ideas, and I probably should add competing personal ideas.

RUSSELL: Was there much in the way of cultural differences? There's British, French, others from Europe.

BACHMAN: No, I don't think so. It was surprising how they were all engineers, who have a common language. I wouldn't think that language was really a serious problem we had. One interesting problem we had with respect to language was that the ISO standards require that all standards be published in both English and French, and they may be published in Russian, but that's a Russian option to create them. And yet the cooperation we had was excellent.

Herbert Zimmermann was the chairman of the French group. At Zimmerman's suggestion, the entire SC16 body decided that, if we were going to get the reference model translated quickly into French we'd take a shortcut. Whenever there was an opportunity to use a Latin-based word versus an Anglo-Saxon word with the same meaning, we would use the Latin-base word. There was almost always an easy translation into French.

I think the French version of the Reference Model came out within six months of the English version.

But there was another thing that was interesting with regard to Herbert Zimmermann. One of the ISO/TC97 rules was the requirement that at the plenary sessions we have a translator and that we have a simultaneous translation. So we had the earphones on and everything had to be translated into French for those who only understood French, and the other way around, everything said in French had to be translated into English.

---

[8] Charles W. Bachman, "Domestic and International Standards Activities for Distributed Systems," *IEEE Compcon* '78 Fall (1978): 140-143. Available from Box 20, Folder 15, Bachman Papers, Charles Babbage Institute.

The members of the French SC16-oriented standards group were required by the French national standards organization to speak in French at the plenary meetings. Typically, we had Zimmermann reporting on something. He would prepare both English and French versions of his text. He would give a copy of the English version to the translator. Then he would hold up a second copy of the English version and he would speak in French, translating the English text to French in his head. In this manner he would have made darn sure the translation was what he wanted. So I mean there was a very, very close working relationship, a very good relationship. Now John may be more sensitive to hurt feelings than I am, but I tend to be easygoing and I like everybody. I hope everybody likes me. Anyway so there are always difficulties, people have different valid positions to take and the challenge is always the need to come to a resolution that's consistent with all the rest of the resolutions. So there were not a lot of unresolved problems.

RUSSELL: If anything it sounds more like fun than trouble, because you get to go to all these nice places and, you're working awfully hard, but…

BACHMAN: You're working, you don't get much time to walk around, but the people did enjoy eating and different places you get different choices, whether it's in London or Berlin or Tokyo. We've done all of those.

RUSSELL: It seems like the travel would be taxing, both physically and mentally. There was this period early on, the period that we've been talking about, where you were constantly going to meetings, either to the ISO meetings or to the DISY meetings.

BACHMAN: Well most people were working—well I know I was working full-time on it. And a lot of people were working heavily at it. But you know we would have made progress faster if we had met collectively more often; because you had to restart every time, sometime have the same discussions over again.

RUSSELL: Looking back at this document again ["Domestic and International Standards Activities for Distributed Systems"], on the second-to-last page and the last page, you emphasized the need to move quickly. You were quite happy in your comments before about how quickly everybody moved in getting the reference model standardized or at least proposed to a standard and then moving forward from there.

BACHMAN: I think its two things. One is that you're glad you're making progress and you sure would like to do it sooner. I've mentioned over before that some people might not want to make a career of developing and defending a particular standard. And you don't want the career people and you don't want a new set of people, all of the time, because then you have to go through the education process again and again. So you're being pulled in several directions.

I eventually resigned my position as the chairman because I had changed companies, because the work I was doing for Honeywell was not really of critical interest to Cullinane Database Systems. I resigned from Honeywell because I thought that Honeywell wasn't going anywhere in the database world and was just too busy doing its own thing.

RUSSELL: So you chose to leave Honeywell.

BACHMAN: I left Honeywell and went to Cullinane and they didn't object to my continued working on the ISO/TC97/SC16 committee, but I knew it wasn't main stream for them. Typically, chairmanship appointments were limited to a maximum of five years. I bowed out at the end of four years, at the end of the SC16 Tokyo meeting in 1982.

RUSSELL: Did you take any steps to prepare your successor in any way or stay in touch with your successor and who took over from you?

BACHMAN: Dick desJardins, who was in ANSI/X3/DISY from the beginning. He was not military; he was with one of the consulting firms that were working on the Apollo mission. We'd worked closely together. As you see the name of Dick desJardins appears frequently through here. So he knew I was going to be leaving and I think it was pretty well known—everyone knew that my last meeting was going to be the Tokyo meeting and so he took over and I was busy trying to do things that were useful to Cullinet and so my attention separated from him pretty quickly. I just couldn't keep up, didn't try to keep up.

RUSSELL: Right. It seemed like the work might have been at a different phase too at that time, that once the Reference Model was set then—

BACHMAN: There were different things like the file transfer protocol, things like that were coming along. I did run into the ISO/TC97/SC16 group in Sydney, Australia, when I was down there for Bachman Information Systems business—and chatted briefly with several of the UK team members, on the Manly ferryboat.

RUSSELL: But other than that you didn't maintain any lasting relationships.

BACHMAN: I just wasn't available.

RUSSELL: Okay. What about socially? I know John [Day] said you had kept in touch with him.

BACHMAN: I hired John to come to Cullinane, because I felt he was a very good contributor and I enjoyed working with him. And when it came time, Cullinane decided that I was all very nice and all that good, but I wasn't doing what Bob Goodman, the new president wanted me to do. Cullinane was in trouble business-wise, and so we agreed, that I was going to leave. I was

concerned that John Day was left there without a sponsor. His interest really was very much communications. So he didn't stay much longer after I left.

RUSSELL: The next document I've got after this is a memo from John Aschenbrenner summarizing the second meeting. It's not the memo so much as the drawing that he's got on the next page that I wanted to ask for your comment. It's—well it looks like a margarita glass version of the seven layer model.[9]

BACHMAN: Yes. That's cute isn't it? Somehow I remember this faintly.

RUSSELL: This was Aschenbrenner's doing, not yours?

BACHMAN: Yes, his. Not mine, no. Well, you know, I guess in one sense the complexity existed at this level, the physical level, and obviously if you think of all the applications people run, you know, it's enormous. It's sort of like a flute, an endless champagne flute. I'm glad to see he gave the Presentation Layer a nice boost. But there's a lot of complexity in the Presentation Layer. The notion of three-level architecture for data independence has gone through some interesting extensions as people have attempted to build three-level systems.

After we retired to Tucson (1996), one of the people I've worked with at Honeywell, George Colliat, was working for a company (Constellar)in the Bay area. Constellar was an English company that had moved over to the United States to further the opportunity of getting venture capital. They were originally in the business of handwriting programs that translated files from one data format to another data format.

As companies merge, they end up with multiple systems that don't quite play together and they have to build a bridge between them. Constellar was quite profitable building hand-tailored programs for each data translation necessary. But they felt that their business opportunity was limited to the number of good programmers they could hire that wanted to do this kind of grunge work. So Constellar had hired, Colliat as the chief engineer of the company. He knew of the database work I had done and knew about the three schema approach to data independence.

And he asked why can't we reformat the data using the three schema approach. In this case both sides of the conceptual schema were internal schemas. Once you say I've got a conceptual schema at the center that understands the semantics of the physical data, knows what the information is, you can translate from an external format to an external format, from internal to internal, from external to internal, and what not.

---

[9] John Aschenbrenner, "Report of Second Meeting of ISO/TC97/SC16," July 18, 1979. Box 18, Folder 19, Bachman Papers, Charles Babbage Institute.

So we started out building a product, the "Constellar Hub", at Constellar. Things started out well, but we ran into a coordination problem. The two development team leaders could not agree on the data structure objects. They argued that they could work out the differences later. But they did not. Integration was not going forward. The disagreements continued until they ran out of venture capital. The fatal problem was that the two teams that were doing the development could not agree on the formats for the meta data description of the conceptual schema objects. And so that a project which had great promise, I thought, came to a halt.

One year later another company, DataMirror, a Canadian company, bought the technical information and intellectual property of Constellar. It completed the Constellar Hub product and began marketing it in Canada and some of the United States. A couple years later (July 2007) IBM bought DataMirror. Their product was called the Constellar Hub, because it was just a circle around a conceptual schema center and all these different physical schemas. IBM acquired the Constellar Hub, and to my knowledge they're not doing anything with it. But again, this business was still helping people transform from one data system to another, so that problem, data independence, is still with us.

RUSSELL: The last memo here is somewhat later, 1981, and it's from Bryan Wood, Chairman of the BSI technical committee. He's complaining about a news article that is attached on the third page of this, where you're quoted speaking somewhat unfavorably by Computer World UK.[10]

BACHMAN: They didn't even spell my name right.

RUSSELL: No, they didn't.

BACHMAN: You know reading this again, I'm sure what it says is true here. I really don't have any memory of it though. It didn't leave an everlasting memory. I know I had good personal relationships with the individuals and I'm sure I was pushing to close the deal.

RUSSELL: So this is just one of the things that come up in the process of building consensus.

BACHMAN: Well it looked like I wasn't totally successful.

RUSSELL: I thought I'd bring it up because, in part, because I'm trying to get a feel for how the different national delegations interacted, but also because you talked earlier about linguistic differences, and this seemed to speak to that point. Where in your earlier anecdote Zimmermann was going out of his way to be clear and try and keep things moving, this article struck me as an

---

[10] Bryan Wood to Charles W. Bachman, April 28, 1981, CBI 125, Box 21, Folder 6.

example where someone could use language and terminology to slow things down if they wanted to.

BACHMAN: Well, John, you have to see this quote about my complaining about the English.

DAY: Oh, yeah.

BACHMAN: Do you remember that?

DAY: Oh, yeah, very well. You don't?

BACHMAN: It faded out. It's faded.

DAY: Oh my gosh, there were knock down, drag out arguments, especially in Berlin. The British were constantly saying oh, that's tutorial, that can't go in the standard.

BACHMAN: Well I'm sure the English speak English better than we do.

DAY: Well, as I told them, we just improved on it—on their language. You know they may have invented it, but we perfected it. This doesn't have anything to do with the language; this has to do with what goes in the standard. According to them, the only thing that should go in the standard are constraints. For example, when you do a screw thread, you say it has to be of a pitch X and the spacing has to be so and so millimeters. You don't say how to build it. So, you know, the trouble was that they interpreted any way we described the procedures as implementation. So rather than saying under what conditions would this message arrive, but you also had to say what you did with it in some way, right?

RUSSELL: And they didn't want to do that?

DAY: No, they kept objecting to any language of that form. I mean this was a very different thing than what normal standards things did, and, you know, [British Standards Institute representative M.J.] Purton didn't know squat about what was going on. Furthermore we were doing something that only a handful of people in the world knew how to do, right? So you had to be telling them, you know, giving them at least some direction as to what was going on.

RUSSELL: We have spoken a bit about the end of your involvement with OSI and your transition away from OSI. One question, to track back a little bit, was if you, when you were Chairman of OSI, how much responsibility you had for creating the different Working Groups or appointing the Chairs.

BACHMAN: People basically volunteered for jobs based on their particular interests. In general, the various working groups just formed naturally. There were some places where someone was really interested in file transfer then they'd say let's have a working group on that.

RUSSELL: Right, and in some of these documents I've seen people cluster pretty naturally into those areas. And so your leadership style, then, was to let people interested in doing what they're doing go and do it.

BACHMAN: The people had come into SC16 because of their interests. And then we had the overall direction—all of it moving in parallel. Even the British, who had difficulty with the way we approached problem solving, weren't against the general solution, they were just against the natty details, I guess I didn't pay close attention to local politics. I had an architectural vision that I was trying to move with, and I probably could have been more sensitive, but wasn't.

RUSSELL: Well, one could say that that might have taken away from your effectiveness as an architect, so maybe it's better you weren't. This is a general question—what do you remember as the best part of your involvement, or are there things that you might do differently, looking back?

BACHMAN: As far as the ISO is concerned?

RUSSELL: Or any of the things that we've spoken about, but ISO in particular.

BACHMAN: I really can't think of any way I could have done it differently. Someone else might have done it differently and probably would. I had criticism from some of the people that the chairman should just be a chairman, just kind of orchestrate things but instead I was wearing an architect's hat more seriously than I was wearing the chairman's hat. To me, that was the nature of the beast. And with a lot of the systems I worked on, that proved to be a winning strategy.

RUSSELL: What do you remember most happily about this period?

BACHMAN: I guess one, the progress we made, and the people that were involved, because they were all good people, interesting people.

There is one thing I said earlier that we'd get back to and I'd like to do that now. It is about a particular contribution the Japanese delegation made. I was trying to explain to them that we should be using data structure diagrams. A data structure diagram consists of a bunch of rectangles that represent "objects of discourse." In our case, they would be the things that interfaces and protocols talk about.

And you write the name of the object type in the rectangle. So this is a "computer", there must be a "memory cell." One thing the "processors" typically have a number of memory cells, and memory cells often belong to processors in a one-to-many relationship. With this kind of diagram, you don't care how many memory cells there may be, whether there's ten or 100 or 1,000 or a billion of them. So this is the way they work. They are concerned about the types of objects that are of interest. They are equally concerned with the types of relationships that join them.

One time I was drawing a data structure diagram, while working with the Japanese delegation. One member said, "It confuses us because you have drawn lines to represent relationships, and you have drawn lines to represent objects. They're all just lines." He suggested, "Why don't you put a shadow around two sides of the rectangles and make its four lines into a single icon?" So it is an icon, not four lines. It is "reads" as a block. You can tell because it makes a shadow. And so data structure diagrams have since come out of the computer with shadows, which is simplified the diagrams. There are two tokens: blocks for "objects of discourse, and lines for relationships. Relationships are more specifically defined as "owner/member sets."

If someone were to look at the shadow-enhanced data structure diagram, they'd immediately know there are two types of tokens and they'd say this is a new algebra. These are the variables, these are the abstractions. We don't know how many instances of each type we may have. We have zero, one, or more. It could be ten or 100 or 1,000. If you look at the memory instructions, they reference the memory cells. A memory cell may be referenced by several different instruction cells.

If it's a single-address computer. The instruction cell references only one memory cell. There are two blocks, one named, ***Instruction-Cell*** and one named, ***Memory-Cell***. There is one line with an arrowhead pointing from the ***Memory-Cell*** block to the ***Instruction-Cell*** block, i.e. a ***Memory-Cell*** may be referenced by multiple ***Instruction-Cells***. If a diagram is illustrating a three-address computer, it has three of these relationship type lines with arrowheads. The three addresses provide the means by which one machine instruction may define a DIVIDE operation (X = Y / Z). So architecturally, there is a big difference between a single-address computer and a three-address computer, which you can see instantly once you look at the diagram. So this again is that part of the architectural thrust of trying to have something that many people could use and understand.

When I first I got to thinking about the word picture that I just drawn for you, I was thinking about the people at the Computer History Museum in Mountain View, California. If you just look at its computer collection, they may have 50,100, or 200 different types of computers. But there are probably only three or four major architectures, and 10 or 15 variations on these major architectures. I would like to ask, why not put a plaque in front of each computer that shows the computer's architecture, graphically.

Is one computer the same as the one behind it, except that the second one has got a new block representing an entity type on it that didn't exist on the earlier computers. Sir Maurice Wilkes of Cambridge, England invented the "B box" that we now call an ***Index-Register*** entity type. I do not know who invented the ***Base-Address-Register*** entity type. People kept adding capabilities from time to time.

The thing of value I brought to the computer is that if you want to talk about an operating system, it's got a set of objects of discourse, but they are it's own objects of discourse. And it has it's own relationships too.

These data structure diagrams are all constructed using the same two symbols: the block, representing entity types and the arrows, representing relationships (more specifically, owner/membership set types). They can begin to have a language, like an algebra, a common language, you can apply to different subjects. And if you become comfortable with the data structure diagrams, you can learn about things faster, and you can appreciate the sameness's and differences between two subjects.

There may be something that appears in a data structure diagram illustrating a software system almost actually like something that appears in somebody else's data structure diagram of a hardware system. With data structure diagrams, hardware and software issues, as implementation aspects, are factored out. They may be the same set of entity types and relationship types, or only partially overlap.

So that was the contribution the Japanese made to improve the readability of data structure diagrams. And unfortunately the chairman of the group, a man from the University of Tokyo, died suddenly during the SC16 period of activity. He was a young man, younger than I was. [Turns to John Day.] You remember him?

I have said all through my career I have been happy to have the jobs I have had, because I've had a lot of freedom to do what I wanted to do, and had the secondary advantage of what I wanted to do seemed to be useful, and therefore people kept allowing me to follow my head, and to follow the opportunities that have been offered to me.

RUSSELL: After you left ISO SC16, did you have any involvement either in standardization projects or in building information systems more generally?

BACHMAN: Well the one thing that falls into that category is what happened after I left Cullinane. I started a company named Bachman Information Systems, LLC. The intent was to build a computerized system to create and maintain data structure diagrams. Up to that point people drew them by hand, or in the case of in Cullinane, they used rubber stamps that would print a block, maybe about 2 x 3 inches in size. There were areas to record the ***record-type-name***, its ***record-size***, its ***record-type-code***, the ***estimated-quantity***, and other implementation details.

And you'd just stamp a block on the paper, wherever you wanted it. And of course one of the problems that people encountered were the changes, over time. These data structure diagrams keep getting redrawn, which is a nuisance, and being a nuisance, they'd tended not to be kept up to date. I always felt that what we needed was a software product, a CAD/CAM system,

where I could design something and I could modify it and it would redraw itself. There would be diagrams might have a hundred or two hundred entity types shown on it Change one or remove one, I'd want to redraw the whole diagram to enhance its readability.

I had the opportunity to visit Boeing, in Seattle, Washington once, during an ANSI SPARC SGDBMS meeting. Boeing had one room that was probably twenty by thirty feet, and all four walls of that room were covered with data structure diagrams. Now Boeing has a lot of products, and that was an important representation. It was clear that data structure diagram were important to Boeing and its operation. Clear because of the care and precision with which they were drawn and displayed.

We started the company, Bachman Information Systems. We started out with my wife, Connie, I, and our son Jon. The purpose was to create a software product that built, maintained and published data structure diagrams.

And there were two other small companies, one called Knowledgeware and the other Index Systems, who were doing about the same thing, data structure diagrams, data flow diagrams. Then IBM in its wisdom, decided to invest in each of the three of us. So we all became "IBM Partners." IBM invested in us and put a person on each of our board of directors. And they also wanted us to standardize our data structure diagrams and our data flow diagrams so there's only one kind, and you could store them all in the IBM repository which is on the mainframe. And so there was, in some sense, a standardization effort all under the auspices of IBM which was not terribly successful, primarily because we had more semantics built into our data structure diagram product than either Knowledgeware and Index Technology did.

In fact, when I talk about three schemas, we had a conceptual schema, and then internal schemas #1 and #2. Well, our data structure diagrams really were designed to be conceptual schema oriented, so they did not deal with the physical aspects, but rather the logical aspects of one of these one-to-one, one-to-many and many-to-many relationships, and also dealt with things like dimensions versus domains.

If you're familiar at all with relational databases, they talk about having identifiers such as ***part-numbers***, and if I have ***part-numbers*** here and ***part-numbers*** there, they would all be defined by the same domain definition. If they are defined by the same domain definition, they all have to be the same physical representation.

If we go back to our engineering training, people were trained about dimensions and dimensionalities, and the allowable operations on each dimension's values. For example, the values of two ***Dates*** dimension can be subtracted, one from another, yielding a value of a second dimension, ***Periods-Of-Time***. But they cannot be added. We had additional descriptions that I could be use to define a two or three or more domains, different

domains representing the same dimension, such as ***Dates***. I can also write an expression that specifies how I would translate the values of one domain into the appropriate values of a second domain. For example, if one domain was used for ***Eastern-Standard-Times*** values and a second domain for ***Greenwich-Mean-Times*** values, than an expression may be written that converts from one domain to values of the second domain.

The standardization of data models across Bachman, Knowledgeware, and Index Technology didn't work. It did not work, because the other two companies weren't interested in modeling at the conceptual level, or at any particular level. As Bachman people said, "Knowledgeware and Index Technology have diagrams, but they have no semantics."

IBM had a marketing theme; they talked about the "Re-Engineering Cycle." The implication was that you could 1) take an existing internal schema, 2) you could reverse engineer it to the conceptual schema level, 3) you could update it or merge it with an existing conceptual schema, 4) you could forward engineer it to the original or a new internal schema, 5) and finally, you could optimize the internal schema. We were IBM Partners implementing the three schema approach. We had the only re-engineering products among IBM's partners.

We had products that would take an existing internal schema, for various brands of relational databases, and reverse engineer them. So we had this additional power that the other people didn't have.

IBM's repository was a mistake in the first place because they had a mainframe that wasn't very efficient at interactive processing. If you could store things interactively the operating system would swap your program out more often, because it wants to move faster than the human could respond. You're dealing with human interfaces that the operating system is not designed to support.

RUSSELL: So then in terms of standards work, this was not very successful at all.

BACHMAN: It was not a success. And in fact all three companies eventually dropped out of the partnership. A couple of them went bankrupt or sold out to somebody else. We eventually dropped out. IBM stopped their work on the Repositry.

RUSSELL: Did you participate in ANSI or ISO standards activities after you left the OSI committee in 1982?

BACHMAN: No, not after 1982. I didn't. By that time I was 68 years old.

RUSSELL: After you stepped back from your active role in OSI, what has been your sense of how that field has developed? OSI didn't do as well as it was intended to do and the Internet took off all of a sudden.

BACHMAN: In some sense, the OSI movement was more oriented towards business transaction processing and moving files back and forth, but essentially different than the highly interactive use of the Internet today, where people are looking up this question, and swapping e-mails back and forth. My background came from a business operating mode, so ISO/TC/SC16 was conceived by me as fundamentally a business operating environment. It turns out that people found, well, someone a long time ago said the most important value of computers is as toys and entertainment, and I think we're largely in the entertainment business.

In fact the most powerful processors today are going into games, because there they need the highly interactive speed that business transactions don't really need, except in strange cases such as if you're trying to track and forecast missile trajectories in real-time, or something. And my impression is that business transactions aren't moving much faster than they did 20 years ago.

And it turns out that business transactions in one sense are not built much differently, because the large OLTP systems are too hard to convert to newer technologies. So for instance with the IBM mainframe architecture, which was set in the early 1960, i.e. the IBM 360s, which became 370s and 380s and so,the old application programs are still running on many IBM mainframes around the world. And the IDS database system, was reprogrammed for the IBM mainframe and is called IDMS. There is still 1,000 IBM mainframes around the world running IDMS. They are running the upgrade of a 1964 IDS product. And one reason they are still running is that no one can afford to rewrite all the existing programming. If it is not broken, don't fix it.

The biggest complaint that the IDMS users have is it was difficult hiring programmers to maintain IDMS/COBOL programs, because the programmers wanted to learn and use more modern programming languages. They don't want to use COBOL, which is a 1960 language. But many business systems have not evolved. I mean, they've reached a point of being satisfactory. One man who works for Ross Perot Industries in Dallas, Texas was saying that they twice attempted to abandon IDMS and go to IBM DB2. But, they failed both times to successfully convert their system.

So they went back to where they were, and they were still running IDMS, because the other systems couldn't meet the production requirements, because they were not as optimized as the IDMS/COBOL programs. You see what transaction systems have to do, you have to dig in someplace, access a little bit of data, update it, and probably get it out, be gone, get another transaction. The relational systems are better at queries and there is some limit to the number of queries you need to run in a business that haven't been packaged into standard transactions.

RUSSELL: So then in the way that it turned out, the Internet does only a portion of what you had in mind at the outset, which was this more comprehensive set of abilities that we've been talking about.

BACHMAN: I said you had more than 1,000 IDMS installations now running on IBM mainframes. I was told a couple years ago there's also more than 4,000 IBM IMS installations. Information Management System (IMS) was the IBM mainframe database tool in the 1970s and the 1980s. They are still running, probably for the same reasons. These big production systems are just too hard and too costly to convert and they're still working and doing the job that they planned to do.

RUSSELL: During your term at ISO until 1982 or afterwards, did you have much contact with the folks who were developing the Arpanet, such as Vint Cerf or Bob Kahn?

BACHMAN: I've met them, but I've not had any technical interaction really with them.

RUSSELL: Do you have any suggestions for other people I should talk to?

BACHMAN: Well some of the names that John suggested, Dick desJardins, or Bud Emmons, or John Aschenbrenner could be suggestions. John has stayed in that community and very close to it and so he's a live one. Hubert Zimmermann, the French SC16 delegate, is living in Paris. John Day knows how to reach him.

RUSSELL: Okay, I'll make sure to follow up with him. A final question, is there anything you had expected me to ask but I didn't, or is there anything more that you'd like to say on the subject?

BACHMAN: No, I don't think so. John has been talking to me about book that he wrote, *Patterns in Network Architecture*, and he was kind enough to give me a copy.[11] In that book he talks about where he thinks that the ISO architecture failed, in fact the general notion of Internet is at a dead end, and that they need a restart. And again, like I said, the big mainframe systems have trouble restarting. There's too much baggage, but someday it may have to restart.

I've looked at a part of his book, the parts he suggested in particular, and I said I would be glad to help him create a data structure diagram of the architecture he's really thinking about, which I think could help him communicate that architecture to other people and also force him to tighten up on the parts of his conversation I don't understand. That would help him in talking to other people and it would help him succeed in his objective.

---

[11] John Day, *Patterns in Network Architecture: A Return to Fundamentals* (Upper Saddle River, NJ: Pearson Education, 2008).

DAY: The thing that I was pointing out in the book was the cultural divide between the PTTs [monopoly Postal and Telegraph Administrations in Europe] and computer companies was insurmountable at the time, and that it was the battle over that that essentially was why OSI appeared to be so complicated. You know the phone company types were making it that way.

BACHMAN: You mean they were trying to protect their existing products.

DAY: Right, right, by trying to fight off the connection-less approach versus the connection approach. The lesson that I took from the OSI stuff was never invite the legacy in. Right? If you're trying to do something new. Now with that said, given no deregulation in Europe, Europeans were almost forced to invite the legacy in.

[Turns to Bachman.] I'll send you the data structure diagrams.

BACHMAN: You've got one?

DAY: I've got a whole plethora of them.

BACHMAN: Oh, good.

RUSSELL: Okay, well thank you very much.