Oral History of Robert L. Patrick / First Person Essay

Interviewed by:
Thomas Haigh

Recorded: February 16, 2006
Mountain View, California

CHM Reference number: X3804.2007

Table of Contents

Robert L. Patrick

Conducted by Software Business History Committee – Oral History Project

Abstract: *Robert Patrick has written significant autobiographical material and has commented on events in the evolution of the computer software and services (and hardware) industry. Some of these materials have been appended to the transcript of this interview along with references to a number of the articles and papers which he has published. Thomas Haigh has therefore focused during this oral history interview on Bob Patrick's knowledge of many of the people and organizations and his insights as to how the use of computers has developed from its beginnings with the Card Programmed Calculator through the early 1990s when he retired as an active consultant. In the interview Bob talks about his work in the US Air Force, his joining Convair and using the IBM 701, and its Speedcode software, working for General Motors and developing a data center operations management system and then going to work for C-E-I-R. He discusses being an active participant in SHARE (including its system software development efforts) and an active member of the Digital Computer Association and then becoming an active member of ACM and comments on some of the industry pioneers with whom he worked. He describes his role in the founding of Computer Sciences Corporation (CSC) and then becoming what he believes was the first independent computer systems and applications consultant. He talks of his extensive involvement with Datamation and with DPMA and the introduction of the Certificate of Data Processing. Bob then covers his long-term relationship with the RAND Corporation and his experience with the development of IMS and DL/1. Finally, he draws some conclusions about the industry's growth. The attached materials provide a wealth of detail on specific projects that Bob Patrick worked on during his lengthy career and share his thoughts on what have been some of the most important milestones in the industry.*

Thomas Haigh: This is Thomas Haigh, and I am conducting an oral history with Robert L. Patrick. The interview is taking place on February 16, 2006, at the Computer History Museum in Mountain View, California. It's part of a series undertaken by the Software Business History Committee. Bob, thank you very much for being here, and agreeing to take a part. Could you begin by saying a few words about your early life and family background?

Background and Education

Robert Patrick: Very briefly; I was raised in a middle-class household and until I got out and had to earn my own way, I didn't know I had been spoiled, because I seemed to get most of what I needed without too much pleading. I believe I was born an engineer. When I was in my teens, during World War II, my father and I hauled an old junk car home and he let me take it

apart. And I threw the parts in his pickup and he hauled it away. Even earlier, I remember my grandfather giving me a clock and a screwdriver. Of course the clock came apart rather rapidly - never to go back together again. So, my attitude and my upbringing was engineering, oriented towards doing things and accomplishing things, or understanding how somebody else had done something or accomplished something. I graduated with a bachelor's degree in mechanical engineering from the University of Nevada with a deep interest in airplanes. Right before I graduated the Air Force had come through on a college graduate recruiting trip and I was offered a civilian job at Wright Field, in Dayton, Ohio. So, when they activated my ROTC commission and took me into the service, I ended up in the same job in Dayton, Ohio, as I had planned on going to as a civilian.

Haigh: Now, I should say at this point, you've been kind enough to provide me with a number of reminiscences and anecdotes and so on. Two of those are published in IEEE Annuals of the History of Computing. I will include others as appendix material to the transcript itself, so that will save you from having to repeat stories that you've already written.

I will assume that people, who are interested in the particular topics that you've already covered elsewhere, will read those accounts. And I'm going to focus primarily on adding context, extra details and covering things that you haven't covered to the same level, in what you've already written. In these documents, you discuss your experiences. In one you tell a story that when you were in college you were not satisfied with the instruction you received from one of the professors, and with the other students, were able to have him shifted to different teaching responsibilities. So, I won't ask you to go over those specific things again. But, to ask a more general question about those experiences: How much of an influence do you think your background, as a trained engineer, and your technical interests in early life, had in the way that you approached your career later on?

Patrick: Well, I think if you sawed off my arm, you'd find an engineering logo inside. I've been an engineer all my life. I take an engineering approach to things. I look at things with an engineer's analysis. I get unhappy with people and things that are not efficient, because my engineering background points me towards better ways to do things. I got into computing - I was doing engineering - flying as a junior flight test engineer in the Air Force because we had to do data reduction on the observed data after we returned to earth. So, I was doing engineering and I used the computer as an engineering tool to produce engineering reports. And I believe this has colored my whole life. I never have been enamored with the computer as a device, as some of the college students are today. I would never sit and blog back and forth. I've always used the computer as a tool to accomplish some larger purpose.

Haigh: And would you say that that kind of engineering background was typical of many of the people that you would've worked with in computing in the 1950s and early 1960s?

Patrick: There were some periods in computing that require definition. I got into computing around 1952, the exact date I don't remember. But, from there, up to about the middle of 1955, there was a terrible shortage of computer power. There had been engineers and scientists over the years that had done in-depth mathematical analysis, but they couldn't solve the equations and they couldn't run case data through them, because they didn't have the tools to evaluate them. When the computer came on the scene, suddenly this dam of engineering analysis came open and there were jobs, a backlog of jobs. We were all working, all the time we could, to get the computer to perform these mathematical processes. So, whenever you finished one job, you jumped into the next one. And quite frequently, there were two things that were inhibiting our progress at that time. One was the fact that programming these applications was such a difficult, time-consuming job to do and to do correctly. The whole effort in machine languages, programming languages, and compilers tried to solve that problem. It was the coding problem in spades. We had work to do and we couldn't do it fast enough. The other big problem in that same period was that there wasn't enough computing power anywhere. When I met my first electronic machine, an IBM 701, Serial Number 7 - IBM only made 17 of these machines. And at that time, senior IBM management had been quoted as saying, "What are they going to do with all these machines? Why do we need to make more than 17?" They could have sold 50, if they'd made that many. There was a tremendous backlog that was unappreciated at the time. So, if you had one of these machines, the goal was to push as much work through it as you could, so that you could serve more people and do more engineering analyses. And that engendered 'Speedcode' which John Backus wrote for the 701 [While the initial development was called Speedcoding, IBM later used the term Speedcode to reference this as a program for use by customers. While it sometimes appears in the literature as Speed Code or as SpeedCode, we will use Speedcode throughout this transcript and the attachments.]; that engendered the FORTRAN's that John Backus and company did for the 704's; that engendered the operating systems to push more work through the machines.

Using the IBM Card Programmed Calculator (CPC)

Haigh: You published an anecdote describing your first experience with using a programmable machine, the IBM Card Programmed Calculator, in 1951, and then it was actually delivered, you say, in late 1952, while you were at the Air Force base. Now, the citation for that is IEEE Annals of the History of Computing, January-March, 2005, pages 78-81. So, I won't ask you to repeat the details that are in there, but I do have a couple of follow-up questions. One of them is really more on the personal side. You had the experience for the first time of being able to control the operation of the machine just by punching the appropriate holes into a card and feeding them in. Did you have a sense then that this was something that you really related to personally, that you would want to spend the rest of your life doing this kind of thing? Or, at that point did you just see it as another way of solving a problem?

Patrick: I was woefully naïve at that time, a 21, 22 year old second lieutenant, first time traveling around the country. They gave me a job to do and I did it. I didn't hypothesize on the philosophy of computing, or upon where the computing field might go. We were answers-oriented. They had a job for us to do; we figured out a way to do it; there weren't a lot of books or references around, although we got some very valuable training from the Bureau of Standards Western Computing Center on UCLA campus. Ev Yowell was in charge of it at the time.

Haigh: With the Standards Western Automatic Computer installation?

Patrick: Yes. They had built the SWAC and they had a Card Programmed Calculator and they helped us put our machine to work. The IBM machines at that time were delivered with blank plug boards and about a 50-pound sack of wires. The machine wouldn't do anything when it was delivered except generate heat when you turned the power on.

Haigh: That leads to a question I had. I know that the machine had originally been invented not by IBM itself, but by an IBM customer.

Patrick: Northrop, by name.

Haigh: And then IBM took this design; adapted it a little bit; standardized it and added it as an official offering to their product lineup. So, by the time this CPC was delivered in 1952, what kind of support did you get from IBM?

Patrick: Nothing. When writing that article, I did a literature search and I found that IBM had had a customer seminar in late 1949, to discuss programming - general purpose programming - for the Card Programmed Calculator. The seminar chairman was a fellow by the name of Cuthbert Hurd.

Haigh: Yes. And he was head of IBM's Applied Science Group.

Patrick: He had some title - they moved all the titles around at that time, so that everybody had a bigger title. But, Cuthbert chaired that seminar. And I discovered it, 50 years later, and when we went to draw it out of the IBM archives, it was like a 10-page article and they had 8-pages of it. And we searched all over the National Bureau of Standards, the Library of Congress - we still cannot find those last two pages. But, the general-purpose concepts were understood in late 1949, and were applied as routine engineering in 1952.

Haigh: Yes. Now, did you have any contact with the IBM Applied Science operation? I know they had a number of representatives. Obviously the traditional IBM representatives knew about accounting methods and punch card machines, but IBM had set up a new organization specifically to support this kind of technical work with the punch card technology. So, did you ever have any contact with anyone on the IBM side who seemed to understand the machines?

Patrick: Yes, there was a fellow by the name of Don Pendray, who later became an IBM Vice President; he was our Applied Science rep. And he came out and talked to us; didn't help us a lot, but his main job was to make sure that the machine went to work and stayed sold. So, he was the 'go to' guy, I guess, in current terminology. If we had hardware problems or if we needed extra support, or we needed manuals, he got them for us, because he was very interested in our account- this was the first machine that was on Edwards Air Force Base - the first machine being used for flight test data reduction purposes. And they saw a business there, so we got really good service out of him. And just as a slight aside there, in 1953 there was an earthquake at Tehachapi, California, which is about 35 miles away from where we had the machine installed. And the machine fell off the caster pads, punched a hole through the old flooring it was sitting on, and broke a bunch of cold solder joints. We were down a week, but they had guys out there with soldering irons diagnosing the machine; trying to find out where the bad connections were, and they got us back up and we were running again. A lot of that is due to the support we got out of the Applied Science people. Later, when I went to Convair at Fort Worth, if I can jump ahead a minute, there was a fellow by the name of Paul Knaplund, who was our Applied Science representative there and he also ended up being an IBM Vice President and he was quite a bit more help. He had more scientific education than Pendray did, was assigned to an engineering organization, and did better.

Haigh: So, with the CPC, what kind of uptime did you get out of the machine?

Patrick: We could get 8 or 10 good hours out of it a day. When we came back from UCLA, Yowell and company had helped us wire a set of plug boards for the machine. And when they wired the plug boards, they also provided us with a deck of punched cards that would test all the circuits in the machine that were exploited by those plug boards. So, when we went out to Edwards, we dropped the boards in our machine; we put the test deck in the hopper; hit the start button; and it started computing and printed out visible evidence of correct calculations! So, by the time we got through several passes of that test deck, the IBM maintenance engineers at Edwards had a list of things they had to fix. The multiply didn't work or the square root didn't reject negative numbers, that kind of stuff. So, they had work to do and when we successfully got through with the initial Yowell tests, our CPC worked. From then on, whenever we experienced any trouble or suspected any trouble, which was almost daily, we ran that same test deck to find out whether the trouble was within the IBM hardware or plug wires, or whether it was in the engineering programs or the analysis we'd done. So, we had a go, no go test very early.

Haigh: And you received those test decks from IBM, did you?

Patrick: No, all that came from Yowell; all that came from the Bureau of Standards.

Haigh: And did IBM supply you with the standard plug board wiring to turn it into a programmable computer?

Patrick: No. IBM supplied blank plug boards with the machine - two sets; a great, huge sheet of layout paper which was a one-for-one copy of the plug board wires, so that you could draw with a pencil on it and show what you wanted the wires to do, and this huge package of colored wires, with the colors depicting different wire lengths. So, that's all IBM supplied. It was a kit. It was up to you to figure out how to use it. It was a user-unfriendly machine. We carried that kit down to UCLA and we came back with a running system. So, UCLA was the keystone. IBM provided part of it, engineering was waiting for answers, and we needed that computer. And UCLA helped us get it together.

Haigh: So, then different CPC installations might've come up with their own versions of the instruction set, for example?

Patrick: Yes. The community of users was very small at that time. I made a couple of trips to the Naval Ordinance Test Station, which was in the desert straight north of Edwards. They were doing similar, but not identical data reduction and optical calculations. They did trajectory work. You have a bomb trajectory and you follow the trajectory of that bomb with a photo-theodolite - that's a camera attached to an optical telescope you can get angles off of. And then they did a reduction to plot that trajectory and to figure out what the speeds of the bombs were. And they assisted us, but they were so different than we were, their help was interesting, but not of too much value. The third organization we were associated with was the RAND Corporation in Santa Monica, California. RAND had an unlettered genius down there, by the name of Cecil Hastings. The computers at that time could not do transcendental functions; there was no way to iterate to a solution. So, Cecil Hastings did polynomial curve fits on mathematical transcendental functions. If you wanted a sine or a cosine, you could put in the value of the angle and do a bunch of multiplications, and divides, to evaluate this equation, and you'd end up with the sine or the cosine. So, for that stage of computing, that stage of engineering and development, Cecil Hastings just did an outstanding job at putting us into business. Because the IBM machines were just one generation away from being accounting machines, and they'd do addition, subtraction, multiplication and division - well, division was slow, but they did the rest of it fairly well. But, if you wanted to do any important engineering functions, you needed some help, and Hastings provided that help.

Haigh: And was there ever any kind of newsletter or formal organization formed for the CPC sites?

Patrick: I didn't even know that computing people drank for five years. We had almost no contact with anybody else. We were out there in the middle of the California desert and nobody came by to see us; nobody wrote us letters; we didn't receive any newsletters. We were just on our own, and it was kind of a fun way to start.

Working for Convair

Haigh: Then in 1953, you left the Air Force and went to work for Convair in Fort Worth?

Patrick: Yes, sir.

Haigh: And that was the end of your time commitment to the Air Force?

Patrick: Yes, that's right.

Haigh: And so, as you were nearing the end of your active duty, how did you start thinking about the future? Did you have an idea of what you wanted to do with your life at that point?

Patrick: The move to Fort Worth was a family move. My sisters lived down there and they wanted to reestablish the family. And they didn't want me to move off to Alaska, or someplace, so they enticed me to come down. And jobs, of course, were available everywhere for a young engineer with computing experience. I could've worked anywhere I wanted to. So, I moved to Convair in hopes of reestablishing that family relationship. When I got there, I discovered something that you mentioned in your thesis, Convair was split down the middle with engineering on one side of a high wall, and accounting and finance on the other side of the same high wall. The engineers, if they had anything to be done on an accounting machine, essentially passed their packages through a mail slot, and the finance people did whatever they wanted to, and mailed something back. There was very strong competition between those two portions of the organization. Engineers had total access and control of the analog computers of the day, and they did everything they could on an analog computer, sometimes too much. They sometimes should've gone digital, but they had no access to digital and they didn't understand the computer side. So, the accounting people had the CPCs and the 602As (standalone punched card calculating punches), and were essentially operating as a service bureau for the engineers. Well, when I went to Convair, I knew what was going on, on the other side of the wall. I knew how to wire plug boards; I knew how to set up machine procedures, and so on.

So, all the special stuff came to me, and I went over and negotiated with the machine accountants, and they gradually would let me get hands-on to their own, very special, private machines, which of course were the same machines I'd used three years earlier out in the desert. So, eventually, brick-by-brick, that wall was taken down, and engineering and finance had a more harmonious working relationship.

Haigh: Now, at the time you arrived at Convair, had they ordered the IBM 701?

Patrick: They had one on order, and maybe they didn't need it at that time; maybe they didn't know what they were going to do with it at that time, but they had this IBM Applied Science rep, Paul Knaplund, and it was a way to break this jurisdictional dispute because the 701 was installed in a specially constructed room in the engineering department. And we ran it ourselves, and pushed our own work through it, and the work we passed to accounting gradually dried up.

Haigh: Did the administrative people use the 701, or did they stick with the regular machines?

Patrick: We didn't let them in. Same lockout in reverse.

Haigh: So, was Convair undertaking a number of military contracts at that point?

Patrick: Yes. The big contract at that point was the B58 Hustler Bomber. The B58 was the first United States Air Force Supersonic bomber. Convair pioneered there, and I did some of that. All of the now common calculations for fuel consumption and performance and flutter and wing twist and so on. I was assigned to work with the structures engineers on the design of the wing for the B58 bomber. We used the CPC for that and in one case, I borrowed some time on the 605 calculating punch in accounting, which was the heart of the CPC. I borrowed some time on that to program a big deck. I wired a custom board that permuted addresses to do a matrix inversion. The matrix was sparse, maybe five columns wide or something like that, but it was 8000 rows long, and you couldn't hand program 8000 instructions, and get them right. But, you could use the 605 to generate those 8000 instructions. Then you slapped a front and a back on that deck, and you calculated the wing twist work for the airplane.

Using the IBM 701

Haigh: So, how long after you arrived at Convair was it until the 701 was actually installed?

Patrick: The 701 came about six months after I was there. I hadn't been there too long when I was chosen to go to New York and get some test time on 701, Serial #1, which IBM used to indoctrinate customers. It was installed at 590 Madison, right downtown. Paul Knaplund and I went to New York, that being my first trip to the 'big city'.

Haigh: So, what kind of preparations were underway to get ready for the machine? Was there already a large number of people being trained and getting ready for it?

Patrick: We had a fairly small programming group at Convair, at that time. We were attached to the engineering department as a staff function. I think there were 12 or 14 of us in the programming department. We had all kinds of backgrounds. There were mathematicians, engineers, and a housewife or two. At that time, there were no programming schools and no training courses, or anything. You'd read manuals and you'd talk to the person at the desk next to you, and you'd try something.

Haigh: Now, in administrative computing installations, it seems that even from the very beginning, there was the idea that programming, analysis, and operation would be three separate jobs. I get the impression that on the scientific side, that the roles were often much less distinct.

Patrick: True statement. On the scientific side, after the 701 was first installed, one person would sit down at his desk and review the analysis or perform the analysis. He'd write out his coding sheets; send them over to accounting for keypunch and verification. The cards would come back; he'd assemble them into a deck; he would walk into the computer room; get the plug boards (because they didn't have standard plug boards at that time); put the plug boards in the machine because the machine was sitting there idle, not doing any work - feed in his deck, and if it ran to success, it was very unusual. If he had problems, which was the typical case, he had to figure out how to dump the memory or get something to print out, so he had a clue what to fix. Then he'd go back to his desk. He might get two shots a day.

Haigh: So, the computer was treated like a piece of laboratory equipment, really? You would sign up for time on it.

Patrick: Yes, that's a very good analogy. It was like a piece of laboratory equipment. And some people were comfortable, in that environment; and some people were 'klutzes' in that environment. I remember one specific case, there was one ill-disciplined guy and somebody shouted, "The machine's yours, Ed." And Ed came running around the corner out of the programming area with a deck of punch cards in his hand, and he bumped the deck of punch cards on the door jam, as he came in. So here are 350 punch cards all over the floor and the

machine is sitting there idle. Well obviously, he missed his turn on the machine, because he had to sweep up all those punch cards and go back and reassemble his deck.

Haigh: Yes. Now, you mentioned wiring a plug board. So, with the 701, where would the plug board come in? Was it for a peripheral or a printer or a card reader, or something?

Patrick: I used a programming package, perhaps the first programming package that IBM supplied, called 'Speedcode'. It was developed by John Backus and a crew. It was almost exactly the analog of the CPC setup. It had a standard plug board on the card reader, which everybody used that used 'Speedcode'. It had a standard plug board on the printer, that everybody used, et cetera, and a standard plug board on the card punch that everybody used, et cetera. You lost some of your flexibility, but you reduced the programming preparation time a lot. So, I used 'Speedcode', and it programmed just like the CPC plugboards (the CPC setup I'd gotten from of UCLA). So, I hadn't been at Convair on the 701 more than two or three days, and I was productive. In contrast, the other people were trying to program the machine in assembly language, and sometimes they'd use symbolic and sometimes they'd use binary; they even had a binary keypunch there for keypunching loaders. All of that was just crap! If the management above us had understood the tremendous waste in what we were doing, they would've standardized earlier, but everybody did his own thing. And I was getting lots of work done using 'Speedcode' and the standard setup, and these guys were still flogging their binary decks, trying to get them to run correctly. However, when they ran, they would run more efficiently than 'Speedcode' did. 'Speedcode' was an interpretive system and was slow to produce answers, but if you only wanted 20 sets of case data, I had my 'Speedcode' jobs programmed, checked out, ran that 20 sets of case data, and it was ready for the next structures job, when these guys were still trying to get a clean assembly.

Haigh: Now as I understand it, one of the other attractions of the 'Speedcode' system, was that it provided floating point that wasn't there in the machine itself. Did you ever use those capabilities?

Patrick: Always! When I left Edwards, I stopped scaling fixed-point numbers, and when I went on the 701, the numbers were automatically scaled, and that was a big step forward.

Haigh: Now, did you ever do any assembly language programming for the 701, personally?

Patrick: I think there was one dump program I had some association with on the 701. But, generally I avoided all that binary junk. It was just too slow, and the guys at the desks around me were terribly frustrated; they drank heavily at night. It just was not the way to handle a

machine - it was unfriendly, whereas ‘Speedcode’ was friendly. And as we discussed earlier, my job was not understanding; my job was answers, so engineers could build an airplane.

Haigh: Do you remember whether there were other useful pieces of software or libraries or system tools provided with the 701 by IBM?

Patrick: None. IBM provided a library of binary/symbolic subroutines and there were several assemblers to choose from. The same transcendental functions that Cecil Hastings had converted to polynomials, IBM provided in subroutines in the original iterative form. So, if you were doing geometry, there were sine, cosine, and cotangent provided as subroutines. IBM provided a couple of different assembly programs because it was a big argument within IBM, as to which was best. Hence, they provided the customers with both. They provided a customer with a subroutine that would convert from binary and print out decimal on the line printer, so you didn’t have to do those hardware-oriented things. But, in the middle of it, where you were trying to solve an engineering problem, we didn’t get much help from IBM, or at least those that were programming in machine language didn’t get much help.

Haigh: Did the 701 come with any kind of routines to help with input and output with the tape drives and the printer?

Patrick: The tape drives were very simple to use. You told the tape subroutine the number of the drive you wanted to write on and you told it the first location you wanted to write out, and you told it the number of words you wanted to write out. The tape drives were rather simple to program, and there were statements in the ‘Speedcode’ language which allowed you to state that in shorthand. The people that were programming in binary at Convair very seldom used the tape drives because that was a strange combination of functions to them. They were mainly working in memory, (not core because they used Williams tube memories at that time). Their programs were mostly in-memory programs at that time, and they had not expanded out onto tape. Whereas the work I’d done at Edwards involved many sets of case data, so one of the first programs I wrote at Convair was to use ‘Speedcode’ and to read card decks and put them on tape. I needed tabular data. Typical tabular data of the time was the NACA (the National Advisory Committee for Aeronautics) standard atmosphere, which was a deck of about 500 or 600 punch cards. It was a big table of atmospheric conditions. The independent variable was altitude, the data were pressure and temperature and humidity for what they considered a standard day, to wash out the effects of clouds and atmospheric pressure changes. So, whenever you designed something, which had atmospheric drag, you designed it using NACA standard conditions. And whenever you flew a flight test, you did the data reduction to reduce observed flight test conditions back to those standard conditions. You made minute changes to the pressure and the velocity and the drag, so that if you had to fly it again, and the conditions

were different - the results were comparable if you reduced them to a standard day. So, that was the kind of table that was first put on magnetic tape at Convair.

Haigh: Were the tape drives reliable?

Patrick: I seem to recollect they were pretty reliable. However, we put check sums on each record because we didn't trust them.

Haigh: I've been looking at the advance materials you sent to me, and you mentioned that at some point during your time in Convair that the processes of operating the computer became more formalized and that you were involved in setting some standards for documenting how the job should be run.

Constructing a Production Monitoring System

Patrick: As I mentioned earlier, we had this tremendous backlog of programming to do. And the initial operating motif was what we called 'programmer present operation'. If an engineer had some case data to run he'd send it to his slave programmer and the programmer would get the input prepared and run it through the program that the programmer had designed and checked out and then would send the results back to the engineer. Well, when the programmer was doing the data setup and running the computer and printing it and making sure everything went well he wasn't programming. We had less than 20 programmers and we had a lot of work to do. The programmers had a strong distaste for running production jobs. They were working on the next job or maybe the next job after. They couldn't remember what they'd done in January because now this was March.

Haigh: So you had this concept of a production job?

Patrick: We had; I had run production jobs when I was at Edwards because I didn't want to work 3 shifts and I had enlisted people that would run the machine. I had made write ups that were sufficient for enlisted airmen who were familiar with the machine to run case data through a production deck.

Haigh: Would those be things like the data reduction process that you discussed?

Patrick: They were like the data reduction process. Taking the data sheets, getting the case data punched into punch cards, inserting the punch cards correctly into the deck, running the

deck through the computer, getting the printed output, taking out the previous case data, putting in the next set of case data, running it all over again.

Haigh: What proportion of the computer's time do you think would be taken up with those kinds of repetitive production jobs as opposed to one-shot formula evaluations?

Patrick: Well in CPC times it was about 50-50 development and production, but in the Convair environment when it was first installed, we had no production at all, so it was all development. Then as you started getting jobs checked out and they became available for production, the programmers would store the tested jobs in their desk drawers. When the engineer would send them a work order and give them some case data they would get the proper deck out of the desk drawer and set it up for production. It would take time away from their current programming effort to set up this deck and run it again. So when this got bad enough Convair put on a second shift for production operation and since I had production experience I started doing production write ups for production codes, for mine and others. I trained a first shift operator to run production during the first shift so the first shift was then maybe 50% checkout and 50% production and then I set up and trained the operators to do the second shift production. We tried to provide service: "If you get us the case data before 5:00 pm we will have the answer for you the next morning."

Haigh: And where did you find the operators?

Patrick: We hired experienced punch card operators because we were still handling punch card decks to feed the electronic machine. And for the first shift, they hired a lady who had variegated experience. She'd worked for Justin Boot. She was a manual accountant. She'd run a Linotype when she was a teenager. She had several skills and I taught her to be the first shift production operator and later married her. We were married 51 years last month.

Haigh: You mentioned that the programmers didn't like running the machine.

Patrick: That's true. Most programmers were not hands-on type of people. They hired lots of mathematicians and there were some psychologists as well. Some people probably had a hard time boiling water to make coffee. So when they got in the machine room and they had to put plug boards in the machine and they had to clear the memory and they had to put in the card deck and they had to press the start and they had to watch the lights and see what was going on and get all of their paraphernalia organized for the run and then to clean everything up and get out. They were uncomfortable. They were just as happy when they were designated as programmers and we went to a professional operating staff so they could program and attend meetings and talk to one another and drink coffee and smoke cigarettes.

Haigh: Which is interesting because I have definitely come across complaints from programmers saying that they, you know, hated the change. That they didn't have an involvement in running their jobs and they were at the mercy of the operators and that they'd have to wait a long time for their job to come back and that kind of thing. Do you think, at least, in this case, that would've been a minority opinion?

Patrick: I think that most of the people who said that were people who didn't know what was going on. Because if you stand back and watch, there was a sign up list where you could sign up for machine time. And you signed up for machine time in like 10-minute blocks. So if you went in and signed up for machine time at 10:00 o'clock and the first available block was at 1:35 you had a 3.5 hour wait. And what they did in those 3.5 hours was program and fool around. They couldn't concentrate and get buried in their programming task because suddenly about 30 minutes before the time that they were supposed to get on the machine they had to assemble all the stuff and get ready to go in. Well, even if they were running the machine themselves and even if everything ran perfectly they had a 2 or 3 hour turn around and they just didn't recognize it. They saw that something had been taken away from them but they didn't realize what was going on. Then we put in an operating system, which was the next stage of this professionalism. When we put in an operating system the jobs per day went from 10 minutes a job that's 6 an hour, so in an 8 hour day that's 48 or 50 jobs a day to 250 jobs a day by using professional operators and automatic sequencing between jobs. Further we got this improvement without changing the rental of the equipment, without changing the size of the payroll - we got 5 times as many jobs through the machine. Which said that the programmer got 5 times as many shots on the machine, 5 times as many opportunities to correct his errors and to produce useful information.

Haigh: What was your formal position within the installation?

Patrick: I was a programmer. A troublesome programmer.

Haigh: So you never got any kind of formal management responsibility?

Patrick: No, there was only one manager. His name was Henry Wolanski. He'd been hired first. He wasn't a very good programmer. He wasn't a very good manager. We kind of worked around him. We were one on one with the engineer who had the job and Wolanski wasn't about to get between us and the customers we were serving. So, he kind of looked after things that nobody else wanted to look after. And after I left Convair I've never heard another peep from him. I've never seen him publish anything. I don't even know if he is alive or dead. I've kept some contact with some of the people that worked down there and nobody has ever heard from since the mid 1950s.

Haigh: Do you have a sense of whether the Convair management were pleased with the overall results they were getting from the installation?

Patrick: Wolanski's boss was a fellow by the name of Johnny Goode and he didn't know anything about computers and was mainly running about 2,000 engineers and couldn't understand why we couldn't give him everything he wanted when he wanted it. So there was a heavy pressure down from him on Wolanski and Wolanski didn't know how to do anything any different so Goode did strange things. This was in the days when they first started putting professional people in cubicles so they would push the desks up as close as they could get them and put these little 2 1/2 foot walls around the cubicle. Well, when we were doing programming in those days it required quite a bit of concentration and somebody else's ringing telephone or somebody telling a joke or something else in the next cubicle floated over to yours, and broke your concentration. This is how in the later days you had FORTRAN statements with no closing parenthesis on them. Simple mistakes were made. We should've been given private offices but of course nobody gave junior engineers private offices in those days. But that would've gotten them a lot more work if they'd looked after our concentration but they didn't understand what we were doing so they didn't provide us the right facility.

Haigh: I think you implied that the management were concerned that they weren't getting the results as quickly as expected.

Patrick: The Air Force was running up and down Convair's back to get the B-58 designed and tested. And they were building it there. They had a big wooden mock up in a secret room there and they had, maybe 1,000 – 1,500 special calculations they had to do for the design, before you could cut metal on the first article. And we were right in the bottleneck. They were really pushing on us and every once in a while we would have a weekend party and we wouldn't work Saturday and they would all be mad.

Moving to General Motors

Haigh: Why did you move on from Convair to go to General Motors Research?

Patrick: I had constant abrasion going between me and Wolanski and Goode and the accounting department. Nobody wanted to listen to the problems and they just wanted more. And I had been working in military stuff then for, oh maybe 4 or 5 years, and I thought I would kind like to do some civilian work. So being an engineer with experience on 2 kinds of IBM computers I sent a resume to GM in Detroit and they said please come. So I left Convair to go and do some automotive work. I thought that'd be fun. I was car nut anyway. So I went to Detroit to do some automotive work.

Haigh: When was that exactly?

Patrick: It's got to be like 1954, late in 1954. We got married in 1955 so it had to be late 1954. I got up to GM and they were beginning to do a lot of the things that I knew or had experience in doing and for all of that I fit right in.

Haigh: What kind of state was the computer center in at General Motors when you arrived?

Patrick: General Motors had built a research and development campus out north of Detroit. It must have been 200 acres of deluxe modern buildings with lots of chrome and glass. It was a beautiful facility. They were just moving into it. And that's where I went. Now, when they started designing that facility there weren't any computers. So, there was no computer facility but they had a building that hadn't been occupied. It was scheduled to be a foundry for metal castings. They had this substantial building that hadn't been occupied for this foundry and they gave it to us. We took over half of it. One end of it was for us and the other end was for metallurgy people. So we moved into a big high bay building. We designed our own work areas with bigger cubicles. We had a computer room that had a lot of power for the casting rooms, so we didn't have to worry about power. They put in a false floor for us with soft concrete and cable trenches. It was designed with a sand floor and so we got to build an air conditioned computer room at the end of this metallurgy building. It was deluxe: tile floors, lots of glass, nice and pretty.

Haigh: Did you have a lot of visitors coming through to admire it?

Patrick: I guess IBM had guys traipsing through all the time. GMR had a CPC there and they had some design work. GMR was also doing some gas turbine development work, and the gas turbine development work and the CPC were the same thing that I was doing at Edwards and then again at Convair so I did some of that.

Haigh: What was your job title? Was it programmer?

Patrick: I was a senior programmer. And then shortly after that IBM 701 serial number 17 arrived. At that time the staff might've been 30 people or so. There was a machine room, a supervisor by the name of George Ryckman, who later became one of the presidents of SHARE. And George installed the machine and got it up and running and hired the crew and had an office on the outside but made sure the work was pushed through the machine and I was a programmer. So I programmed the CPC until it left and then I programmed the 701 in Speedcode again and that was the time that I formalized the mode of operation that I'd been

pioneering at Convair: Professional operators and batches of jobs on tape. We built what I think is the first batch operating system for any machine.

Haigh: Did it have a name?

Patrick: We called it the 'GM-NAA I/O System' standing for General Motors - North America Aviation Input-Output System. I'd designed it on paper using some of the analysis techniques that GM used for automotive production line design on how to make parallel processes fit together. Henry Lawrence Gantt was the thinker; he was industrial engineer serial number 1, circa 1910.

Haigh: The Gantt Chart.

Patrick: He made Gantt Charts and had a whole bunch of ideas on how to move things efficiently. I adapted his concepts to computing and I drew up this operating plan. (I gave the originals of these design charts to the CHM last year.) I presented that plan at SHARE #3. As I gave that presentation at SHARE a fellow came up and talked to me. His name was Owen Mock. Owen said, “We at North American have been thinking right along the same lines. Let’s do it together.” So his boss and my boss worked out the details and Owen and I did what we now call architecture. And there were programmers on the GM side and programmers on the North American side and after we each got our half of the system programmed and checked out, we exchanged our portions and we had a system.

Haigh: So that was prior to your involvement in SHARE?

Patrick: That was my first visit to SHARE.

Haigh: So you met Mock at a SHARE meeting?

Patrick: Yes.

Haigh: But that wasn’t an official SHARE project at that point. That was just a bilateral thing between the two companies?

Patrick: It was not a SHARE project. SHARE started in Los Angeles with Paul Armer and Frank Wagner and Jack Strong. And the first SHARE meeting was in Los Angeles. They number SHARE meetings and SHARE number 3 was in Boston and that’s where Owen Mock

and I met and talked technically. And so it was after SHARE 3 that Owen Mock and I put together this operating system and it became quite popular and we got maybe 4 or 5 times the amount of work done per day that we previously got done. The programmers could've been chained to their desks because they didn't have to get up and go and carry their decks to the computer room and run them or anything. They could sit over there and program or do whatever they wanted to do. It was quite an innovation at the time. Out of that came the SHARE SOS project. Owen Mock and I were both on that committee, but before the first SOS meeting was held GM gave me a high priority engineering job to do so I resigned from the committee and Owen proceeded on the SOS project.

Haigh: Now let me ask you for a clarification. It seems from here that this system that you mentioned was for the 704 rather than the 701?

Patrick: That's a true statement.

Haigh: Did they ever have a 701 at GM or did they go straight with the 704?

Patrick: No. GM had a 701 and I pulled all my operating thoughts together and was running an experimental operating system on it, what I'd call a mock up operating system on the 701, to prove that all worked. And then we designed for the 704.

Haigh: Getting ready for the 704 that hadn't been installed yet?

Patrick: Yes.

Haigh: Did the 704 arrive while you were still there or had you left by the time it went up?

More about Production Monitoring

Patrick: It arrived while I was there. We used it heavily. Since I had architected this operating system I understood it very well. Where other programmers would put one deck of cards in through the input system which was a card to tape machine, and then the tape was carried over to be run on the computer, followed by a tape being carried back to be printed, I realized that at the end-of-job card the computer had no recollection of what it had done. So in the later stages of testing a big missile program, I made up 10 copies of my missile trajectory program, loaded each of the copies with different data and ran 10 test cases on one shot on the machine. I am working in my office and someone else ran them and I got back 10 tests, let's

say 8 of them were successful and 2 of them screwed up and were dumped so I had essentially increased my throughput by a factor of 10. I was running a little programming crew at that time and that kept us just working like hell. We really shortened up the test schedule which at that time statistics said that it took a typical development about half of its life for the analysis, programming, and coding and the other half of its life was testing to get it right. Only after thorough testing could you run a few sets of case data.

Haigh: So that was literally batch processing by batching together several sets of input for copies of the same program?

Patrick: Yes.

Haigh: Because I know that often people use batch processing for a sequence of things. Not necessarily a true batch.

Patrick: A batch consisted of independent jobs that were arbitrarily gathered together as a group and put on a tape. So I had the electronic equivalent of a batch of jobs hung on the computer, the computer processes each job in the batch, sequentially at electronic speed, the batch of outputs were all printed. Again I had a batch with 10 job cards in it if there were 10 jobs in tape and then all that was carried over and mechanically printed so each of the 10 jobs could have been from a different programmer or the same programmer. It didn't make any difference.

Haigh: I am curious to know a little bit more about the system. First, when you and Mock were just working on this initially, can you remember what you would have called an operating system? Did you have that idea or was that the name?

Patrick: I think we called it a monitor. And then you had to have a more formal name before you could go in front of your colleagues; so, of course, you had to call it an operating system. But it was a monitor and what it did is that in the previous work that I had done I'd standardized enough stuff so that you could use a professional operator and keep the programmers programming. When you got into this stage of the work if you didn't standardize on the operation you had to have a genius like John von Neumann to operate the machine because every programmer did something different and every programmer wanted some different output. The console had no typewriter keyboard on it. There were only binary keys and it was an impossible machine to operate well. With no standardization you didn't know what to do next even if you had designed the machine. A Gene Amdahl type couldn't have pushed much work through it whereas when we standardized the job submittal part we could also standardize the machine room procedures and we did that. So now just a senior punch card person could run

the machine and you never lost a job. You either got back good output or you got a dump telling you why you didn't get good output. But you didn't have to go back and say "Dummy, you left my output on tape and you didn't get my answers and I have to have priority to get back in again." All that disappeared.

Haigh: So it was then called an operating system because it was automating some of the jobs that the operator would have formerly done manually?

Patrick: Yes sir.

Haigh: I am glad to hear you say that because I had an argument with another historian and I said, "That is why I thought it was called the operating system," and he didn't believe me. That's nice to hear. You mentioned that you would've originally called it a monitor. Now it was my understanding that the monitor was something that would stay resident in memory while the system processed jobs and basically monitor what was going on, do a core dump if something went wrong. Is that correct?

Patrick: That's pretty close. As I recollect, there were 2 versions. North American had a different version than GMR did. The versions were 95% or 98% identical but there were a few differences. As I recollect there were about 300 words of low core that were resident. If we had had the ability they would have been read-only and those 300 words read the next card off of tape and if it was a control card it called off of drum the operating system programs to set up the run, and set up the accounting and billing, and to erase memory or assign tapes or whatever was necessary and then it loaded the programmer's code, the application code, and transferred control to it. That part was truly a monitor in every sense of the word. If your job ran to normal completion under programmer control the monitor didn't see anything until the end card was read. However, if something happened because of bad data or a bad program or untested routines or a special case, then sometimes jobs didn't run to normal completion. The binary keys on the keyboard were set up to go dump the job. All the operator had to do was press the keyboard entry button that passed machine control to a section of this 300 word monitor and that monitor knew what to do. It knew how to dump the job and give you the output that you needed and give you the picture of memory in man readable form. So that you could go back to your desk and find out what screwed up.

Haigh: So the human operator also monitored the process? They just waited for the machine to go wrong and then they pushed the button and it jumped to the monitor?

Patrick: Right. There was a trouble entry into the monitor for the operator to use, and other than that he hung tape, generally kept track of what was going on, did the fire watch, and he

watched the lights to make sure the program wasn't lost in a loop or a repeatable pattern from which it would never get out. And mostly he monitored, guided, if you will, the running production. The machines of that ilk had a pattern to them, a sequence to them. You would read the input tape and you would do something and you would write a couple of the output tapes or sometimes the machine would hum. There were machines that you could put a radio on top of and if you wanted to demonstrate with a special program you could make it sing songs. The operator in production work got used to the beat of the machine. If he was anywhere in the room and that beat changed he immediately went to the console.

Haigh: Actually, I know another historian who has been looking at how people listened to the early machines and it seems that at least some installations had a speaker built into them so that you could hear without needing to have a separate radio. Did you ever come across that?

Patrick: Yes. Several of the machines I met had that. That tended to drive the operators crazy. I mean, these are like clicks or squawks and you do that for 2 or 3 hours and your eyeballs tend to cross. The machines at that time had high speed blowers in them to keep them cool so you had quite a bit of background noise going on. The 605 calculator on the CPC, the main frame of the 701, and the main frame of the 704 were all high heat producers so they had vent hoods on the top of them that would be appropriate if you were frying hamburgers. These vent hoods sucked all that heat out of the room. The operators at that time rotated assignments every few hours. We rotated them from the main machine room to the card-tape machine room and back and forth. If you didn't an operator would screw up. Not because you asked him to do too much but because the noise would beat him down and he couldn't perform well.

Haigh: You had mentioned that you considered the monitor the core of the operating system. Were there any other components to it like a loader or a linker or an assembler? Would you consider those as being part of the operating system?

Use of FORTRAN

Patrick: Yes. FORTRAN I was a language translator in this machine. The FORTRAN I program was loaded on the drum which was the system residence device. When the monitor was reading control cards if it called for decimal to binary conversion a program would be called to unpack the data from card fields, convert it to binary and store the numbers. If it read a control card that said compile, it would then call the FORTRAN compiler and feed the FORTRAN statements to that compiler, the FORTRAN compiler would run until it hit an end card and give control back to the monitor which then loaded the application program that had just been compiled and transfer control to it. So we had a loader, we had a compiler, and we had an assembler all available as part of the input translator. We also had a binary to decimal conversion program and a dump program as part of the output translator.

Haigh: All right. I think you say here that this system would have gone into operation in 1956 and my understanding that FORTRAN only became available in 1957.

Patrick: Something's wrong with the dates. I was working for C-E-I-R in 1958 and all this was running at GMR before I left. We might have had an early copy of that FORTRAN compiler but I agree the dates don't match up. I went to GM in 1954 and got married in January of 1955 that much I am sure about. I went to work for C-E-I-R in early 1958 as I recollect because I had a 1958 Cadillac. That is how you remember things when you work for a car company. I was a bonus employee of General Motors and you could buy one Cadillac a year at discount and my wife thought that was great.

Haigh: The first FORTRAN manual was dated October 1956 but I think I do remember reading that the code actually shipped sometime later. The manual was finalized but the system wasn't actually released until later. I also actually know one of the problems with the SHARE Operating System project was that it was finished before transitioning to FORTRAN and the SOS system as produced was really designed to work with assembler so that limited its usefulness and practice.

Patrick: Yes. I think there was a change that they put in to make it do the same things that we had at the GM system.

Haigh: So you're saying as well as the monitor itself, the monitor would be able to call on these other parts of the system and when you said operating system would you mean all those different pieces?

Patrick: Everything that was on the drum it was monitoring. It was a full set of software.

Haigh: So your 704 was configured with a drum that these routines could stay resident on?

Patrick: The memory was quite small. I want to say originally 8192 thirty-six-bit words. We did a lot of work with very small memories in those days.

Haigh: You mean the main memory?

Patrick: Yes, the main memory, the electronic memory.

SHARE and the SHARE Operating System

Haigh: Now the SHARE Committee, I actually have here from SHARE, it's the report from the SHARE 709 System Committee dated April 9, 1957. That just has a list of the people who were originally on it. So it says that on January 7, 1957 a request approved by the SHARE Executive Board was sent to each SHARE installation for volunteers for this committee. Your name is there as one of the people on the committee and I think Owen Mock as well.

Patrick: Greenwald was from RAND, Tom Steele was from SDC. I knew Maureen but I don't remember where she was from. Ira Bolt I think was IBM at the time. Harvey Bratman was SDC, I think. Elaine Boehm was IBM I think. Charlie Swift was SDC. Roy Nutt at that time was with United Aircraft.

Haigh: Do you remember any of those people being particularly active in this area? I know you left the committee before anything actually happened but do you know if any of those people would have been the real driving forces in this area? Do you have a sense of who they were?

Patrick: Well, Mock was quiet and smart. Nutt was kind of introspective so he would not have been a leader. Don Shell was a leader out of GE. He had done a competitive system that hadn't worked out but he really wanted to do another system. Tom Steele could have been the leader of that but I believe the leaders were IBM people.

Haigh: So Elaine Boehm and Frank Beckman were the IBM members.

Patrick: Elaine Boehm was outstanding. She was one of the early liberated ladies; she didn't offend; she didn't go out of her way one way or the other. She was just smart.

Haigh: I have one more question about the system that you did produce with Owen Mock. You mentioned that the monitor was integrated with the other tools to assemble the code, link it, load it, those kinds of things. Did you work with IBM-provided routines or was it all written from scratch?

Patrick: We got a FORTRAN from IBM, and the assembler from United Aircraft, and while we may have adapted some, much of the rest was all written from scratch. GM did the output end of it and North American did the input translator.

Haigh: You weren't using any IBM-supplied software with the 704 as far as you can remember.

Patrick: Well, since I wasn't an implementer of the system, if you will pardon the term, I was the architect of this system and then I went off to do an engineering job and the team put it together so I am not really the right person to ask. As I recollect, Owen had the input translators either under his thumb or pretty close and the GM guys, a fellow by the name of Jim Fishman and another fellow by the name of Don Harrouf that were on the GM side and they just knew a lot about translating binary into formatted decimal for print.

Haigh: Just before we move on to the next section, I'd like to ask for your general impressions of SHARE. You had mentioned that you met Mock at the SHARE meeting, and you've also told the story of how you were originally part of the committee that produced the SOS system, but then you had to withdraw from it before you really did anything. So do you have anything else to say about your impressions of SHARE, the role it had played, any of the people who were involved?

Patrick: I think SHARE was a rather innovative management concept. The guys from Los Angeles had produced a language translator for the 701 called PACT. And L.A. was a hot bed of computing. There were about six or seven big aerospace companies all within the confines of Greater L.A. and if you were going to work in aerospace, you probably had to work sequentially for more than one of these companies, because as the contracts moved, people got laid off and got re-hired, so you had friends in other companies. So when PACT was put together, the North American guys and the Douglas guys and the RAND guys all knew one another. They probably drank beer once a week. So when they decided, that we're all doing the same thing and it's all very slow because we don't have a language translator, they decided-- I mean, they didn't use those words - but they decided to build the language translator, and PACT was the result. PACT was such a success that other people wanted to get in on the game. And these were big companies that were not in the Los Angeles area, so it was a natural development out of the successful PACT project for these guys to say, "Well, all right, we'll work the bigger circle, but we have to have a Secretary or somebody to call meetings and get the doughnuts sent in on the meeting dates and so on." So it was just kind of a natural evolution of middle management there, and they went to do their management things, which the first priority was to lean on IBM to do better. And then gradually technicians like Mock and myself were invited to go along, and they had some technical sessions and some management sessions, and IBM closeted them up and told them about unannounced equipment or told them about deliveries and so on. So it was kind of a natural evolution of technical people working on a common problem.

Haigh: Yes, so I think the broad history of that is fairly well known. I was just wondering what it was like subjectively at the meeting. Do you have an impression of going to those meetings; was everything very serious, were people laughing, were they enjoying themselves?

Patrick: Well, there were serious sessions. We worked on serious stuff. But it didn't take very many SHARE meetings for the management to decide that we really ought to have an open-bar mixer. So when you registered for SHARE, you paid 10 dollars, disguised as a registration fee. It was called SCIDS [often given as SHARE Committee for Imbibers, Drinkers and Sots, though other variants such as the more respectable Sessions on Common Information Discussion are given], and it was disguised on our expense accounts as a SHARE ticket or access, or something that would pass through the accountants. SCIDS was the place where everybody really got to meet one another and talk about the informal part of the program, and some of us slept late and worked the technical sessions in the afternoon and went to SCIDS and didn't pay any attention to the morning sessions. The morning sessions were all attended by our bosses, and we didn't pay any attention to them, either.

Haigh: So was that then when you first started really thinking that you were part of this larger community?

Patrick: Yes, I believe that was. The community was growing from the original seventeen 701 users who had not met, maybe the half a dozen in L.A. knew one another. When I was at Convair, we didn't know anything about PACT; we were 1500 miles away and completely oblivious to this good work that was going on out there. But when we went to migrate from the 701 to the 704, we discovered that there were a lot of people who had the same needs, and if you could get somebody to do a cotangent function and you had the tangent function, you got twice as many useful subroutines just by doing it to common standards and exchanging.

Haigh: So did you ever use any routines from the SHARE library yourself?

Patrick: After I left the SOS project, I did a big missile job. Four of us worked full time locked up in a secret basement room with a guard outside the door doing a big missile job. And we regularly were programming in machine language using the SAP assembler, which was a SHARE program that Roy Nutt had designed. We used all the transcendental functions because we were doing ballistics and geometry and that kind of stuff. We used the subroutines that had been provided in the GM library. They didn't come with SHARE's name on them, but I suspect that was their source. All of it had been integrated into the operating system and we used it. And IBM regularly visited us to pick our brains, because it didn't take long for us to have more experience, good practical experience, in the use of IBM equipment than IBM had internally. So IBM came out, and they were friends. In contrast to the isolation we had in the old CPC days, we now saw IBMers regularly, and we got publications, and we got newsletters,

and we were invited to submit stuff, and the salesmen bought a drink every now and then, and so on.

Association for Computing Machinery (ACM)

Haigh: All right. And while we're on this topic of involvement in organizations, I would like to ask you about the ACM. Do you remember when you first became aware that the ACM existed and when you would have first joined it?

Patrick: As I recollect, I was still working at Convair and inveigled a trip to MIT. And ACM was just getting started then, and was essentially owned and managed by the academics that had time to write. And part of their job was to publish or perish, and we industrial people benefited from that. But if we'd stopped to publish, we would have perished, because our managers were really beating us up to get jobs done. So we were outsiders to begin with, and I felt like I was an outsider all my life as far as ACM was concerned, and I was a member for 25 years or something like that.

Haigh: So you think that would have been back when you were at Convair in about 1953?

Patrick: Yes, and then they published these little undersized journals, and I had a whole shelf of those, and we read them all to see what was in those journals that was pertinent to our jobs. Integration techniques and practical differential equations were learned that way. We didn't learn those out of textbooks because we were engineers. We learned them via the ACM route. So it helped, but we were not full members, we were peripheral members.

Haigh: All right. So you held your membership, you read the journal, but you weren't more actively involved. And did that change then?

Patrick: They didn't even have SCIDS.

Haigh: No, I think they might have looked down on it. I talked to one academic, and they were saying they'd been to a DPMA meeting once, and they were shocked that it all just seemed to be a drinking society and no serious discussion of intellectual matters.

Patrick: I actually met Norbert Weiner once, because he was in on the early formations of ACM, but we couldn't understand the title: "The Association for Computing Machinery?" We weren't building any machinery; we were building programs, application programs, and we only

built systems software when we needed it. I mean if someone had given us all the software we needed, we wouldn't have done anything but applications.

Haigh: And did that change at any point later on when you were involved, for example, with the Los Angeles chapter or with the data processing special interest group that was set up in the 1960s?

Patrick: Well, the collection of aerospace industries in Los Angeles had started an informal monthly meeting called the Digital Computer Association. We'd meet once a month, we'd have drinks, we'd have supper, and we'd have a technical meeting. And it was-- in the later days it turned into a marching and chowder society - but in the earlier days it had a serious technical component to it. And it was the predecessor of the ACM in the Los Angeles area. It preceded the ACM and overlapped the ACM, but the ACM with its national charter lasted longer. The DCA people just turned into a marching and chowder society and had a great time, and we eventually dispensed with the technical programs altogether. Once, Lou Gatt was reporting some early FORTRAN experience he got at Los Alamos and he fell off the stage, and we all laughed about that and had to go get him. He was stuck between the stage and the wall, had to have help to get out. The late Bob Bemer once attended a meeting, and he brought a goatskin full of red wine and sat at the meeting drinking wine out of his goatskin. The DCA wasn't too formal.

Haigh: And later on, were you personally involved in attending meetings of the L.A. ACM chapter?

Patrick: Yes, as DCA went down and ACM ascended, I was a member of the ACM and was getting all their meeting notices, and I would attend meetings if the subjects were interesting to me. As an independent consultant I had to keep my name and my face in front of the community. I had to buy a few drinks; I had to talk to a few people to keep my business flowing, so as DCA dropped off and ACM ascended, I started attending ACM meetings more regularly.

Haigh: But you never held elective office or anything like that?

Patrick: No.

Haigh: And whom do you remember as being the main driving forces in the L.A. chapter?

Patrick: Can't help you there. I don't remember a single name.

Haigh: But do you have the impression that the character of that chapter was less academic than the national ACM?

Patrick: Yes, yes. It was about halfway between the national and the nuts and bolts people. Most of their talks were useful stuff to the L.A. community, and about half of them were useful or interesting to me. And the numbers would be 10 percent interesting and I'd skip 90 percent of the rest.

Haigh: And what time period are we talking about here? Would that be the 1960s or the late 1950s?

Patrick: I went to the last DCA meeting in the 1980s or mid 1980s, when DCA finally stopped having meetings. ACM and DCA were competitors through the middle to late 1980s. DCA would have an annual bash, and somewhere through there I was elected chairman. And you're not allowed to turn down the election, which meant that I had to arrange for the annual bash. I chartered a stern-wheeled steamer in L.A. harbor, and we put all these drunks on board, and were portable.

Haigh: I had no idea that the DCA went on that long. I only had heard of it from the PACT era, very early on.

Patrick: I'm trying to remember. We moved out to the ranch in 1980, and I drove back for the final DCA meeting to see all those old people and get drunk one more time, and I stayed over that night.

Haigh: Let’s shift back to your career, then, at this point. When and how did you come to move on from General Motors?

Patrick: Well, after I'd done automotive work for General Motors and missile work for General Motors, I got interested in the production of automobiles, figuring that I'd have a better career if I went into their mainstream. I was interested in curve fitting and die sinking, because that's the way you make an automobile. The stylists make a full-size clay model of an automobile, and then you have to translate that clay model into engineering drawings, which are done on walls about the size of the one in this room, drawn one-on-one full-scale on aluminum plates, and then you have to design the dies and create male-female die pairs before you can put on blanks and press out a fender, as an example. And that was the bottleneck-- that sequence was the bottleneck in putting out a new car. You could get engines and drive trains and springs and stuff a lot quicker than body parts. Exterior body parts took many months, like 24 and 28 months to produce in quantity. And I had charts of that process. And I was working on the

bottlenecks in that sequence of production steps and got into a jurisdictional dispute where the vice president of the General Motors process development staff called my research vice president and said, "You're treading on my toes." And they didn't offer to transfer me from research to process development, and my boss told me to stop, and I thought that was untoward. And I'd been there five years or something like that.

Moving on to C-E-I-R

Haigh: All right, so what year was that when you left?

Patrick: That had to be like 1957, and I moved to C-E-I-R in late 1957.

Haigh: All right, so can you describe what C-E-I-R was like at that point?

Patrick: C-E-I-R was in Arlington, Virginia, not too far from the Pentagon, about a mile, in one of those industrial parks.

Haigh: So they only had the one office then?

Patrick: That's right. One office in an industrial building, and they had modified the second floor and installed a couple of computers up there, IBM 709s as I recollect.

Haigh: Let me see, they got the IBM 650 in 1956, and they got the IBM 704 in late 1957. And then they got two 709s a little bit after that. So it sounds like they were relatively slow moving into computers, and then they ramped up very quickly in the late 1950s.

Patrick: That's an exactly correct explanation. They were sitting next to a gold mine in the Pentagon. The Pentagon was even slower to get into computing than C-E-I-R was, and so here is C-E-I-R with a couple of big machines, and the Pentagon had computing to do and no computers to run it all. So they got almost sole source contracts, or guaranteed contracts. The salesman walked over and filled out the paperwork for them. And we did analysis and programming and production, and we had a secure classified facility. And that was when I moved up from supervising a small group of four, five, six people. When I moved to C-E-I-R, I moved up to middle management. I had 200 people working for me. And that was a big step in maturity as far as I was concerned. It broke me out of the pure technician role and into the managerial role where I worried about budgets and people and hiring and firing and billing and all that stuff.

Haigh: So how did that happen that you went from General Motors? You've said why you left General Motors, but why did you go to C-E-I-R? How did that fit in with your plans?

Patrick: I had an opportunity to work for a small service bureau in Detroit. Tesaro, as I recollect. He had an IBM 650 shop and was doing payroll, accounting and stuff, and that seemed like small potatoes for me, and my IBM contacts told me that there was a big shop in Virginia. And I had the military clearances that allowed me to go to work almost the day I arrived, whereas if they'd hired somebody out of academia, he would have hung around for three to six months before they could put him to work, so I went right to work, because I had transferred all my clearances.

Haigh: And what was your job title?

Patrick: I was the assistant director of computer services, and my boss was a guy by the name of William Orchard-Hays. And Bill was mesmerized by linear programming, and he would go into his office and close the door and invert matrices and figure out how to optimize things. He did some work for a big feed miller in Minneapolis to determine how much wheat and barley to mix into the feed so that they got optimum protein into the cattle. Very good work, but it had damn little to do with running that 200-man shop.

Haigh: But he was the director?

Patrick: He was the director and I was the assistant director, or associate director I think I was called. And I had experience on the two big machines, and I brought in a machine room supervisor from General Motors who came in with me. And I had all these programmers, and I knew what they were doing and how to do it, and could have done it all myself if I'd stretched out the time far enough.

Haigh: And what kind of jobs was C-E-I-R taking on at that point?

Patrick: Well, a lot of it was military jobs, whatever the Pentagon needed to be studied or calculated, or inventoried; you know bread and butter kind of stuff. We did some of all that work. We had a contract that took me over to the Library of Congress for my first working assignment over there. We had a contract from Virginia Power. They had put big transformers on poles that had more KVA capacity than was needed, and a big transformer hanging on a pole, lightly loaded, was expensive. And as their service area grew, they needed big transformers elsewhere. So they could buy little ones and swap out the big ones to use the big ones where they needed them. So we inventoried all those transformers and figured out the

loads for them, and saw what the load was and said go, take this one down and put on a 9KVA over there and you've got a 100 KVA for inventory.

Haigh: And how long did you stay at C-E-I-R?

Patrick: About 16 or 18 months.

Haigh: So what year would you have left them?

Patrick: Well, I left C-E-I-R to go to Computer Sciences, and Computer Sciences was formed in April of 1959. I've just researched that and dug up the original articles of incorporation of Computer Sciences.

Haigh: Right. So then that would be early 1959, then?

Patrick: April, to be correct.

Haigh: So that fits, C-E-I-R only started nationwide expansion in 1960, so through that time they would just have had operations in the Washington, D.C. area.

Patrick: I worked in Washington, Arlington to be exact, until I went to work for Computer Sciences.

Haigh: Right. That's what I'm saying. In 1960, C-E-I-R started expanding very rapidly, so they began operations in different states.

Patrick: Yes, and I was not associated with them at that time.

Haigh: And do you think the company was well run?

Patrick: Well, there were a lot of companies, and we heard quite a bit about them in our Professional Services meeting yesterday. We heard a lot of words that there were professional services companies where there was a dichotomy between the people who controlled the company and the people that were doing the work, that sometimes the people who controlled the company, while they may have been smart and had good background, had no appreciation of what was going on down on the lower floors. And I thought C-E-I-R was one of those.

Unfortunately, I worked for several of those companies where the workers were doing a good job, programming and figuring out what needed to be done, and running the computers and stuff, but the management had no concept of what was going on, and C-E-I-R was one of those.

Haigh: And what were your impressions of Herbert W. Robinson?

Patrick: Well, Herbie was kind of out of it. He was a Ph.D. in mathematics or something, economics?

Haigh: Let me see if they've got that information. I believe he had operations research or modeling, some kind of background like that.

Patrick: He never once came into my office to talk to me. And I was running the big money part of the operation. He never came to me and talked about my programmers and my machines; mine, because Orchard-Hays was locked up in his office doing some modeling. But Herbert Robinson never once came down to see me, which is a kind of a strong disconnect.

Haigh: Now, with C-E-I-R for the first time you were moving from computer installations within a user organization into a firm that was providing services to external customers.

Patrick: At that time it was the biggest service bureau in the Country, and that made it the biggest service bureau in the World.

Haigh: Now, was that something you did on purpose because you liked the idea of being in the consulting services field, or was it just an accident?

Patrick: I did it on purpose. I recognized that working with big machines paid more than working with little machines. I mean, you could be in computing on the small machines, and go home, lock your office every night at 5:00 PM and go play golf on the weekend and have a moderate income, or you could go up there and ride that wave and have a lot of fun and earn top dollar. So I went to C-E-I-R because it was the biggest shop.

Starting Computer Sciences Corporation

Haigh: And why did you leave it?

Patrick: We thought there was a better opportunity in contract programming. And Fletcher Jones and Roy Nutt and I had discussed this several times at SHARE meetings, and when it looked like it was the time to move, we moved. And Roy was working for United Aircraft, and Fletcher was working for North American, and I was working for C-E-I-R.

Haigh: And you tell the story in here about how you and Roy discovered that Fletcher Jones was not devoting the time to managing the startup he should?

Patrick: Yes, there was some unusual accounting taking place there. And Fletcher was spending a lot of time in Hollywood, and Roy and I were down there trying to get this compiler out, and it was a very, very strange time. The client came out from Boston, and we thought the client was going to run roughshod over us and make us work real hard, and he spent his week at the swimming pool at the Hyatt, and we didn't understand that either, because we were working like hell trying to save his ass.

Haigh: And the client in this case, was that Dick Clippinger?

Patrick: Yes. So I didn't understand why he wasn't more interested in getting down into the nuts and bolts where we were working.

Haigh: So what was Clippinger like? He seems to pop up in various places in these early computer days.

Patrick: Well, George Trimble of CUC said at breakfast yesterday that he had met Clippinger the first time when he was working at Aberdeen. Clip was a Ph.D. I believe he had a mathematical background. He was the technical staff to Walter Finkey, who was the president of the Datamatic Division of Minneapolis Honeywell Corporation. So Clippinger was the one to either discuss and/or approve all the major technical decisions that Honeywell made. And at that time a lot of big companies, General Electric, National Cash Register, and Honeywell, all thought that computing was the big thing, and they were going to set up computing divisions and get in on this next big thing. Well, as we have seen, they lasted until the next economic downturn, and only Snow White was left and all the dwarfs died.

Haigh: And I think you said Fletcher Jones had been responsible for the SURGE report generator. That was a SHARE project?

Patrick: Yes, that was. Fletcher Jones at one time was SHARE's Secretary, which gave him access to the SHARE files. He maintained them and kept them. It gave him as many contacts in the SHARE organization as the IBM Corporation had. He knew, or had, every name, address and phone number of every manager of every big scientific computer shop. Outstanding-- if you give a salesman a Rolodex preloaded with that stuff, he's just got to be a hero.

Haigh: So that was essentially part of the business plan of CSC, to exploit those contacts?

Patrick: If you could call it a business plan. Fletch and Roy and I were unbelievably naive at what we were doing. We thought that Honeywell needed this compiler and was willing to pay for it, and we were going to work like hell to establish our reputation and then build from there, but it didn't work out quite that way. I left after six months and a big argument with Fletcher. Roy and I argued with Fletcher, and then Roy decided to go back. Fletch and Roy kept the company. The project we were working on didn't turn out well. Honeywell left that business later, but Computer Sciences survived and is now very successful.

Haigh: Didn't Honeywell stay in the computing business into the 1970s and buy out General Electric?

Patrick: Well, Honeywell had a 1401 program called the Liberator that would translate 1401 programs to the Honeywell 200, as I recollect, and that had to be a mid-1960s program. That's when they were liberating 1401 users. And they built a couple more machines there, but if you and I could secretly access the Honeywell books of account, I suspect it was a loss leader all the way. Other divisions of Honeywell were very healthy, and I later worked for some of those.

Haigh: So you were working on this FACT compiler that was going to be for business applications. Now had you personally had any experience with compiler design before?

Patrick: No. I made two contributions, I think, to the art while I worked for Computer Sciences, although I didn't work there very long. When we staffed up at Computer Sciences, most of the people that Fletcher hired were North American engineering programmers. They all spoke IBM. Honeywell, at that time, thought that their business strategy was that once they captured a client, they could hold him for life, and they did a lot of things that would look to you and me now as strange as part of this effort to keep the club together. And Honeywell terminology was just unbelievably confusing to us IBM people. As an example: Engineering wise, you're writing on magnetic tape, and every time you want to write a bit, you flip it 1 to 0 or

0 to 1. When you get to the end of that record, you re-set everything and you get a pattern. That pattern mathematically is a hash total of what has come up before, computed on the power of two. In the IBM world, we called that a check sum. Honeywell called it an orthocount. Now that made our heads ache. There were other similar phrases. So I wrote like a 20 or 30-page manual in a very short period of time, and it was the introduction to the Honeywell H800 for 7090 programmers. I wrote about Honeywell H800 features using the terminology that the whole crew was familiar with, and we just jump-started that project by about a month. So after the project got going and we'd split it up and had guys working on the language translators input and so on and so on, I did a literature search. Clippinger had this idea that you ought to program for a computer in natural language, and that's where FACT came from, Fully Automatic Compiling Technique. Well, I didn't know very much about natural language, so it seemed like the right thing to do was to go find out about natural language. And in doing a literature search, and I reviewed this information not too long ago, I found that there was a big debate going on in the late 1930s about the language for commercial enterprises and overseas contracts. And the debate was between Esperanto and Basic English. And the linguists had worked a long time in the 1930s trying to find out what was the minimum vocabulary you needed to conduct commercial transactions.

Haigh: And you have that story in the notes?

Patrick: Yes. And as the story says, I discovered the work that led to Basic English ("The System of Basic English", C.K. Ogden, 1934, Harcourt, Brace); I wrote examples, and CSC adopted Basic English as the backbone of FACT. That found its way to the CODASYL committee via Roy Nutt, and into COBOL, and here we are 50 years later. We owe a debt to those linguists of the 1930s.

Haigh: And so you have this anecdote which is appearing in the January-March 2006 IEEE Annals of the History of Computing, "Getting Started in Consulting."

Patrick: That's the one that's just coming out.

Haigh: And that I think describes pretty well your entry into the consulting business, so that's what happened next. Now I guess I would just have one question: when you entered the consulting business, essentially for the third time, because first you had been doing services with C-E-I-R, then you had been part of Computer Sciences Corporation, and now you were entering the business again, this time on your own.

Deciding to Start a Consulting Practice

Patrick: Really, there was a fourth time. In the late days of the Cold War, General Motors recognized that there were some skills that were in short supply throughout the Corporation, and they formed an internal consulting stable. And they had metallurgists, Ph.D. level mathematicians, maybe a dozen of us. And I was the computer guy in the General Motors consulting stable, and they loaned us to all the divisions throughout the Corporation.

Haigh: So in that sense, then, you'd been already in similar parallel businesses before and now you were doing it as an independent consultant?

Patrick: Yes, sir.

Haigh: Now, at that point, did you have the idea that independence was something you wanted to stick with, or did you think this just might be just another short phase before you would either grow the business or leave it?

Patrick: Well, I sat down and did a simple calculation as to how many billable hours I could have if I was a solo practitioner and how many billable hours I could have if I was also running a crew and raking 20 percent off of every billing like in the body shop businesses we were talking about yesterday. So I ran a simple calculation and figured out how big the crew had to be to give me the same take home pay as I could make by just being an individual consultant. Turns out the number was like 18 or 19, so I would have had to have a facility, secretaries, machine equipment, and about 20 employees, and salesmen, to breakeven running a body shop.

DPMA and the Certificate of Data Processing Program

Haigh: That brings us up to your entry into the field of independent consulting and I'll be asking you a bit later about a few of the jobs that you've worked on. But I'd like now to return again to your involvement with other organizations in the broader computing community. One of the things was that you were involved with the Certificate in Data Processing Program, which was run by the Data Processing Management Association. I think from 1963-1964, you were an initial member of the Certificate Advisory Council that the DPMA set up to advise it on its attempt to basically certify junior data processing managers. Can you talk a little bit about how you became aware of that program and what led you to become involved with it?

Patrick: Well, as an independent consultant, I was constantly looking for opportunities to get my name professionally in front of potential clients. As I mentioned to you at lunch, I had been

invited to be part of a stable of writers for Datamation Magazine and I wrote maybe three or four articles a year and fielded 20 telephone calls a year from the professional staff. Well when the Datamation editor got hurt, I became temporary editor of the magazine.

Haigh: What year was that?

Patrick: That's got to be about 1963, and during that period, DPMA came out with this test. And it was exactly as you described. It was a way to get the DPMA community, which was mostly punch card supervisors, to pull themselves up by their bootstraps and become more professional. Well, without me, Datamation was all arty and journalistic people. When I was editor of the magazine for a while, I injected a technical flavor into it of course because that was my natural attitude. So when DPMA came out with a test, we thought it would be informative for me to take the test and report on it. So I did. I went and sat for the test as a regular supplicant and it turned out it was a pretty good test. I had done a lot of things, but if I had been just a scientific programmer, I would have had a hard time passing it. So it was a pretty good test. When I got through with all that, I went back to the magazine and I wrote an article that complimented them on the test. ("The Maturing Field", Datamation, January 1963). Well, if you flip this same coin over, DPMA was looking for acceptance and what better acceptance could they get than an editor of the top magazine of the field who had written a complimentary article. So they interviewed me, and I was invited to join the Certificate Council. We controlled the content of the test, and they had an organization that administered the test and distributed them and graded them and all that stuff. I didn't have anything to do with that organization. So I was a content guy.

Haigh: What was your impression of DPMA as a whole?

Patrick: Well, I might have answered this question differently if I hadn't read your dissertation.

Haigh: Oh, dear. I hope I haven't prejudiced your mind.

Patrick: No. I think you had it right. We in the scientific community had parochial viewpoints that were self-serving. We thought that the ACM was academically oriented and something to be tolerated. We thought the DPMA were people that, at one time, were on the other side of the block wall we had at Convair. These were the old punch card people, and they clearly didn't understand what we engineering geniuses were doing. Well, as it turns out, when you got a little more sympathetic and you got to know them better, they were working on pretty tough problems too. They were just different problems than we engineers were working on. They involved a lot more data, and a lot fewer equations. But they had a long way to go and there weren't very many college degrees among those people. A college degree was kind of precious

in the punch card world, and the people that came up from that community, even those in big punch card shops, wouldn't claim they knew what was known as the scientific method and they didn't do experiments and they didn't write very well and so on. So the certificate program was a real top-notch effort at solving a tough professional problem. I was kind of glad to be a member of it, to be part of it.

Haigh: As you were on the committee and had a chance to meet some of the DPMA people who were involved in it, I wonder if you could give your impressions of them, perhaps starting with Cal Elliott, who was the executive director.

Patrick: Well Cal was a good administrator of a stable professional society. He kept everything running smoothly and didn't seem to be very creative, but obviously he'd been there when the test was created and he put all that together. So he was better than his first impression gave you. First impression was that he was a solid, stolid person, but he was capable of solving problems within his sphere. Then Cal was followed by a fellow by the name of Jim Adams and Jim Adams at one time worked in scientific areas. He might have worked for North American. Jim Adams was a hail-fellow, well met. Jim would laugh and drink and was easy to get along with.

Haigh: I think Elliott remained executive director, but Adams I believe held some other posts within the organization. Elliott stayed as executive director I think pretty much through the whole of the 1960s.

Patrick: Okay. Well I didn't see very much of Elliott after the initial blush.

Haigh: Right. I guess he probably just introduced you to the other people and then left you to the Council.

Patrick: The guy who was the serious thinker, and he was approaching things in his way, was John Swearingen. He was at one time president of their association, and I don't know for how many terms, but he was seriously concerned about raising up the DPMA membership by its bootstraps. He was working on a good problem, on a real live problem. He had the membership and the society at heart and he wasn't abusing any of that. I liked John. I wouldn't have put him in an engineering assignment, but that wasn't the assignment he had. He had a good solid executive management approach. He, at one time, in the latter stages of our friendship, went to work for a U.S. Senator and moved to Washington, D.C. He was on the staff of a senator whose name momentarily escapes me and I met him a couple of times on my Washington trips.

Haigh: And I think he was responsible for some aspect of automation for the Senate.

Patrick: The Senate had a voting system that they put in, I believe it was the chair-side voting system, and I believe John was instrumental in that or at least in the middle of it.

Haigh: As I think of all the senior DPMA people, he was the one who was most in favor of making ties with other computing associations and broadening the association a little bit to communicate with the outside world.

Patrick: Yes. He had a good vision.

Haigh: Do you have any memory of any contributions that the Certificate Advisory Committee made to the design or operation of the program?

Patrick: No, I'm afraid I do not. No. All those details have escaped me. Sometime later in life, I became a member of the IEEE. So in my career, I've been a member of all three professional societies. The IEEE Society was as different from the other two as the other two were different from each other. IEEE was an old established organization -- you think of living in buildings with ivy-covered walls. They just knew the way they were going to do it and they weren't going to do it any other way. Some of that flavor still exists in IEEE even today. They almost missed the computer revolution.

Haigh: So do you mean IEEE, the parent organization or the IEEE Computer Society?

Patrick: Well, I was a member of the parent organization in order to get health insurance through them. I had to be recommended. I was recommended, endorsed by Willis Ware of RAND to be a member of the parent organization. As an individual consultant, health insurance is highly prized. If you can get in on a group, and IEEE was a group, it made it much better. But I have a nephew by marriage who's a power engineer in IEEE and they are just pretty staid.

Haigh: And just with regard to the DPMA, I know of one controversial issue which was brought to the attention of the advisory committee during that time when a critical editorial was published about the program, encouraging members to write letters of protest.

Patrick: The engineering computing fraternity, if you will, kind of pooh-poohed all that DPMA stuff. So they had a hard row to hoe there.

Haigh: Yes. So I think your term on the committee was just for two years. Do you have any further involvement with DPMA or knowledge of what happened with the CDP after that?

Patrick: No, I'm afraid I don't. I wrote an article or two for the DPMA journal and I think they published one or two of them. I was sympathetic to their cause, but I was so busy earning a living for myself and my family, I didn't have any more time. You have to have a pretty good sized company behind you before you can spend a lot of time being chair or secretary or president of the DPMA or something like that.

Datamation

Haigh: Then of course you were involved with a number of other groups. You've already mentioned Datamation. So let's talk about them. When did you first become involved with Datamation?

Patrick: Well, first I was an editorial advisor to the magazine. The magazine was owned by a company called F. J. Thompson Publications. Frank Thompson owned it, and he had a couple of minor stockholders, one of whom was Joe Landon. And they had a stable of magazines. There were four or five magazines in the stable, one of which was Datamation, one of which was the R&E Journal. Those are the two I remember at the moment.

Haigh: Oh, I think actually Datamation was originally called Research and Engineering and then it got subtitled, Research and Engineering, the magazine of Datamation, and then there was I think a new owner who thought Datamation was the title to go with so it just became Datamation.

Patrick: True. True. When it was pretty clear that computers were going to take off, well they dropped all that R&E stuff and decided to be the mouthpiece of the growing field. The first editor when it was R&E Journal or whatever it was called was a fellow by the name of Kluge, different from the Kludge that we lampooned many years later.

Haigh: Yes, that was spelled K-L-U-G-E whereas the Kludge Computer Corporation was K-L-U-D-G-E.

Patrick: Yes. Kluge the person - I never met him. Kluge the person was replaced by a fellow by the name of Sandy Lanzarotta and Sandy was a good guy. He essentially put the magazine together and it was his idea to have a stable of advisors who were working in the field to give

him story leads, to answer questions about questionable stories, and have a drink with every now and what not. So I was in Sandy's stable.

Haigh: Was that something you got paid for?

Patrick: I got paid $300 an article for every article they published and they didn't publish all of them. They published most of them, but not all of them. I think I got $200 a month flat for being an editorial advisor, and then later Thompson Publications had computer problems of their own. They were based in Illinois in Barrington, north of O'Hare Airport. So they called up Datamation and they wanted somebody to help them with their accounting and computer choices. They had a Honeywell 200 or something. So I was recommended and I worked some for them. And then Landon mentioned they were getting their typesetting done by a commercial typesetter in Chicago. They passed me on to him and I helped him get comfortable with electronic typesetting and so on. So off and on I worked for either Datamation or Thompson Publications for about 25 years.

Haigh: All right, and what were your impressions of Lanzarotta's performance as editor?

Patrick: He was a good editor. The field was fun at that time, and he managed to carry some of that ridiculousness over into the magazine and he also combined it with some serious stuff. A lot was material that manufacturers would never have published. And furthermore, he made it easy for a busy guy to get published in the magazine, which is different than the year I spent with IEEE trying to get my last anecdote in. So Lanzarotta established the pattern for all of this and did a good job.

Haigh: And then in 1965, Robert B. Forest took over as editor.

Patrick: Harold Bergstein was in between there. He was the guy who was injured when his VW got dumped over and I replaced him for three months. Then after Bergstein left, Forest came. Bob Forest and Bill Rolfe had been rummaging around computer publications, ghost writing house ads for magazines, editing company brochures and so on, and they both went to work for Datamation. And Rolfe was essentially Mr. Inside and Forest was Mr. Outside. Rolfe was a nice guy, but wasn't outgoing and Forest was delighted to sit up with you all night and drink a bottle of scotch.

Haigh: Right. I think Forest had a reputation as a rather lively editor and personality.

Patrick: Yes, he was. That was a deserved reputation. During his regime, Datamation had some annual parties that were usually right before the Fall Joint Computer Conference and those turned into magnificent bashes. Those were the tickets to have. Everybody would call you up on the phone and say, “Do you have any extra tickets to the Datamation party?”

Haigh: So what kind of people would come to the party? Would they be company executives? Heads of computer centers? Writers? Academics?

Patrick: The Datamation crowd was aimed at the “doers” of the field. There were very few academics. We had a friendly connection with Dick Canning of Canning & Sisson. There was one time when we were talking about recruiting people and there was an annotation they were going to put on the ad: "Ph.D. disqualifies" because at that time, to get a Ph.D., they removed all your sense of humor. Everything was stiff and straight and even if it was funny, you couldn’t admit that you were laughing on the inside. And that’s not the lively magazine that Forest wanted to be in charge of.

Haigh: Would you say that Datamation enjoyed controversy?

Patrick: Oh, yes. Maybe not as much as a daily newspaper, but they weren’t above poking fun at IBM. I think Bob Forest is the source of the phrase “Snow White and the Seven Dwarfs” when they were talking about IBM and all the little also-runs most of whom left the computer market during the following recession. Forest sponsored the Kludge Series. Jackson W. Granholm was the first Kludge author.

Haigh: Right. So can you describe Granholm?

Patrick: Granholm was an engineering programmer who never got into management as far as I know, shaved his head, was two inches taller than I was and didn’t have a serious bone in his body. Whenever something happened, he’d see the funny side of it first. At one time, he and I were both working at TRW in Canoga Park, California and they assigned us a closet. They had a windowless closet on the first floor with this big sewer pipe that ran down the wall. So when you’re sitting there working away and somebody upstairs flushes the toilet, that breaks up your concentration. Granholm wrote that up once in a Datamation article.

Haigh: And I think the Kludge articles are primarily associated with Granholm, but you had mentioned that you had written something yourself.

Patrick: Yes. I wrote one as Oswald I. Orthmut and we were just all lampooning the dwarfs and their pursuit of IBM and the antics of IBM itself. We were not respectful of anybody in those days.

Haigh: Right. Now I think people sometimes thought that Datamation had an anti-IBM bias.

Patrick: Well, it may have had a little anti-IBM bias. I kind of think I countered some of that. There was a bias in the whole country to try and level the playing field. RAND loaned me once to the Federal Bureau of the Budget and I worked on an assignment in the Executive Office of the President to try to prepare a computer purchasing policy to be adopted by the United States Congress. And we bent over backwards trying to level the playing field so that the main computer manufacturers got an equal chance at the government business. We regretted it later when many of those manufacturers voluntarily left the field and for reasons like we heard yesterday where you could make all the policy you want at the Washington Federal level, but if Bill Lonergan (RCA) fools with the compatibility architecture of the machine, they're dead and there's nothing you can do about it. The world at that time was concerned and IBM was being sued for their monopolistic tendency. It was a more serious concern then than was any of the concern I saw about Microsoft and what they've done recently. So everybody was trying to help the little guy then, but the little guy was doing it to himself.

Haigh: All right, and would you say that your role as a frequent contributor to Datamation gave you some kind of celebrity within the computer field? Would you find that people would run up to you and start talking about something that you had written? That kind of thing.

Patrick: I did get name recognition and some face recognition from that. I also got leverage where I needed it. In the early 1960s, I was a faceless consultant to IBM. My name did not appear on anything, but I was part of the inner circle. I worked directly for Fred Brooks. When the time came to announce IBM's 360 line, there I was - a trusted advisor to Datamation and an insider on the 360. So I offered and IBM accepted to have me negotiate between IBM and Datamation to publish the 360 announcement. And it was a very hush-hush process at that time. Nobody knew what was coming out, but Datamation had the copy and I think it's in the April 1964 issue, the same month the machine was announced, we had a definitive technical article that we were proud of in Datamation. So there was a case where my interests conflicted in a healthy way.

Datamation Advisors

Haigh: Right, and let me ask you about a few of the other people who were prominent contributors to Datamation. I'll start with Robert V. Head.

Patrick: Oh, Bob Head. Forest thought that we might have too much scientific engineering flavor, and he wanted more of the DPMA style flavor because it was clear that the guys who had big commercial machines were as interesting to Datamation's advertisers as were the guys who had big engineering machines.

Haigh: Probably more interesting because there were more of them.

Patrick: The flow was going that way. It was going towards them. So Forest made Head an advisor and it was the DPMA discussion all over again. Head wrote eloquently about things that weren't really all that deep. But it was the kind of things that those readers were involved in. So on some academic scale, he was farther down, but as far as the big commercial readers were concerned, he was right on top. So he sensed it correctly.

Haigh: And how about Herb Grosch?

Patrick: Herbie. Herb was a clown. He had a Ph.D. and I think it was in physics or meteorology or something.

Haigh: He'd been working out astronomical orbits.

Patrick: And Herb was a self-appointed speaker for the field. Sometimes he made sense and sometimes he didn't make any sense at all. At professional meetings and stuff, he'd get up and he'd speak on important issues and sometimes, everybody got up and walked out. Herb at one time was in the National Bureau of Standards as manager of information technology or director of information of technology or something. He had a good job there, and I approached Herb trying to get him to set up a dictionary clearinghouse for the words of the field, with which the Bureau of Standards could have established a standard dictionary. We had other professions that had standard dictionaries and I was still suffering under the orthocount versus check sum syndrome and today, we're suffering from people that *burn* DVDs. I mean that's a piece of the argot of the field that has now gotten into use. And I had proposed a way for the National

Bureau of Standards to annually publish dictionaries of approved terms and approved definitions and I couldn't get Grosch to move. He had a high leverage position and refused to use it. He got up and he spoke, easy to get on meeting agenda because he was an interesting speaker. But at one time, one of the later DCA meetings was held at the Schlitz Brewery in Los Angeles. And everybody knew at that time that the DCA was a marching and chowder society. Grosch was the speaker and he came on board and we all had been drinking Schlitz beer for two hours and he came on board with some serious talk. We couldn't get him to give way or anything, so we wrapped him in toilet paper. We started at his feet and then wrapped him and when he finished up, he looked like a mummy and he was still - there were three of us - and he was still talking on this serious subject and everybody's quaffing down beer. It was the funniest looking mummy you had ever seen. Herb sometimes was just out of it.

Haigh: Do you think in the end, he made much of an impact on the field?

Patrick: We had some clowns in the field. I guess I'd have to put him in that class. I can't think of a single thing that Herb did that had a lasting influence on the field. He'd probably say the same thing about me.

Haigh: Richard G. Canning?

Patrick: Dick Canning was a personal friend of mine. He and Roger Sisson were the early consultants and advisors. Sisson went to work, I think for Aeronutronics. Canning published his newsletter. The newsletter was aimed at the same audience that John Swearingen was worried about -- at the data processing people who needed to learn and needed a voice. And it was a well-written newsletter. I wasn't interested in all the subjects, but it was good stuff. Dick and I became friends. I never submitted anything to him or published anything with him. He always had a backlog of stuff, and he traveled extensively and in contrast to Datamation he was working maybe six or eight issues in the future. He had folders set up. He'd picked the subject. Had folders set up for these travels, had established contacts. Well, he got bits and pieces and copies and stuff put them in his folders. When the time came when he needed copy, he wrote up the next one. He was a professional viewer of the passing field.

Haigh: And Dan McCracken.

Patrick: Daniel D. McCracken was a Datamation advisor when I was. Dan wrote a couple of excellent books. You might think of him as an early Dummy's author. He wrote a FORTRAN book that was a collector's item where he explained FORTRAN better than anybody else had explained it - manufacturer or otherwise. Dan wrote, as I recollect, three books. The one I

remember is the FORTRAN one. Then as I recollect, Dan got involved with religion and he disappeared. He may have been working for the Lord, but we didn't see much of him after that.

Haigh: That's interesting. That would that have been in the 1970s and the 1980s? I know he was involved with religious things, but I hadn't realized that he had decreased his involvement with the computing community.

Patrick: Well, he may have tried to stay up with computing, but at that time, I was running about as hard as I could just to stay abreast of the changing field and you couldn't go off on some tangent and come back or you'd be three generations late.

Haigh: So when do you think that happened? 1970s? 1980s?

Patrick: Oh, dear. I haven't talked about Dan in so long, I wouldn't know.

Haigh: All right. I'm seeing the names of a couple of other advisors. Apparently Howard Bromburg was an advisor at one point.

Patrick: Yes, he was. He was a friend of Forest's and he wrote fairly well. He was into a lot of professional services activities, kind of a hail-fellow, well met. We had some wonderful bottles of scotch together.

Haigh: Who did he work for? Was he self-employed? On my desktop I'm seeing RCA.

Patrick: Yes, and then he went independent and for many years was an independent consultant operating somewhere in the San Francisco Bay Area. I haven't heard from or seen him in years.

Haigh: I don't know much about him. I've quoted several of his articles about COBOL and Compilers. So I'm assuming that he stayed specialized in that area.

Patrick: The things that he wrote about and the things that Head wrote about were pretty close. They were writing to what we engineers call the great unwashed but that was the biggest piece of business.

Haigh: He'll probably come up in the RAND context too, but Paul Armer was I believe also an editorial consultant.

Patrick: Paul Armer was a World War II weather officer. How he ended up at the RAND Corporation, I do not know, but he was my contact at the RAND Corporation and he gave me a consulting contract to RAND and when he moved on, my contract stayed and I worked for RAND 33 years and my recruiter was Paul Armer.

Haigh: All right. So we'll talk about RAND next then. Just to wrap up on Datamation; were there any other people involved with Datamation who you think made a particularly important contribution?

Patrick: There's a lady by the name of Cleve Boutell and she was an outstanding avant-garde artist and responsible for all of the award winning covers that Datamation put out. I don't think Datamation would have been a resounding success without Cleve. She put out interesting covers month-after-month. Some of them or many of them got awards from the Cover Artists Association or whatever it was called and as far as I know, nobody ever had any trouble with Cleve. She just did an outstanding job.

Haigh: And do you remember any of Datamation's competitors like Business Automation?

Patrick: As far as I recollect, Datamation didn't have any competitors until Computerworld came on the scene. The Datamation senior management had gotten comfortable and Computerworld came on like gangbusters. Somehow, Datamation refused to compete. Although some of us down below were agitating to do different things, Datamation wanted to continue doing as they did and they continued, but it ended sooner than it had to.

Air Force Officer Classification

Haigh: All right. Let's move on to RAND. So, when did you first come into contact with the RAND Corporation computer people?

Patrick: Right after I left Computer Sciences, November of 1959. And at that time, nobody had ever heard of an individual freelance consultant and they were sure I wasn't going to be a success. And Paul Armer wanted to hire me when I crashed because he wanted me as an employee. I worked through several managements at RAND as they evolved and so on. But, RAND was a good client all the way.

Haigh: I think you'd said that you had a contract where you were always nominally doing one day a week for RAND.

Patrick: One day a week, unless I had a higher priority assignment. And RAND negotiated with me a lower daily rate than anybody else did but I had the advantage of a flexible schedule. Now, that may sound like I was floor sweeping and picking up and so on, but I managed to do some fairly interesting things at RAND. Two of them -- well, three of them, I guess -- come to mind that should be emphasized. Your dissertation contained a picture of a pyramid.

Haigh: Yes.

Patrick: It was a six-sided pyramid.

Haigh: Yes, that was taken from a 1969 paper that you published. If I remember right, that one was the presentation to the computer personnel research group of the ACM, but I'll verify that. You carry on with your point.

Patrick: Previously, what that model was really made for was a presentation to the Air Staff of the United States Air Force, all generals. After my Air Force tour, I'd followed subsequent AF computer activities. The Air Force had people who had computer skills, but they were not identified with the proper skill number. In the Air Force parlance, that's called an MOS, Military Operational Specialty. I'd seen guys sent to Korea when we had a crying need for officers with computer skills here in this country, and they were sent to Korea because at one time they flew C-47s. And they sent them to Korea. They lost all their computer skills, and some of them didn't even come back from Korea. So here was the Air Force wasting a very precious resource. Through the auspices of RAND and the Air Force politics, we managed to establish a Military Operational Specialty for computer skills. And what that practically consists of is an officer assignment desk in San Antonio, Texas where the orders for all officers who have computer skills pass through. And they leave the orders alone if it's in the best interest of the Air Force, and they change them if it's not in the best interest of the Air Force. And that was the way the Air Force hoarded their precious computer talent in the days when they needed it very badly. The pyramid is the way that we explained to the air staff that there were six basic skills for company grade officers up to Captain or Major: computer operations, programming, analysis and so on. But if they didn't round out this skill base, they never would get the top senior level officers they needed.

Haigh: So, actually, I have the diagram here representing the computer field. And there's a three-dimensional pyramid version. The pyramid depicts specific six areas - - operations, research, sales, hardware, software and applications. So, in the context of the Air Force

operations, applications, software. On the other hand, with hardware, it seems like they would be purchasing rather than building.

Patrick: You have to specify it before you purchase it.

Haigh: All right. How about sales? How does that fit in?

Patrick: The military has needs. And the people that see the needs and are willing to dedicate part of their careers to fulfilling those needs become advocates of the need and the solution. And they move around the Air Force and into the Pentagon and the senior levels just like a salesman does, saying, "Hey, we've got this problem, we need body armor in Iraq, and the body armor we have won't deflect these kinds of slugs, and this is the company that's got the technology to fix the body armor." So that is a sales function. They don't call it sales and marketing. But it looks just like what the IBM salesman did when he came by and said, "Hey, looks like you have this need, and I have this solution to it."

Haigh: And I should say somebody who wants to see more of that can find it in Robert L. Patrick's paper is called "Selection Starts the Cycle," the proceedings of the 7th annual SIG-CPR Conference on computer personnel research published by the ACM in 1969. So, that kind of conception of the field as a whole, and the idea that the people on top need to know something about all those areas, what led you to come up with that?

Patrick: Well, I had seen that the Air Force was incorrectly assigning their people, and I had had consulting assignments in the various areas. And from time to time, I'd meet military officers who had some of those skills and they had outstanding intellect and abilities but were never going to make admiral or general or even colonel. And the only way to do that was to formalize the way the Air Force handled their personnel. And that was the Air Force Way. They had MOS codes for fighter pilots and shred-outs under those for what kind of fighters you flew. So if they needed fighter pilots, they knew these guys' names and knew what they'd flown, and if you had to establish a fighter squadron, you knew whom to pick. But they didn't have that same discipline in computing until we called it to their attention.

Haigh: Okay. So, I can see the personnel management side, where you're looking for people who need to be assigned in a particular area. You want to pick up the search string and find the skills. But it seems to me that with that representation and saying that the people on the top of the profession would want to earn a lot of money and be in charge of things need to know something about all the areas, that you're really making an additional point there, that you believe that all these things fit together as part of one profession, and you believe that the top people should be generalists who know a little bit about each area.

Patrick: Or at least be skilled in some area and knowledgeable in the others.

Haigh: And I was wondering, do you think that's related to your own experience where you'd worked as a consultant, worked in so many different roles that you thought they needed that kind of all around ability?

Patrick: I guess if you had dissected me, that's what I'd been doing. Both RAND and I were concerned that the Air Force was buying things it didn't understand. And there were senior officers that had to sign off on these purchasing contracts and they had no idea what that contract said when they put their name on the bottom of it. So previous to my MOS work, a bunch of us would talk at AFIPS conferences and so on, that we had to do something to raise the quality of the government procurement, that they had contractors selling junk to the government. And it was all done very systematically, but it was still junk. So, a consortium of about twelve companies got together and we said, what we really need to do is to train, literally train, the current crop of generals and admirals so that they know what they're signing. And if we do that, we also establish this experience pyramid. Pretty soon, the system will be self supporting, have guys at various levels of experience who grow as they climbed up to the top, they get to be colonels. Good colonels get on the selection list for generals. Some of them will make general rather than fighting in the trenches. Some of them are going to be in procurement and we will get better procurements. So, RAND had a very special relationship with the Air Force. And Willis Ware and I worked the RAND politics around and got RAND management to go to the Air Force chief-of-staff, Curtis LeMay, and he set up with a stroke of the pen, the Department of Defense Computer Institute.

Haigh: And what year was that?

Patrick: 1964.

Haigh: And do you think this Institute helped to address the problem that you've been talking about?

Patrick: Yes. This was set up specifically to give some training to the senior military. We started from an Air Force base, but it was a DOD function, we had generals and admirals as students. Everybody wore stars that came, with the exception of one Marine colonel who was a general designate. He asked his first question on the first day and got such dirty looks he didn't ask any more questions for the whole week-long session.

Haigh: And that was a project that you did with RAND?

Patrick: It started out at RAND. LeMay established it as a DOD function. It was assigned to the Navy as a host, and they had some facilities we could use in the old Washington Naval Yard. And these twelve or so civilian companies all contributed lecturers and Willis and I parsed out who was going to speak on what and I was acting director for the setup and through two cycles, until the permanent staff could be brought in. And each of the twelve of us, me included, gave a lecture in his specialty during a week-long session to these generals and admirals.

The RAND Corporation

Haigh: All right. Let's just move back and say a few more words about RAND as a whole. So, how would you say RAND was as a place to work back in the very late 1950s, early 1960s?

Patrick: Well, RAND was an interesting place in the late 1940s and when it was set up. They were set up to acquire -- pardon the expression -- "genius level people" and let them have freedom to work on interesting projects, under the theory that, if they saw who signed their paychecks, the project would be somewhat oriented towards Air Force and governmental activities. So when I got to RAND in 1959 that was their form of management. And they were totally unprepared to deal with a graded work force which had some entry level people and some -- everybody had a college degree -- but some entry level people, some mid level people, some high level people, and some very senior people. So, they didn't manage us very well, and a lot of the good talent RAND boiled off and went some place else. A premier guy is Paul Baran, who is given a lot of credit for the concepts behind the network.

Haigh: Switching networks?

Patrick: Yes, Paul did that because he was interested in making a survivable military communications network so that if a bomb or a terrorist or a natural event took out one link of it, there were alternate links and you could still get your communications between the forces to the headquarters. But RAND management didn't seem to realize how big computing was going to be. And although they were in on the ground floor, a lot of the good people boiled off and left. And as part of that, it's almost like a culling process in reverse. They left behind a lot of average people. So, in the later years, RAND had a collection of some good people, some average people, and some I would fire.

Haigh: All right. And can you say some words specifically about Willis Ware?

Patrick: Willis was a renowned pioneer of the first order. He had a Ph.D. in electrical engineering and he had worked for a hearing aid manufacturer. He found his way to RAND. He recognized the importance of computers and he wanted to build one, and he sold the RAND

management on building the JOHNNIAC. It was an early computer, and it worked, and it was moderately reliable for that time and age. Not by today's standards, but by those standards, it was a good, solid machine. He had good people around him, and they kept it maintained. And then the age of the JOHNNIAC passed. New equipment was available. The primary lessons on the JOHNNIAC had been learned. And so they wrote an online system for it called JOSS. And JOSS was in some sense a competitor to BASIC, but JOSS was not supported and documented and sold, and BASIC was. And JOSS whimpered over there in the corner and died.

Haigh: Yes. I was thinking more about Willis Ware as a person.

Patrick: Oh, I must have had three hundred lunches with him. And we'd talk about the computer field. He was at one time president of AFIPS.

Haigh: I think he was one of the main people behind its founding. AFIPS was the U.S. complement to IFIPS.

Patrick: Yes. I think he and Keith Uncapher, who also was at RAND at that time, were the spark plugs in that whole international computing umbrella organization. And RAND, being fairly free with assignments, supported him financially and gave him time to work on that, although it wasn't directly related to government work. I think their AFIPS work was good work. Willis and I worked very closely together on an Air Force system called ALS, the Air Force Logistics System. After the IBM S/360 was produced and Fred Brooks had a thousand programmers there, the Air Force set up a logistics system development team to provide worldwide logistics control over all their parts in inventory. It was based at Wright Field, Dayton, Ohio. They had 1500 programmers all managed by these same military officers that we'd been trying so hard to train. So, the military people who were in charge of the logistics system had one year's experience three times, whereas the guys who worked on the IBM development had many years experience many times. The officers just weren't up to the task. So RAND organized quarterly trips by a consulting team. We went to Wright Field every quarter and we tried to discuss problems and solutions with the guys who were managing this big logistics system development. Every time we went, they had followed less of our advice and things were going further downhill. After a couple of years of this, this got so big and so expensive that the Congress insisted that the Air Force call this a line item in the Air Force budget. There was some maliciousness involved there, because when the Air Force called it a line item in the budget, Congress could do a line item veto and kill it. And that's the only computer project I've ever worked on that involved an act of Congress.

Haigh: And back to RAND, I'd like to ask about Fred Gruenberger and the RAND Corporation symposium series that he was responsible for.

Patrick: Well, that was another goof on the part of senior RAND management. Gruenberger recognized a need for technical information interchange. And he organized the RAND symposiums to bring leading edge technical people together. And did that very well. And I was invited to several of those, and being a member of the Datamation staff, I managed to get RAND symposiums into print. Fred taped them and transcribed them, and they ended up being inch-thick books. Datamation abstracted sections and published them, I seem to recollect some two-part articles in consecutive months in Datamation, to try to get that same kind of information to a broad population because at that time the only education that was taking place in the field was manufacturer specific, and it was specifically to sell hardware. There weren't any computer sciences courses in colleges at that time, and there weren't any Ph.D.s in computer sciences. All of us guys had training in some other discipline.

Haigh: Now as a Datamation editorial advisor, were you personally responsible for getting the excerpts published on the occasions when they were?

Patrick: I would say I'm probably as responsible as anybody was. I'd talk to the editor of the moment, whether it was Lanzarotta or Bergstein or Forest. I'd say, "We're going to have this symposium." They were aware of what previous symposiums had been. There were notable names who had attended the symposiums. The transcript was due out on March 15th or so. And we would spend a month editing it. Our close date was April 15th, and it would hit the readers on June 15th. So we weren't but a couple of months behind the actual symposium with the Datamation articles, and our circulation at the time was over a hundred thousand. So we were a cog in the educational scheme.

Haigh: And are you aware of any occasions where you would say that the ideas and the discussions at these meetings had any particular effect in the broader world where ideas that surfaced during a meeting were acted upon or had an impact?

Patrick: Well, the RAND tablet was discussed there. It wasn't a commercial success -- RAND had one, but the mouse came out of the Xerox PARC people. It was a graphical input technique, which was the predecessor of the mouse that we all know as a way to move a pointer around on a screen. And that was an outstanding development that comes to mind immediately. I'm sure there were several more.

IMS Database Management System

Haigh: Okay. Well, let's move on to a different topic. We don't have an enormous amount of time left, but there is at least one of your consulting projects I'd like to ask more about beyond what you've written in the notes, which is about what became the IMS/360 database

management system. You wrote that you were involved with this project as a consultant for the Space Division of Rockwell?

Patrick: IMS was an outgrowth of a legitimate need. The Space Division was building Apollo capsules and needed to track current engineering drawings for those Apollo capsules. And if the engineer that owned a part had to change that part, he wanted to somehow inform all the users of that part there was a change pending.

Haigh: Oh, so that basically came down to a parts explosion problem, did it?

Patrick: It was really more a parts control situation. Space had a predecessor of IMS running on an IBM 7010. And when we started planning to get rid of the 7000 series equipment and go into S/360s, we needed to have software that would support that same application because they hadn't gone to the moon yet.

Haigh: All right. So, by a precursor, do you mean a file management system?

Patrick: It was a file management system, but there was an important difference in IMS. The computing field was so complex that if you had the data and the applications programming intertwined, which is the way we originally built applications programs, and you made a change to the application, you had no way of making sure that change didn't affect the data in such a way that some downstream program was impacted. I remember from a slightly later date, Hughes Aircraft had a suite of financial programs for keeping track of costs and billings and account closings, and the financial suite had 126 programs in it. Well, they used a monumental amount of data and more data was flowing in every day. And if you could control and manage the data, define the data, back up the data and change the data format without impacting that suite of programs, that was a big simplification in the operation, in making sure that when you set out to run an inventory that it was going to run to completion and it was correct. Well, that intellectual split between the management of the data and the applications processing was recognized by two guys, Uri Berman and Peter Nordyke. They did it on the 7010s. We carried that profound concept over into the IBM systems on the 360s, and IBM and North American were co-developers of IMS. Then Caterpillar Tractor got in on it as a later developer. And eventually it was a big seller for IBM.

Haigh: Now, a minute ago you'd said it was Rockwell. Were they at that point part of the same group as North American, or were they a separate company?

Patrick: Originally it was North American then it changed to North American - Rockwell, and finally Rockwell. IMS/360 was developed at the Space Division of Rockwell.

Haigh: So then, when was this?

Patrick: That'd be 1966 or 1967.

Haigh: Okay. So they had a working application for the 7010, and they needed to migrate it to the 360.

Patrick: Yes, sir. And we did it in a generalized way, and it really paid off.

Haigh: So, internally this also solved the hierarchy problem so that each parent part could have a bunch of child parts?

Patrick: Yes, yes.

Haigh: That you want to track. So, now, you say here that one of the innovations was to run as an application under OS/360. So, had the previous system, been a part of the operating system itself?

Patrick: Yes, let me back up for a minute. Under the previous operating system on the 7010, if the system died, your application died, and you had to restart the application. When OS/360 came up, OS allowed you to run multiple tasks, first MFT, multiple fixed number of tasks, and then with MVT, multiple variable number of tasks. OS/360 maintained the purity of individual applications. Because of the size of a big 360, you might have four applications running in parallel to utilize all the I/O capacity. And if you had a mistake in one of them, you couldn't impact the other three applications. You might have 500 terminals you're going to drop. So, OS had the mechanism in it to keep these applications separate. And we brought up IMS as a top priority application. So, if another application faltered, or if OS died, IMS hesitated for a moment, but did a hot start on the operating system, and the application set was going again, and you didn't have to rebuild the database. And for 24-hour operation with precious files, that was very important.

Haigh: All right. So let me ask a few more questions about IMS. What was the process by which it went from being an internal Rockwell application to becoming an IBM-supported piece of software?

Patrick: Rockwell had a manager, Dr. Robert R. Brown. He was an ex-IBM applied science employee. And since I'd been working with OS/360, I counseled making this an independent

application running under OS. Don't take any special privileges out of OS to make this system run, because if you do, you're now dependent upon the OS maintenance process, which would have killed us. R.R. Brown took that concept and he went to IBM, and he sold them on a joint project. So, IBM had contributed half of the programmers and Rockwell contributed half of the programmers and we put them under an IBM manager, and programmed to IBM Type 2 software standards, Type 1 being the programs that were built in Poughkeepsie. Type 2 was built in a user environment to IBM standards. So we built it to Type 2 so IBM could carry it off and sell it to other people. And the reason we did that is that we didn't want to do the maintenance on IMS. If we did it to IBM standards, once we had it done, then IBM's maintenance of the package would benefit Rockwell and we'd get our programmers back. Then we could do applications and didn't have to maintain the software.

Haigh: Right. And do you know what year this was being produced internally?

Patrick: It was going to be 1966 or 1967, because that's when the 360-65's were available.

Haigh: Right. And then it would be a couple of years later that the IBM version surfaced to the outside world.

Patrick: It didn't take that long, about a year later.

Haigh: Are you aware how much changed from the internal version to IBM's IMS?

Patrick: Not much. What they called IMS/360 consists of front end code, which handles application terminals and control, and the back end code, which handles the database. The back end code was called DL/1, Data Language 1. DL/1 was the concept that Berman and Nordyke had come up with on the 7010. They had the concept and they had a running program, but they couldn't write it up so anybody could understand it.

Haigh: What's Data Language 1? The field definitions? The links between different records?

Patrick: Yes and the control package which provided queuing, backup, and restart. I was assigned to find out if this thing worked, and it did, and to write up a description of the package. And I did. In 2005, I gave the original write-up to the Computer History Museum. After IMS came out, the front end of it was attractive to big users who were managing inventories. But DL/1 was attractive to a lot of people. So, DL/1 with a new front end on it was in CICS. And CICS was a very popular transaction processing system for the IBM Corporation. If you count numbers of installations, IMS was only in big installations, and hence that's a small number, a

few hundred. But I think there were a few thousand CICS installations that had the DL/1 code in them. Burt Grad can tell you all about that because he for some years was Mr. CICS.

Haigh: I just have a couple of other questions about how IMS was working. So, would the records be stored on tape or on disk or could it work with either?

Patrick: No. The update was on disk with backup going to tape.

Haigh: So my understanding of what you've said is that you could have a whole bunch of simultaneous application programs running and one copy of IMS in memory as a high priority user application that would service multiple applications and multiple users simultaneously.

Patrick: Yes, sir.

Haigh: So, this was designed for use with online applications?

Patrick: Yes, sir.

Haigh: Now, was it designed to be used by application programmers or by users with ad hoc queries or by both?

Patrick: There was no programming capability within IMS. It was a production package for user query. The programmers didn't develop any code online. IMS was supporting terminals out on the factory floor. And these people didn't have any programming capability. If they had troubles, they picked up the telephone.

Haigh: All right. So, there was nothing like, say the Mark IV system or RPG, where you could run an ad hoc query against the database?

Patrick: Ad hoc, there were some programmers who supported the system, and they had the ad hoc query capability. But nobody got directly to that precious database.

Haigh: So, if you were an application programmer, and you needed to work with some data that was in the database, how would the application programmer handle it? Would you embed some pieces of a special language in your code? What would you do?

Patrick: Bob Brown I think was the first man to create or to name the concept of the 'database manager'. A database manager was an individual ex-programmer who knew the data and worried about backups and corrections and so on. So, if you were an application programmer you went and presented yourself to the database manager and he told you what was in the file and how to get at it and approved of your access methods and so on.

Haigh: Okay. So it would be like a database administrator?

Patrick: He was, in fact.

Haigh: And, was it called a database management system back when it was an internal project?

Patrick: Yes. If I could remember its earlier name, you'd recognize it.

Haigh: All right. So I'm just thinking, just purely on a technical level, presumably the application programmers are writing in COBOL?

Patrick: That would be a good assumption.

Haigh: So, I know COBOL had built-in data features, but they're really sequential, right? They don't really work with random access. So presumably, you have to have some way for the COBOL programmer, when they reach a point where they need to retrieve or update a record, they must be able to ask IMS to get the record for them. So, can you remember how would they do that? Would they call a procedure?

Patrick: As I recollect, there were packaged procedures for that, and if you were processing the file in serial sequence, there was a condition code that told the database system it could read ahead because you were going to work down this file in sequence. If you're processing in random sequence, that condition code was set differently and after you made a request you had to wait for every key you presented, you had to wait to get the response back.

Haigh: Was there any need to support online and batch applications in a different way? Or could both kinds of applications use IMS?

Patrick: No. Batch was just another application program. And that's how you did your backups; that's how you did your long reports and so on. All that came out in batch.

Haigh: Did you have any awareness of earlier work in the area, particularly Charles Bachman's IDS produced at General Electric?

Patrick: We didn't pay a lot of attention to what was going on outside of the IBM world and more specifically we had maybe 500 application programs to move off the 7000 series machines onto the 360s and we looked for common pieces of those programs that you could develop into software, for that meant you only had to program that routine once and there would be less total code to develop for the 360. So we weren't really interested in advancing the state of the art. We were interested in getting a capsule to the moon.

Let me talk about performance for a minute. I have had maybe a dozen computer performance assignments. I learned a little bit about performance and measurement when I was in the Air Force. When we built IMS, the question was can it carry the load? Being on a new computer, the 360-65, using discs that nobody was really familiar with, and supporting terminals that were all over everywhere, the question is what's the performance? So we had two 360-65's in the room and it was possible using standard IBM hardware to connect the channels of those two Model 65's. So we set up a driver program in one model 65 that supplied terminal transactions to the machine under test. So we drove the IMS machine with a front end model 65 and did performance measurements, some hardware, some software, to send a variety of transactions to the machine under test and collect its responses, and we published an engineering report. As I recollect we could handle 4,000 transactions an hour tracking parts and getting status. Throughput was sensitive to the mix of transactions we had to process. We finished up the IMS project with an engineering performance analysis of what we had created.

Data Center Audit

Haigh: Now let's skip over some of the other work you did and talk about the data center audit. In your writings I think you did a good job describing what the idea was and how it worked, but I was wondering if I could ask you to generalize a little bit over this period from the late 1950s when you set up as an independent consultant through the 1970s. Do you think much changed in the way that data centers were organized, especially at the beginning of that period? Initially, you have all these people who really have punch card backgrounds and are trying to get to grips with computers. Over the 1960s and the 1970s, did data center management get better?

Patrick: Both the customers got better and the management got better. The engineering customers and the commercial customers didn't know what to ask for when we started out so they asked for a system that would do exactly what the previous system had done and as a result of that we did a lot of superfluous calculations that were hard to check out and sometimes the previous systems had been producing erroneous results and we did the same thing until we

figured it all out. So as general (computer) education improved and as we started getting customers who had had some computer training in college or had been to trade schools, the computer applications designers had to match those customer skills. So the applications were much more sophisticated, the price per instruction produced went down and we were much more efficient in getting the computer integrated into the organization.

Haigh: One of the things that the data processing managers always cared a great deal about was who they reported to, where their department fit and were they reporting to the accounting manager who reported to the finance V. P., and they really wanted to move their way up the organizational chart and ideally to report directly to the CEO. Did you see much evidence of that kind of rise?

Patrick: Yes. Somewhere in this period, we're talking about maybe in the 1970s; it's one of those 'be careful what you wish for' things because they established a chief information officer or some equivalent.

Haigh: Maybe a V. P. MIS?

Patrick: Yes, and they put all this under him. The engineering manager was tired of getting excuses from the computer center and the financial manager was tired of getting excuses from the computer center. They just wanted to have one guy who was responsible for providing those computer services so they put those organizations together, sometimes with a great deal of blood, sweat, and tears. Some people left after it was put together, but the current organization has information officers at the vice presidential level. The budgets got big. Five or six million dollars a year was not an unusual budget in those days. Budgets got so big you had to have somebody who wasn't a clerk to worry about them.

Haigh: It seems like the transition to the third generation machines, with random access storage on hard drives and online applications, was an extremely difficult one. How long would you say it took for most computer installations really to get to grips with that?

Patrick: Five years.

Haigh: So if the machine would have been installed in, say, 1966 then they would only be starting really to get comfortable with it by the early 1970s?

Patrick: Yes. Now the machines we put in in 1966 had performance problems so you wasted a lot of time. We put in a Direct Couple setup at one of the Rockwell divisions and it wouldn't run

24 hours' worth of work in a day. It was slower than the previous machines so we had to solve the performance problem immediately because pretty soon you run out of weekends to catch up. Of course most everybody's unhappy because if you're not first in line on Monday you are in a queue and there's some additional delay involved. So we had to solve the performance problem and that was done mostly by IBM and some by the early customers. Then after we did that, we had to solve these organizational problems because some of the organizations were too weak to assimilate the new complexities. I was sent on a consulting assignment to Switzerland. There were three big banks in Switzerland and they all had almost identical IBM installations. Two of them were running successfully and one of them was floundering and that's the one I was sent to visit. It turned out that the other two shops had been through three generations of computers. They had started out with 702's or 705's, transitioned to the 7000 series, and translated all of that to the 360 series. They had strength and experience in depth. The third shop was a relatively recent conglomeration of smaller banks and their DP people were running on their first big system and the salesmen had sold it to them and the management had bought it because the other guys had similar configurations and they had as many accounts as the other guys did but it wasn't working and that's why I went over there trying to help.

Haigh: As data processing centers came to grips with this, do you think there were any particularly important software inventions or managerial innovations that played a part in improving matters, any ideas that you think really made the difference? There are always these ideas like structured programming or new ways to manage the IT department that are floating around. Do you think any of them really made a difference?

Patrick: No, I didn't see that many breakthroughs. There was a lot of talk about all that but I didn't see that any of that helped a lot. The biggest determinant of your success was talent. If you had a bunch of dumb guys, they could structure all the programs in the world and then you got dumb structured programs. The innovation I saw in that period added another level of complexity because when we first brought up these online systems they were on dedicated communication lines. Then we started going into shared communication lines. It was probably Sabre (the airlines reservation system) that pioneered that complexity and then guys down at the Cape Canaveral wanted access to the database we had in Downey, California, and so we had to have long communication lines which were not dedicated solely to one purpose that served geographically dispersed terminals. So we added geographic dispersion and that made us all stand up and work harder.

Haigh: As the packaged software industry for mainframes developed during the 1970s, do you think the spread of packaged software made a big difference to how things were run?

Patrick: Well, we thought it did. But it depended on how well the package fit your needs. In many cases it ended up being that we had just as many programmers looking after the packages and making the modifications to them. The key was slowing down the customers' demands and that took us some time - simple, old cost accounting systems started to bring that under control so that a customer wouldn't insist on everything he thought of. Then he asked for just the function he needed, that was a big step.

Being an Independent Consultant

Haigh: Returning to your career, you had said you thought you were really one of the very first independent computer consultants?

Patrick: I may have been the first.

Haigh: Over the decades, how did you feel basically about your competitors? Did who you thought of as being your competition change over time? Did they become more or less threatening to you?

Patrick: Well, I don't think I ever had a competitor, the reason being is that somewhere in the mid 1970s, about the same time that big information systems got established, the field specialized. There were operations guys who were very skilled at running big systems but who had never programmed. There were programmers who were very good at taking somebody else's specification and writing code and testing it but they'd never done an analysis. The early people in the field, and I'm sure there were more than I, had experience in all of the six facets that we were talking about on the little pyramid. Later, and it's even worse today, it took a VW bus full of people to have the same kinds of personal hands on experience that I was able to bring to my clients. As we heard yesterday these body shoppers weren't doing a lot of training, they were not doing any cross training. They were trying to make a buck and the quality of the outside advice you got went down as the skills you needed went up.

Haigh: You had said before, that you had figured out that if you were to expand your business and bring in other people you would need to be quite a bit bigger. Did you ever feel tempted to change your business model or to go and work for anybody else?

Patrick: Well, Booz-Allen, a big consulting firm, tried to hire me once but we would have had to move to Chicago and drastically change our lives and work for a big company in a job that didn't look like it was going to be successful, so that didn't do much for me. I've never been an IBM employee despite what your thesis said. I've only worked for IBM as a consultant. My last job as a consultant in 1992 was a big insurance company who had listened to a lot of big-time

advice from one of the big-five accounting firms and they were trying to follow that advice and convert from 360 oriented, some online, some batch, to a fully online environment using top down and inferential database methods. Their specifications had all of the right words in them but the crew could not absorb all of that and they were spending a lot of money and every milestone date was missed. So the vice president of finance found me and had me come in and he paid me directly so I didn't have to work though anybody. My relationship with his people was that I'd ask questions and they'd answer. This super development approach just wasn't going to work the way they thought and I had to counsel him, don't go for the moon here, back down a little bit and don't even try for the top of the highest mountain, bring your goals back a little bit to where they are doable with the staff you have. He had a 600 man staff and they weren't going to make it, turnover was high and everybody was unhappy and morale was poor and they were missing all their dates and every day they learned some new wrinkle and they already had too many wrinkles that were not working.

Conclusions

Haigh: I will finish up with asking you two general questions about your career and then you can make your closing statement and show that chart. The first one of those would just be that looking back over your career as a whole, what would you say would be your single biggest disappointment or regret either about something that you did yourself or just about a direction the field took that you think was the wrong one?

Patrick: Well, I guess my biggest regret is that as an individual consultant you don't have much leverage and you can give good advice but the customers don't have to take it; they have to pay for it but they don't have to take it; and there were several cases where I thought I was really leading them in the right direction and then they didn't go - when the client doesn't follow - and these have to be viewed as at least partial failures on my part because there weren't but two of us talking and I was one of them. So some of that was disappointing and tough.

Haigh: Now reverse my question, what would you say is the single accomplishment of your career of which you are most proud?

Patrick: I had fun. Every day I'd get up and I was delighted to go to work. I learned something every day and I enjoyed my work. In 33 years I canceled three contracts because they were unworkable. Other than that I worked for 122 different companies in 33 years and many of them gave me four, five, six assignments. I must have had 300 different assignments. I have one little chart I'd like to close on. I looked through all of my assignments and although at the time I thought each one was unique I found that they fell into nine different flavors. I didn't do much programming and I didn't do any personnel work because that looked like a swamp, but the assignments I had fell into the nine categories shown on the chart below. As I

reviewed the past assignments I had, I asked what was I doing down there? In one case I was talking to the vice president or the president or somebody, who in the consulting business we call Mr. Can Sign. He is the guy that can pay your bill. You don't have to go any higher for approval. When we built IMS, the president of the Space division of Rockwell sent for me and he said, "I just got this job and I'm spending $25 million a year on computing and I don't really know what I ought to know. Can you help me?" I asked, "Will you read?" "Yes." So I pulled together a very carefully culled reading list for him. The reading list I gave him was suitable for a senior engineering executive who suddenly inherited a computing mess. I said, "Here is what you ought to know." I gave that selected set of documents to the Computer History Museum last year. They have my original set.

Haigh: For the benefit of those who may be reading the transcript, I will just read your chart:

Flavors of Consulting

Confidant to a senior executive
Application, definition, and design
Programming methodologies and standards
Machine operations and performance measures
Conversions between generations and steps up in service levels
Project planning and management
Site surveys and catastrophe planning
Technical writing
Troubleshooting.

Now I want to return to the first question I started with, your background as an engineer.

Patrick: Mechanical Engineering.

Haigh: It seems one of the sensibilities that you brought from an engineering background is this concern with documenting processes and improving them and improving throughput. Are there any other aspects of your initial training and background that you think have helped you in these nine different areas?

Patrick: Measurement was a key. Sometimes you were measuring keystrokes, sometimes you were measuring the volume of work and relating it to the staff required, sometimes you were measuring the performance of computers, but as somebody said: "In the land of the blind the one eyed man is king". If you've got numbers from his organization, a client tends to listen to you.

Haigh: Thank you very much.

Additional Materials Provided

Prior to the oral history interview on February 16, 2006, Robert Patrick also provided various other materials to Thomas Haigh and has also given the Software Business History Committee other materials to be included with the interview transcript. In addition, Mr. Patrick has also donated a number of his personal files to the Computer History Museum which can be accessed by authorized researchers. The following donated items have been attached to this transcript:

1. A list of the 70 Datamation articles which Robert Patrick wrote or co-authored starting in 1959 until 1984.
2. A reference list for a few of the other articles which he wrote for other publications and reference to a previous oral history interview transcript.
3. *Consulting Adventures: An Autobiography by a Computer Specialist*
4. *Robert's Reminiscences: A Semi-serious look at a Curious Life*
5. *The Roots of OS/360*
6. Three selected vignettes: *Early SHARE; Operating Systems; Commercial Compiler*
7. *Milestones in Computing*

List of Datamation Articles (Comments on the Passing Scene)

(Seventy outings in Datamation magazine by R. L. Patrick, consisting of articles, reports, and reviews during the period of Big Iron 1960 - 1984)

Special Notes: Datamation was an avant-garde magazine and the published titles are not always descriptive of the contents. Most of the following were solo articles under my own name. A few were co-authored and a few were written under a pseudonym.

No.	Date	Title
1	5/59	One Compiler, Coming Up (CSC Announcement)
2	5/60	A Customizable Computer (Article) and 9/60 sequel
3	5/61	The Gap in Programming Support (Article)
4	8/61	Documentation - Key to Promotion (Article)
5	9/61	1961 Rand Symposium, Part I (Participant)
6	10/61	1961 Rand Symposium, Part II (Participant)
7	11/61	1961 Rand Symposium, Part III (Participant)
8	11/61	Audio Technique Cuts Coding Time (Article)
9	2/62	Contracting for Computer Services (Article)
10	5/62	How to Market a Kludge (Orthmutt Article)
11	6/62	Let's Measure our own Performance (Article)
12	10/62	1962 Rand Symposium, Part I (Participant)
13	11/62	1962 Rand Symposium, Part II (Participant)
14	1/63	The Maturing Field (Article)
15	5/63	The American Way (Article)
16	10/63	So You Want To Go On-line (Article)
17	12/63	Prayer of a Computer Specialist (Editorial)
18	7/64	Measuring Performance (Article)
19	8/64	Centralizing Computer Studies (Panel)
20	11/64	A Panel Discussion on Time-Sharing (Report)
21	2/65	Collector's Items (Review)
22	9/65	Patient's On-Line (Article with Rockwell)
23	10/65	Interactive Session on Interactive Software (FJCC review)
24	1/66	The Future of Private Transportation (Article)
25	6/66	Not-So-Random Discs (Article)
26	9/66	Configuring Multi-Tasking Systems (Article)
27	1/67	Computing in the '70s (Article)
28	6/67	Planning Checklist for a Computer Installation (Assisted)
29	7/67	EDP's Wailing Wall (Editorial)
30	9/67	10 Years of Progress? (Article)
31	11/67	Time-Sharing Tally Sheet (Article)
32	7/68	Let's Reprice Obsolescent Equipment (Editorial)
33	1/69	Obsolescence (Panel Discussion)
34	1/69	Obsolescence - People Plateaus (Article)
35	10/69	I Object (Article)
36	3/70	Objection Sustained (Forum)
37	4/70	Slowing Pains (Co-authored Editorial)
38	5/70	Voting Systems (Co-authored Article)
39	10/70	Flavors of Compatibility (Editorial)
40	5/71	Datamation Industry Directory (Products Catalog)
41	1/72	Vocabulary for Information Processing (Review)

42	3/72	Letters re Vocabulary for Info Processing (Letter Exchange)
43	1/73	Computer Dictionary and Handbook (Review)
44	1/74	Privacy, People, and Credit Services (Article)
45	4/74	Eight Security Books (Review)
46	5/74	Communications Network (Review)
47	9/74	Proposed Law Threatens DP Users (Editorial)
48	3/75	Keystone Documentation Package (Review)
49	10/75	Structured Programming Textbook (Review)
50	12/75	You've Come a Long Way, Baby (Forum)
51	1/76	They Lit a Candle (Article)
52	2/76	Computer Audit Guidelines (Review)
53	5/76	Dispersing Responsibility (Article)
54	12/76	Software Engineering and Life Cycle Planning (Report)
55	4/77	Humanized Input and Software Metrics (Review)
56	10/77	Accessing Individual Records from Personal Data Files Using Non-Unique Identifiers (Review)
57	12/77	Sixty Ingredients for Better Systems (Article)
58	8/78	The Other Side of the Small Computer Picture (Article)
59	11/78	Auditing and DP: Redressing the Relationship (Article)
60	1/79	Things are Looking Up for Software (Report)
61	11/79	SRA Data Processing Glossary (Review)
62	12/79	Productivity Gap (Report)
63	1/80	A Checklist for System Design (Article)
64	9/80	Probing Productivity (Article)
65	10/80	Systems Analysis: Key to the Future (Townsend Article)
66	5/83	A PBX Cookbook (Article)
67	5/83	The BCS Review Manual of PBXs (Review)
68	7/83	Encyclopedia of Computer Science and Engineering (Review)
69	8/83	CBX Comparisons (Article by RLP et al)
70	5/84	The Seed of Empire (Article)

Oral History Citations

1. General Motors/North American Monitor for the IBM 704 Computer
   Pioneer Day Session
   Proceedings: 1987 National Computer Conference - Chicago
   June 1967

   Also available from Rand Corp, Santa Monica, CA
   as P-7316 (19 pages)

2. Oral History: Robert L. Patrick
   Robina Mapstone, Interviewer
   75 pages (typewritten, includes index)
   Archives Center
   National Museum of American History
   Smithsonian Institution
   Oral History Collection
   2/26/73

3. Security
   System Review Manual
   AFIPS Press
   October 1974

4. The Computerized Society
   In Time-Life Series: Understanding Computers
   Chapter 1: Taming the Mainframe
   Pages 14-18
   1987

5. Anecdote: Early Data Reduction on the IBM Card Programmed Calculator
   IEEE Annals of the History of Computing
   Jan-Mar 2005
   Pages 78-81, including photographs

6. Anecdote: Getting Started in Consulting
   IEEE Annals of the History of Computing
   Jan-Mar 2006
   Pages 93-95

Consulting Adventures: An Autobiography by a Computer Specialist,

By: Robert L. Patrick

March 2005

Author Blurb

This autobiography contains highlights from the 40-year work life of computer specialist Robert L. Patrick. Bob did some programming, some operations, some management, some planning, and a lot of consulting. Much of it involved some aspect of computers and data processing. However, if there is a nerd among Bob's readers, he will find Bob dabbled (successfully) in many other venues that did not specifically involve electronics.

Many people know a lot about some one facet of his busy life. This treatise is an attempt to show that it was possible to balance family, civic service, personal finances, and technical endeavors over a working lifetime. Some of the lessons may interest current practitioners since the principles learned still apply.

Prologue

Briefly, I was the youngest of three children, an average student (unless challenged), received a degree in mechanical engineering (University of Nevada-Reno), married a divorcee with two boys, was employed by several corporations, and was an independent freelance consultant (I called myself a computer specialist to avoid all those consultant jokes) for 33 years. During that time I gave/wrote over 200 talks/papers, published a book, earned and sold a simple patent, did billable work in 22 states and 5 European countries, formed a corporation and helped establish another, designed a house, developed a ranch, and flew myself almost 2,000 hours (mostly in commuting to and from consulting assignments).

Naturally, I did not accomplish all of this alone. I worked with geniuses and dullards and learned something from all of them. I could not have accomplished half of this without my wife, Corinne. She had secretarial, accounting, and computer skills of her own. When given the opportunity she discovered some financial and investment acumen as well. We were a team: I earned it and she spent it. When she said I could retire without our standard of living changing, I did. I was sixty-two.

The sections that follow relate my early years, my corporate experience, and my consulting adventures. I also include some of the important lessons I learned in 30+ years of consulting.

During my working years I felt a day without learning was a wasted day. I maintained four distinct types of customer accounts -

* Research, to add to my knowledge base.
* Commercial, to practice today's state of the art.
* Military, so I could try things at the fringes of the art.
* Computer manufacturing, to feedback experience from the field into the next generation.

For thirty years I maintained this client mix. If I lost an account I would actively seek another of the same type. Thus, I was always learning something new, applying what I already knew, and reducing ideas to practice.

Four things stand out in my professional life above all others:

- I had the pleasure of working with Bill Sharpe at RAND and when he published his book on computer economics, he gave me some credit. Later Bill won a Nobel Prize.
- Also, I worked with Fred Brooks during the development of the IBM 360 series. When Fred published his book on the management of software I got a mention.
- In February 1973, I was interviewed by Robina Mapstone of the Smithsonian Institution for their Oral History Project.
- In February 2006, I was interviewed by Tom Haigh of the Computer History Museum for their Oral History Project.

R.L.P.

May 2006

Table of Contents

Table of Contents

(March 2005)

Consulting Adventures of Robert L. Patrick

A note to the ladies - Much of the following is written using male personal pronouns. This is merely to make the prose less cumbersome and easier to read. No female slight is intended. The female programmers I worked with are ranked with the best and many of the female engineers I worked with were excellent.

Early Years

In a long and varied career luck played a part in my getting good assignments. However, I was also frequently in the right place at the right time with the right experience and the right preparation. If you know me well, skip this section.

1. I was born in Dallas TX, October 21, 1929 the third of three children.

2. When I was about five, my grandfather gave me a screwdriver and an old mantle clock. I took it apart.

3. When I was nine I was allowed to take the street car from our residence on the east side of Dallas across town and then catch a bus to Grand Prairie where my married sister Frances met me. Her husband was night mechanic for a flying school.

4. In 1939 my folks moved to Reno NV. WWII came and I went to public school. After the war my sister and her mechanic husband ended up in Reno to rebuild a damaged stagger-wing Beech (craft) which had flipped over while landing on some ice. My brother-in-law was an artist in wood and the Beech had a lot of wood that was damaged in the accident. I was about 14 and in love with airplanes. I joined the project after school as official churn. I cleaned parts, held wrenches, handed tools to those in need, and was free low-grade labor.

5. In high school I played trombone, worked on cars, got fair grades, and was bored to tears. After my junior year I had enough credits to graduate if I took two courses in summer school. I did, and went to college while still 16. They mailed my diploma to me.

6. My father had been an officer in France in WWI. It was the best time of his life. He wanted me to make the Army my career. Ted Carville, former Governor of Nevada and then U. S. Senator, lived across the street from us. At Dad's urging I asked Senator Carville for an appointment to West Point and got it. The fellow ahead of me on the list failed and I passed. I went to West Point in July 1947 and lasted 29 days. I had been through Hell Week to get into my college fraternity. Now the Army thought I was going to put up with a whole summer of the same crap. I suppose it was designed to weed out the cadets who were psychologically weak, and the cadets who were physically subpar. However, they also weeded out some who had different plans for their lives. I got back to Reno just in time to register for my sophomore year in engineering.

7. In addition to an engineering degree, I collected an Air Force R.O.T.C. commission from my 4+ years in college. I was a senior with good grades during the Korean War. Government recruiters made a deal with the university administration that any seniors in good standing would receive their degrees if called to active duty after the mid-term exams of their final semester. A week after our mid-terms all of the Reserve and R.O.T.C. officers got orders to report. I did, and they mailed another diploma to me.

Corporate Life

8. The Air Force put all new 2nd Lieutenants through a series of intelligence and placement tests before assignment. I stayed sober and tried real hard. They thought I was smart. My placement officer said I scored in the top 2% of those tested. I did not know what all this meant, but the most desired assignment was the Air Research and Development Command at Wright Field in Ohio. I was sent there, and later to the Air Force Flight Test Center at Edwards Air Force Base, California. While others went to war, I spent my entire service tour doing engineering work for the USAF.

9. I got into computing via a short course at UCLA. The instructor was Ev Yowell, a true pioneer who ran the SWAC project. IBM had married a calculator and a tabulator and called it a Card Programmed Calculator (CPC). Typical of the time they provided no programming aids to make the new machine useful. Ev, and his assistants at UCLA, devised a set of plug boards to control the machine's functions and provide an interface for engineering work.

Today, software provides this interface function. In l952 it was done with wires and plugboards. Ev's plugboards allowed the user to evaluate formulas one mathematical step at a time (programming) and then record each step in a punched card so the machine could be instructed what to do.

The wiring and testing of the plugboards was painstaking and complex. Once wired the card programming was natural and straight forward. Senior engineers had been devising stepwise calculations for years, converting them to spreadsheets, and having humans at desk calculators carefully follow the procedure over and over again for different sets of input data.

With the CPC, each step on the spreadsheet was converted to a punched card and the computer (with a competent operator who was not mathematically trained) evaluated the set of equations for each input case provided.

It worked well for small sets of equations, but the machines were not very reliable, and the handling of large decks of punched cards was error prone. With care, and some test cases whose answers were precalculated, calculation procedures in the range of 2,000 steps could be accomplished. The time was 1952.

In the history of things, the Yowell CPC boards were the ancestor of Speedcode for the IBM 701 and roots of the FORTRAN compiler at the beginning of the world of engineering computing.

10. I got out of the Air Force in 1953 and went to work for Convair in Ft. Worth. As an aircraft manufacturer, Convair had fallen behind the state of the art in computing. They were trying to catch up. Since I was current in computing techniques my experience was valuable. Shortly after being hired I was doing big wing load/deformation calculations using punched card techniques. In one case I used a calculating punch to prepare the program cards for a 1,000 by 1,000 quasi-diagonal matrix which then determined the deformation of a proposed wing structure under air loads.

    Convair then firmed up an order for IBM 701 Serial No 7. I took programming instruction in NYC and helped with the installation in Ft. Worth. By the time the 701 got to Ft. Worth, John Backus of IBM had devised a software system called Speedcode.

Speedcode was an interpretive system which provided features similar to the CPC plugboards devised by the Yowell team at UCLA. My experience programming the CPC was directly applicable and I was instantly productive in the new computing system.

That worked out very well as Convair was in the throes of designing the supersonic B-58 Hustler. I did the weight and balance calculations for it.

11. In 1954 Convair had a quantity of engineering applications which were programmed and tested. The customers in the engineering department needed to run case data through these completed programs to support the continuing design of the B-58.

    In those primitive days programmers were an independent lot, with no formal schooling in programming or computers. All of us had at least one degree, usually mathematics or engineering, plus on the job training. The only common trait was an aptitude for the detail work of programming.

    Once a job was tested the programmer was quickly given another assignment and lost interest (and skill) in his past work. However, the original programmer was still saddled with setting up the case data for his engineering customers and running it on the machine. This was called programmer-present operation as the original programmer worked the computer console, hung the tapes, and supervised the printing of production work.

    Not only was this a significant distraction to the programmer developing new code for another job, but quite frankly, many were not very good at production operations. They would forget something so the machine would sit idle while they ran after the program/data/run book they forgot; or a machine error in the middle of a long run befuddled them; or they were all thumbs hanging tape on a manual drive; etc. I'm sure you get the picture; their programming aptitude got them hired and they weren't very good operators.

    In the Air Force I had faced this problem before and knew how to write instructions so someone other than me could successfully operate using the programs I developed. I applied this experience to the Convair situation and documented my jobs so I did not get (many) urgent phone calls.

Soon my documentation standards became the norm and a bright person with good finger dexterity was hired to be our first production operator. The time was 1954, and the woman hired to run first shift production eventually became my wife. She may have been the first production computer operator.

Computer time was at a premium in those days. The criterion was not the high cost, but raw machine availability. I set up a production second shift for overnight production runs. "Check the data before you go home and we will have the output ready in the morning."

12. Eventually, as the programming staff grew, machine time got very tight. The jobs taxed the reliability of the machines which only ran one job at a time. Further, if you were processing input or printing output the machine was just as tied up as if you were doing heavy calculations. The limitation was in the single sequencing hardware.

    Speedcode supported magnetic tape, but they were not popular with the programming staff. I had read an industrial engineering article on time and motion studies (Later published in Gantt on Management, A. W. Rathe, AMA, 1961) and found I could do useful debugging work while another programmer was just getting organized at the console. By storing programs and data on tape I could load my job in a few seconds, process a few cards through the card reader containing data or programming patches, and get the printed output from a test shot without delaying the incoming programmer or impacting the formal schedule.

    This tape to tape operation was very efficient for the configuration then in use (small primary memory, equivalent sized drum storage for swapping, and fairly fast magnetic tapes). This experience proved some ideas I had for production computer operations.

13. In my early days I was impatient, rambunctious, and hard to manage. Further I had been working on things related to the military for many years (West Point, R.O.T.C., Air Force tour, the design of military aircraft). After about 18 months at Convair, I went to work on cars at the General Motors Research Staff (GMR) in a new research center just north of Detroit.

GMR had a Card Programmed Calculator and was just getting the last IBM 701 manufactured (Serial No. 17 as I recollect). I was productive the day I went to work. At that time in history many installation managers thought they had a unique problem set that could not be addressed with available software. This led inevitably to a software development effort. A hotshot from Allison Division undertook (with my management's permission) the development of a competitor to Speedcode on GM's machine. They dumped time, talent, and machine time into it. It failed. Meanwhile I was providing engineering support with Speedcode and the CPC (while it remained).

With my flight test experience it was natural for me to provide the data reduction support for GM's automotive gas turbine development effort. As a new-hire I was delighted to use computer systems I knew for an engineering application I was familiar with, and start building a solid reputation.

14. Sometime in 1954 IBM announced the Model 704 to replace the 701. To the customer the 704 offered more reliability, more primary storage, faster circuits, and a pair of offline machines which would read cards and write magnetic tape, and read tape and print - - both completely independent of the mainframe. Suddenly I was center stage as I had been simulating that style of operation with my Speedcode sneak-in tape operation for over a year.

    SHARE, the IBM scientific users group, had been formed in Los Angeles and at their second meeting I proposed a new operating system for the 704. Owen Mock of North American had some similar ideas. A project was launched, the work was divided, and in a short time we had an operating system for the 704. (Really two versions were produced because there was insufficient time to fully negotiate on one set of features. The code in the two versions was 95% identical.)

    As we went into operation at GMR with what was called the General Motors I-O system, programmers built test or production decks at their desks, every hour a mail person would pick up the decks from the programmer's desk, and then carry them to the card-to-tape machine.

    As the decks arrived and before the cards were read by the machine, the human operator sequenced the batch of programs on external priority. The decks then went

card to tape in a format called Binary Coded Decimal. The tape was then handed to the mainframe operator in the next room.

When the mainframe was available (it never was free) the tape was hung and the batch was run. There were no run books. The process was non-stop: An input phase converted all the input tape to binary and wrote a binary version of the BCD tape produced by the card to tape machine.

The binary tape was then rewound and the jobs loaded and executed in sequence. If a job hung up or the system crashed, a standard button sequence dumped the offending job and started the next one from the input tape. As a job produced output it was written in binary onto an output tape by the originating program. When the input tape was exhausted, the compute phase was over and the output tape was rewound.

Another conversion program was automatically loaded and it converted the binary output tape to BCD tape which was then handed to a Tape to Print operator in the next room. Printed output was physically matched with the original card input and both were returned by messenger to the programmer's desk.

If all went well, two non-professional operators ran batch upon batch without working too hard. The programmers got desk to desk service, and the CPU ran non-stop as long as there was work to do. Test and production work ran intermixed. Trouble was handled by standard procedures. The time was 1955.

Internally the system was efficient because the system programs for input and output translation were much larger than the size of the usual job. Thus the translator programs were loaded from drum memory only once per batch and processed a whole tape before relinquishing control.

As in earlier experiments, standard input card formats, standard deck setups, and standard calling sequences for printed output provided a way for efficient programmer-absent operations by non-professional operators who knew the system but not the job stream. At last count some variant of the NAA/GMR operating software was used in 43 installations.

In one case the program I was developing was in the later stages of test. I made up multiple sets of card input and ran ten tests as a single batch. Even though one or two failed and had to be dumped by the operator, the next case in sequence got a fresh setup and could run as the program dictated.

15. The next IBM machine in the scientific sequence was the IBM 709. In addition to a bigger primary memory and more reliability, the 709 offered I-O channels. A channel, despite the IBM parlance of the day, was nothing but a special purpose computer which shared the memory bus with the mainframe. On the other side of the channel controller one attached all of the electromechanical input/output devices: Tapes initially and later readers, printers, and punches. Still later additional devices could be attached such as a very large disk drive, and eventually through a channel-to-channel connection, another computer.)

    In the days of the 704, IBM offered software as first among equals. The user community did their own thing [led by the user groups SHARE and GUIDE (for the data processing community)]. Sometime an installation used IBM software, sometime software developed by the user community, and sometime home-grown code. While there was some standardization within an installation, there was less software standardization between installations serving different corporations. IBM saw this chaos and decided to support, as a manufacturer, software development across a machine type serving a variety of similar users. Thus the IBM-SHARE software development was born. It was a committee made up of people who had successfully done things and a horde of IBMers who wanted to pick our brains. Owen Mock of NAA and I were both members. The effort produced the SOS operating system, a FORTRAN compiler, and eventually a whole bag of other programs.

    However, before the first committee meeting I got another assignment. The cold war was pretty hot and the U.S. Government wanted badly to maintain missile parity with the USSR. The 107 Weapon System was a 5,000 mile intercontinental ballistic missile (code name Atlas) which had been under development for several years. The Government wanted a shorter range, smaller, more mobile missile system. When the critical components appeared to be within the state of the art they established a similar parallel effort to produce a 1500 mile range missile. The program was designated WS 315A and the code name was Thor.

Overnight I resigned from the SOS committee and was assigned to lead the GMR Thor support effort. At that time the only big scientific computer in the entire General Motors Corporation was the GMR machine in Detroit. North American had successfully developed a liquid fueled engine, Douglas was to build the airframe, and the AC Spark Plug Division (ACSP) of GM was to do the guidance.

ACSP had been manufacturing precision gyros and looking over the shoulders of the guidance experts at MIT's Draper Laboratory. With the establishment of the Thor program, ACSP was to go it alone. My team and I were to support the ACSP effort.

I'll never forget the split off meeting: the participants consisted of USAF Generals, GM corporate VPs, Ph.D. guidance experts from MIT, and me. I was a mouse in the corner and did not speak until spoken to. I was 26 years old.

For the next two years my team was locked in a high security basement room with a guard stationed at the door, and had priority on all the machine time we wanted. Dick Sullivan was the mathematician who developed the differential equations we programmed. Eventually the Thor was flight tested and went on station in Europe with the launch codes we developed.

16. After having my horizons broadened I was reluctant to go back to my old job as programmer-analyst. Fortunately, about this time, GM corporate management was building a team of in-house consultants. I was the computer specialist on this team. We flew all over the corporation on consulting assignments, were expected to keep current in our specialties, and did personal research when not on field assignment.

    I was part of the Allison team proposing on the Minuteman Assembly and Test project. We didn't win the job, but I saw a lot of Indianapolis. One rush meeting had a chauffeured limousine pick up a suitcase my wife had hastily packed, and then the limo picked me up at work and hauled me to a corporate DC-3 waiting at Detroit Metro airport. I then flew off to Milwaukee, the only passenger.

17. An unsolicited job offer got me thinking about opportunities outside of GM. I had created a project I wanted to work on, but it was assigned to another GM Staff activity. Since a transfer wasn't forthcoming I started looking around. I ended up as Deputy Director of

the largest (at that time) commercial service bureau in the country (and world), C-E-I-R in Alexandria, VA. At C-E-I-R I had about 100 programmers and a 709 to look after. I learned to manage through a layer of intermediate managers and did less technical work than I wished. My boss was buried in some linear programming applications so I learned bid, proposal, sales, and administration. During quiet times I could do a little technical work. I had long held a string of hard-to-get military clearances. Some of this work was really interesting.

One project dealt with a manual for military targeting. It was classified Top Secret. The time was 1958, right in the middle of the cold war. Then, as now, highly classified work was compartmentalized so that no one person below top rank saw the whole project in all of its splendor. We did nuclear fallout calculations directly for the Pentagon. While asking where the current map data came from was out of the question, I wondered how we knew accurate latitudes and longitudes for all those places in the USSR.

18. Fletcher Jones, Roy Nutt, and I had met several times at SHARE meetings and we had some common interests. Fletch had just led a successful project that produced SURGE, a fixed format report generator for commercial work; Roy was well known for SAP, his symbolic (machine language) assembly program for the 704; and as mentioned earlier I had most of an operating system to my credit.

    We were young (about 30), liked to do interesting work, and saw a golden opportunity in contract software. Fletch was a good salesman, so when he said he had a startup contract in his pocket, Roy and I were ready to go.

    Fletch had the corporate articles of incorporation drawn up, all three of us signed them, and I took some GM stock to the bank to finance the venture. Fletch had 66% of the stock and Roy and I each had 17%. All three of us were to be directors of Computer Sciences, the name chosen for our little corporation. The customer was Dick Clippinger of Honeywell. The job was to produce a "natural language" commercial compiler for the Honeywell 800 series business computer. The compiler was to be known as FACT, Fully Automatic Compiling Technique, a name chosen by Clippinger.

    We parceled out the work as follows: Sales and Admin - Fletch; Project Architect (although the term was not in use then) - Roy; and I was to be Project Manager. We

worked a few months in Boston where the Datamatic Division of Honeywell was located and then moved to Los Angeles where we wanted to locate our company.

19. In LA our troubles started almost immediately. Fletch was good looking, and liked to run with some acquaintances he had in the Hollywood crowd. I did some administration for him. Roy was a loner. He would show up for work about the time those of us with families left for lunch and then work late into the night after everyone else had gone. As project manager I plodded along wondering what I had gotten into. I even carried a little extra money in case Roy forgot his wallet when he came to work.

    One day we found out the payroll deductions were being taken out of the lockbox and used by Fletch (he borrowed them intending to pay them back before they were due each quarter to the Feds). Roy and I were aghast since all directors were equally responsible for such a transgression in the eyes of the IRS. We dissolved the corporation.

    Clippinger was furious. He had a new machine coming out and the possibility of no software for it. He said the contract would go to any two of us who could get back together. Roy and I drew up articles of incorporation for a new venture to replace CSC. Fletch buttonholed Roy on a plane back to Boston, and suddenly I was the odd man out.

    According to the original articles of incorporation they had to repay the money I had loaned CSC, and I had to return my stock to the treasury for book value (not much so early in a startup).

    Ever since, official CSC documents have ignored my part in the beginnings of the corporation and I have become the "Mysterious Third Founder".

20. While I was still with CSC I got involved with the design of the FACT input language. During a search of the natural language literature, I discovered Basic English by C.K. Ogden.

    Prior to WWII the intellectual community debated whether a common language for world commerce should be Spanish based or English based. The offerings were Esperanto

and Basic English. The debaters became mute when English took over due to the WWII efforts by the USA.

The book I checked out of the library had a vocabulary of only 500 words and said this was sufficient for commercial intercourse. It even contained a copy of Lincoln's Gettysburg address in the proposed English subset.

During these same weeks Grace Hopper used her Government contacts to form a committee to try to draw up a standard language for stating solutions to commercial data processing problems. IBM was already the dominant computer supplier in those days, and Hopper, who worked for Univac, saw a standard commercial language as a way that the other manufacturers could assault Fortress IBM. Roy was on the Hopper committee to make sure Honeywell's interests were considered.

Meanwhile CSC had a deliverable to make to Clip: The compiler's language specification. The CSC team prepared the spec, and I wrote much of the manual. I drew heavily on Basic English since the linguists of the 1930s had worked pretty hard putting together a coherent set of words whose meanings were unique when properly used.

I peppered the spec with examples. Clip bought the spec, Roy carried it to Grace's committee, and they adopted some sections in toto for COBOL. Out of curiosity I looked at a COBOL language manual some 20 years after COBOL became a national standard, and the examples I wrote for the FACT compiler language spec were still intact in COBOL.

21. Needless-to-say I was soured on corporate life. With the support of my wife I decided to be an independent freelance computer consultant. My first client was Dick Clippinger of Honeywell. My assignment was to do a series of test cases for the compiler CSC was building.

    When I started consulting, I had a rare combination of experience. I think I was the first individual to go solo. There was no school, book, or advisor. I made all my mistakes myself.

Consulting Adventures

22. RAND wanted to hire me. I wanted to consult. So they put me on as a consultant for one day per week until I came to my senses. This arrangement lasted for 32 years. I went through several changes of management and a wild variety of assignments. On several occasions the Federal Government wanted my services and RAND management wanted to earn brownie points, so RAND paid me and loaned me to the Federal agency free of charge, travel expenses included.

    During Lyndon Johnson's term I worked for the Executive Office of the President and helped formulate Federal Computer Procurement Policy.

23. But mostly I consulted on technical assignments. I had experience with missiles and operating systems so when a friend recommended me for a job at the Aerospace Corporation, I got it. Aerospace had a big computing center which was constantly overloaded. They had IBM 7090 equipment, which was programmed like a 709 but it was faster, much more reliable, and with a wider variety of I/O devices available.)

    I worked there for a few years (one day per week) on projects to instrument and micro-measure the Input/Output (I/O) load which characterized their workload, devise a new machine configuration, design a major modification of the IBSYS known as the Direct Couple, install all of this in a large machine room, and tune the configuration to their engineering load.

    Two things about a Direct Couple were interesting. First it never would have existed if IBM had not goofed on the pricing of the IBM 7040. The channel boxes on the 7090 were more expensive than the whole 7040 computer. The 7040 could service all I/O devices of interest in a large scientific installation, could communicate channel-to-channel with the 7090, was compatibly programmed, and was cheaper than a couple of channel boxes. Once the users discovered this, IBM got a lot of channel boxes back for junk, built a lot of new 7040s, and made less money.

    The other interesting thing about a 7040-7090 combination was the internal scheduling we built into the 7040 software. Jobs were entered into the 7040, either remotely or online, and stored on disk. A scheduler looked at the external priority, the current

workload, and the characteristics of the job being scheduled. A scheduling algorithm on the 7040 called for tapes and when the tapes were mounted passed the job to the 7090 for execution. As output was produced by the 7090 it was passed to the 7040 and stored on a disk. When a printer was available the output was converted and sent to a printer by the 7040. All the while the 7090 was churning away on some heavy calculations. All of this is reminiscent of early three-phase operating systems except no human operators were involved in the scheduling.

However, if printers were available, and no output had accumulated, the scheduling algorithm ignored external priority and used job characteristics to send jobs with more output and less compute to the 7090 so the printer capacity would not go to waste. Thus it was not unusual for jobs to exit the system in a slightly different order than their priority would dictate.

Other options in the system provided direct simulated online device support if your external priority was blistering high (as if supporting a missile launch). A different algorithm entirely was used for overnight service which attempted to optimally exploit all available resources since the customers would not pick up their output until the next morning. This was all an early form of adaptive scheduling.

On top of all this, the machine room had conveyor belts behind each printer so the operators were not tempted to queue completed output behind a printer instead of walking over and placing the output in the appropriate bin for customer pickup.

24. After going freelance I tried advertising to get new business, and tried formal bid and proposal efforts (singly and in combination with others). Neither worked.

    Then I started writing articles for Datamation magazine. That worked. After I built a warm relationship with both Datamation management and staff, I wrote four articles per year and they published most of them. It was mutually beneficial. I could write, wrote of current events, and they developed techniques to handle my stuff. On their side I could hurry up or slow down depending on when they needed copy. They got comfortable with me.

I got some consulting assignments from the magazine and from their parent corporation. Once when the editor was recovering from a car accident, I was acting editor and put out three issues.

25. While I was helping out at Aerospace I published an article describing a hypothetical computer with microprogrammed internal architecture. Shortly thereafter, Gene Amdahl, one of IBM's premier engineering employees, invited me to sit down and talk. Soon I got my first contract with IBM.

    It seems the Datamation article about microprogrammed machines was close enough to a machine IBM had on the drawing board that they wanted me under contract. Suddenly I found myself a member of the team that produced the IBM 360 series computers. The work was done in Poughkeepsie, NY by a cast of thousands. At the center of the hardware universe was a very talented trio: Gene Amdahl, Jerry Blaauw, and Fred Brooks. I was very pleased to be associated with this effort and got my assignments from Brooks. I think I was the only consultant on board, and the only outsider on the team.

    My first assignment was a prepublication edit of the machine reference manual which was the compatibility spec for the five machines in the soon to be announced set: Models 30, 40, 50, 65, and 75. After such a monumental editing job, I knew the machine language pretty well.

    Prior to the announcement (April 1964), the Federal Systems Division (FSD) of IBM was bidding on an FAA air traffic control automation contract. The crux of the bid was the time required for the Model 50 to execute a frequently occurring kernel of code. When programmed by the FSD crew it needed three Mod 50s to do the required work. They didn't think that would get them the contract.

    Fred Brooks collared both Amdahl and me and asked us to program the kernel. We spent a couple of days at it and the work fit comfortably in two Mod 50s. Gene did all the heavy lifting and I carried his briefcase. FSD got the contract. Then Gene and I went back to getting the 360s ready for announcement. This was the only time I programmed in machine language for any 360 machine. Some years later I met a VP from FSD and he said that the two days Gene and I spent programming meant $50M in business.

26. After the 360 hardware architecture was well in hand and the engineering teams for the five machines were forging ahead, the architecture team broke up and Brooks went to North Carolina. When IBM management found out the mess the software was in they called Brooks back. Fred kept me to work as a troubleshooter. We were late and could not make the time up. Early manufacturing and hardware deliveries had to be scheduled around software availability. Much tuning was required by early users. When North American installed a 360 Mod 40-65 to replace a highly tuned 7044-7094 (an upgrade from a 7040-7090 Direct Couple), the 360 configuration could not do a day's work in 24 hours . . . very embarrassing. Eventually it all got worked out and the System 360 project went on to be the most successful computer development ever (by several measures).

27. The American Chemical Society had a commission from Congress to gather, digest, and publish the authoritative tome for the chemists of the US (world). The tome was called Chemical Abstracts. The work was done in Columbus, OH. I had a friend from GM who was technical director.

    IBM produced a device that could electronically hold and project varying type fonts electronically. These were mislabeled photocomposers. Chem Abstracts wanted to get out of hot metal typesetting and into photocomposing. For several years I helped with their photocomposing developments and later with their data base work.

    The chemist-editors had such high publishing standards that they could tell a normal period from an italic period in the same font.

28. I got into data base software at the Space Division of North American (builders of the Apollo space capsule). I had been working at Space on general computer consulting work for some time (they had 7044-7094s and replaced them with Model 40-65s). Uri Berman and Pete Nordyke came up with a piece of software they thought was better than sliced bread. The only problem with that claim was that nobody else could understand what it did. I was assigned to find out what it did and do a description of this breakthrough so Space management could decide whether to support it further.

I listened, drafted, discussed, and finally published. At the heart of the matter was some software (that they had previously written for the IBM 7010) which allowed the description of a file to be encoded and a general processing program to use this encoding to manipulate files of information (the 7010 online application was a list of parts for the Apollo capsule). This was a very important piece of work since Space faced the conversion of many file processing applications to the 360s they were installing.

Once the manual was written, the benefits were understood, and a joint IBM/Space Division project was launched to produce the required software for the 360. I was on the development team and did some of the architecture and some of the documentation. The result was IMS.

There were several innovations from this project. The online software ran as an application under OS360. This made us independent of OS maintenance. The software would support several simultaneous online applications, and if one failed the others continued to run (as long as the OS survived). The entire configuration was measured and tuned (both hardware and software configurations) for the workload expected. During the latter stages of system test one Mod65 simulated a load of terminals and drove another Mod65 running the test software. For transactions with the characteristics found at Space, 4000 terminals could be supported. And last, but not least, R.R. Brown (one of the managers at Space) coined the term "data base manager" and hired the first one.

IMS was the first commercial data base management system. It was a resounding business success for IBM and after many years of continuous development is still in use today. Berman got a cash award from IBM with a $50,000 check attached.

29. The biggest data base job I was involved in was at RAND. One of the first times Congress debated health insurance the Great Society was in full swing. There were several competing health insurance proposals. RAND was awarded a contract to evaluate cost and service levels to compare competing plans.

    The project was unusual for RAND since setting up field offices, enrolling people in the various plans, paying the costs, tracking the usage, and drawing conclusions were all

required. It was a multi-year $50M project. I was involved in helping design a custom data base for all the information gathered.

Tracking people relationships was one big data problem. Who was related to whom, just what was this relation, and how those relations changed over time was important to understanding the usage of health services. We designed data base software which linked individuals through a variety of couplings.

The other problem related to data quality. With a fact based data base, quality can be assured through intensive editing. (E.g. List the presidents of the United States and their inaugural dates.) If the data has a 100% duty cycle, the data base is self-cleansing as quality is assured whenever a cycle is complete and the individual involved has had an opportunity to approve or correct the data which are personally his. (E.g. Monthly bank statements.)

Neither of these is totally true for a file of observations. While the name and address of the person may be verified (especially if you send them regular checks), the hospital services utilized by that individual are somewhat suspicious if several hospitals and clinics are involved since they probably do not have uniform data capture and editing standards. (E.g. One set of submitted data said a woman just had her second hysterectomy.) Finally, the data may not exist on the day of capture. (E.g. Some day laborers just do not know how much they earned annually if they were paid in cash.)

During the RAND project we invented data quality indicators which accompanied each field in the file and conditioned its processing. Unfortunately personal computer data base software became popular before data quality indicators became common. Too bad. Many current health files could use such indicators. (E.g. has this data entry on drug interactions been approved by the FDA?)

30. Another RAND project had broad impact. Being an ex-Air Force officer I was aware of the critical importance of the Military Occupational Specialty (MOS) codes in classifying officer skills and making duty assignments. I also saw many officers with precious computer skills being assigned to jobs which could be done equally well by others while there were computer issues that needed attention.

Over a period of about a year I made a strong case for new MOS codes which would properly identify those officers with computer skills so they could be properly assigned, rotated, and promoted. After convincing the appropriate RANDites, we took on the Air Force. The matter eventually got to the three star level of the Air Staff and a computer MOS skill program was launched.

31. RAND was a hotbed of computer activities in those days principally due to Willis Ware and Keith Uncapher. We concluded that the USAF needed to make better computer procurement decisions. With the MOS skills properly identified, senior Air Force officers would eventually have proper experience and seasoning. However, a little education for incumbents would improve our decision making in the intervening years.

    RAND management convinced Curtis LeMay, then chairman of the Joint Chiefs, to establish the Department of Defense Computer Institute. While the official bureaucracy ground on, a group of about a dozen civilian lecturers were provided by several major Defense organizations: Bellcomm, DDR&E, DIA, IBM, Mitre, RAND, PRC, SecDef, SDC, Univac, and the USN volunteered to establish the Institute and present the first course. Our students were generals and admirals (you had to have stars to come), it lasted about a week, and the Navy was the host service. I was acting Director and lectured on applications. We cycled the course several times until the permanent staff came on board.

32. Uncapher and Ware were very active in the politics of computing. The American Federation of Information Processing Societies was their baby. At one time or another each of them was president of AFIPS. In those days AFIPS ran national computer conventions which consisted of technical programs and displays of manufacturers offerings. These were money making meetings and the surplus was used for good works.

    In the late 1960s and early 1970s I had been in and out of many computer centers. With my clearances, I visited many centers which held precious information. After much frustration about the unnecessary exposure in both the military and commercial worlds, Ware and I decided computer security had to be openly discussed. Ware kicked off the effort with an AFIPS technical paper in 1965 and I finished the first round of AFIPS efforts with the publication of the System Review Manual for Security in 1974.

33. During my long exposure at IBM I tried to get them to officially provide security features as part of their products. It was a long hard sell since their customers were not clamoring for security and embracing security meant adding dollars and time to development schedules.

    After several years of debate, IBM produced a secret list of OS360 security holes. After more debate, each development manager was tasked with closing the holes that were in his portion of the product and keeping them closed. IBM then allowed customers to submit security flaws through the APAR system and get them corrected just like function and performance flaws were.

    Later I was a junior architect on the IBM 8100 minicomputer system for small offices. In the specs were several security features that were developed, tested, and delivered just the same as all of the other functional features. The 8100 would have been a major security milestone if it had sold well.

    In passing I note that computer security cannot be added on later; it must be part of the bedrock architecture. Memory must be partitioned by hardware limit registers (like a 360) so that rogue programs cannot tamper with the system or enter space reserved for some other program, software must dynamically maintain these limit registers through normal operations and when errors are encountered. All software must be check summed and the sums must be verified each time a routine is loaded. Physical keys must be provided to lock console functions. Delivery of the basic software and the physical keys must be done by certified messenger. There's more, but needless to say all of these protections must be maintained at the vendor's plant and on the customer's premises for the life of the system in the field . . . hard and expensive to do.

34. Sandwiched in among my technical work I performed some civic services. I was living a good life and volunteer effort seemed to be part of fulfillment. The Data Processing Management Association had a program where individuals could sign up to be tested and if they passed they received a certificate. I was skeptical so I took the test, passed, and wrote a Datamation article about my experiences. DPMA invited me to be a member of the Certificate Council to help improve and administer the program.

The Association for Computing Machinery saw computers going to the boondocks where there was no depth of experience. So ACM launched a national lectureship program where selected individuals would travel around the country giving lectures at local ACM chapter meetings. I made a swing around the country as an ACM National Lecturer.

I lived across the street from a guy who was executive assistant to the then mayor of Los Angeles. He and I would have a drink now and then and I would berate him on LA's traffic problems. When there was an opening on the city Traffic Commission he said, "O.K. big mouth, come help." I was an LA Traffic Commissioner for five years (it took a half a day per week) and made some small progress while getting a Ph.D. education in the politics of big city public service.

35. However it wasn't all work and no play. While I worked for General Motors, both my wife and I were members of the GM (Corporate) Chorus. The 5,000,000th Chevrolet was a major occasion. The chorus was assembled (about 75 of us) on the massive hydraulic stage in the orchestra pit of the Fisher Theater in Detroit. A golden Chevy was driven out on the main stage and we rose up from the depths singing.

    While living in Virginia I sang in a barbershop chorus. I tried again while working for CSC in California, but the press of work was too much and I dropped out.

    Sometimes work and play blended very nicely. One of my customers was in the middle of a proposal effort and the computer team member took sick and went to the hospital. Could I come? The deadline was tight. "O.K., but I will have to bring my own secretary."

    So, on an expense account, my wife and I went to Seattle to work the weekend. We dropped our boys for a long planned Scout outing, flew to Seattle, did the required work plus a little extra, had two fine meals, and returned in time to pick up two tired scouts on Sunday afternoon.

36. In 33 years of free-lance consulting I had several hundred assignments. A cursory review of these assignments shows differences in clients, locales, facilities, personnel, hardware, software, and applications. But all this work fell into only nine classifications:

    - Confidant to a senior executive
    - Applications definition and design

- Programming methodologies and standards
- Machine operations and performance measures
- Conversions, between generations or steps up in service levels
- Project planning and management
- Site surveys and catastrophe planning
- Technical writing
- Troubleshooting

I shied away from staffing and recruiting assignments or anything involved in overt company politics. Occasionally I would get into direct project management (temporarily), as when the line manager on a critical project was injured or hospitalized; and a couple of times I was a legal advisor/expert witness.

37. When I worked for GM it was not unusual to encounter a lobby display of an experimental automobile, truck, or engine. So I was awash with curiosity when I saw a hall display of a big camera mounted on a cradle while being escorted to work on a classified job. Then I knew how the latitude and longitude coordinates originated that we used so many years earlier in the nuclear weapons fallout calculations.

38. One day I got a call from the assistant to an executive who was spending a lot of money on computers and staff. He said his boss did not feel he was getting his money's worth. He did not know what the problem was but the high cost was surely a symptom of something. From that call I launched a new consulting service: *The computer center (management) audit.*

    Thereafter whenever I received such a request a quick conversation determined whether I had the right skills and experience to audit the operation, that I had the right references/clearances to make the visit unescorted, that I could schedule a week onsite and several days shortly after for the report of findings, and that I was talking to the individual who could authorize my visit and guarantee facility access. If all these conditions were met I requested a package of background documents and scheduled a visit.

    These audit clients were usually new to me, and due to my independent nature and my frank non-political reports of findings, they were usually one-shot assignments. I did not

go out of my way to avoid follow-on work, but if the problem was management and you said that very clearly, you were seldom invited back.

I did 26 management audits in the US and Europe. I would appear, get my badge and introductions, work some of all three shifts, explore budgets, training, programming, installed software, hardware configurations, applications definition, user interfaces, trouble reports, and summary reporting for upper management.

Usually I'd find someone who had objected to my coming, and hence cooperated reluctantly. Usually I'd find the someone who had instigated my visit. Sometimes the good guy was the culprit, and sometimes the bad guy was the hero.

In one audit of a military installation the two star general welcomed me and offered his full cooperation (assuming he would not hear from me until I paid my respects on Friday afternoon). Right after I started to observe computer operations I found a massive security hole. It was so important that I was back in his office by mid-morning of the first day. After I explained the problem he fixed it. It was not yet Monday noon.

In another case, a VP said they had recently (six months) installed an online terminal system on a dedicated Model 65 and he had not seen a single improved schedule, any reduction in the programmer workforce, or an improvement in program quality. What was he getting for the $1M extra expense he was incurring?

The most frequent problem I found was in management objectives and measures. Senior management and computer management came from different eras, different schools, and spoke different languages. They did not communicate well. In these cases what management wanted and what was being delivered were fundamentally different things.

Of course, I often found incompetence of one sort or another. A slick accounting firm had done a consulting study that caused the creation of a development environment that overwhelmed the programming work force. In another case a salesman sold a bank a system its staff couldn't operate reliably. The top bankers were befuddled because the bank next door had a similar system and they were doing all right.

Many was the time I found management was unaware (or uninterested) in what was going on. They were satisfied if they shoveled money into the center and did not get any irate phone calls. At 3 AM one morning I was talking to a Navy Chief who was in charge of 3rd shift operations, when the division commander (a full Captain) came in to show the flag. After he left I asked the Chief how often the boss stopped by. "Never seen him before in my life" was the answer.

I found technical problems too. Every time the vendor would issue a batch of software changes the shop would be unstable and deadlines would be missed. Instead of learning how to apply fixes better, they held all the change tapes for a year and lived with the known software flaws.

The most frequent problem I encountered was in development schedules. Applications development is hard to estimate. Even if properly estimated, a management change on the customer side can obsolete many of the agreements. Sometimes programming management is not competent. People *really are* promoted just above their level of competence.

Many shops considered time cards unprofessional. Without time records correlated to past development progress, 100% of a schedule is uncertain. Most development jobs have a high percentage (say 85%) of the tasks which are similar to tasks done before. These should schedule and proceed quite well so management can devote technical and management talent to the other 15%. However without good timekeeping they all look the same.

Fred Brooks, in his book on the management of software, said big jobs got late "one day at a time". While that's very true, some big jobs are inadequately planned and may be seriously late on day one. When doing a management audit you never knew in advance what you were going to find. Each one was a challenge. There probably was something seriously wrong or I wouldn't have been called. My job was to find it and describe it so the problem could be fixed.

Only once did I find an unsolvable problem and that was in Sweden. There was a commercial batch service bureau whose inputs and outputs were transported by courier. The rest of the world was transitioning to online operations. The batch service bureau

was disappearing and they did not have the technical talent to make the changes they needed to make. Further, the Swedish social environment forbade them to make wholesale staff changes. Nice guys, but bound to crash. I'm glad they paid my invoice.

39. After about 20 years of assignments improving and hardening computer centers I observed that computer centers were stronger and more secure that the buildings housing them. I then turned to corporate disaster planning.

    I would start outside the corporation location and work my way into the computer center. Once I found a power pole on the outside of a curve on a fast road that fed all the power into a whole corporate campus. One inebriated driver could shut the entire corporation down.

    Sometimes I found a corporation with an emergency generator which may have been properly sized initially, but years of computer configuration changes and added communications facilities would have overloaded it during an emergency.

    One building had a beautiful architecture and was situated on a hill with a nice view, but a single car parked at the wrong place at the curb in a heavy rain storm could have flooded the underground parking garage. The garage housed the air conditioning compressors, the emergency generators, and the sump pumps which were sized for seepage and normal rain. A little extra water would wipe out the whole installation.

    As I was trying to make corporations more resilient, the manufacturing community was embracing "just in time scheduling." JIT is a Japanese scheduling concept which has suppliers and assembly plants scheduled together to meet external product demand with almost minimum investment in intermediate stockpiles of parts. In JIT scheduling an outside order for manufactured goods is broken down by the primary assembler and parts are ordered from suppliers, trucks/rail cars are reserved, and all of the parts move towards the assembly point in synchronism like beads on strings. With computers and communications this is very attractive and very vulnerable. Picture the assembly plan for an automobile as an information network. The plan for an engine is another network. The transmission is another, etc.

When assemblies are built for inventory, each network is buffered from the adjacent network by the inventory. When the inventory is reduced to zero the information systems are linked and the total network gets quite large very fast.

A labor dispute in a plant building starters can stop the assembly of cars. So can a snow storm, a flood, or an earthquake. In California, earthquakes are the worst hazard. They can't be predicted, give no warning, and tear up the geography in unpredictable ways.

During the ten years before I retired, I did a lot of earthquake recovery planning, lectured on the hazards, wrote articles, made a video, appealed (unsuccessfully) to the insurance industry to make the hazards better known, and even did some volunteer work for the CA State Seismic Safety Commission.

Emergency services are administrated by several entrenched bureaucracies. All are hard for an outsider to penetrate. I made some progress. While those company employees I lectured to prior to the Northridge earthquake of 1994 complimented me on my insight, a big improvement in EQ readiness will depend on someone who follows me.

40. Another branch of my consulting efforts related to the analysis and improvement of large information systems containing one or more embedded computers. A typical large system might be one which produced illustrated parts manuals for commercial aircraft. The manual to maintain a Boeing 747 in the field runs about 1500 pages. A pair of facing pages consists of perspective illustrations of a main subassembly on one page and a hierarchical parts list on the facing page. In use, the mechanic finds a failure, refers to the picture to see how to disassemble the unit, takes it apart, locates the bad component, and uses the parts list to order the next higher assembly in the hierarchy.

    Computers are used to build and check the parts list, create the illustration, and publish the book/microfilm so mechanics in out-of-the-way locales can get the plane back in the air. Whenever an engineering change is made in a part, the whole system is cycled to issue the corrected pages. A staff of 2000 people worked part time to produce and maintain the service manual for each aircraft type. Various aircraft types from a single manufacturer have both common and unique parts. Thus various manuals have pages identical except for the page number.

RAND researchers devised a highly automated system to support analysis of such man-machine systems. It was good, but due to the level of automation, professional analysts could understand the product, but customer executives could not. Planning Research Corporation tried to use it to analyze the paperwork flow within a military headquarters, but could not get their recommended changes approved.

I de-automated the technique and went pictorial. In the case of the Boeing 747 manual, the process flow chart was 60 feet long. It was a bit unwieldy but readable by all levels of management. I would gather a team familiar with the system being analyzed and teach them how to make these big charts. After the flow was recorded and checked, we would annotate each box with the effort (man-hours) needed to accomplish one cycle of the work. Computer processes were annotated with machine type and hours. Finally we would draw supervision boxes around sets of process steps so each line supervisor could check that his area of responsibility was correctly depicted and could see how he related to the rest of the process.

The charting was done in an open door environment and it was not unusual for a line supervisor to show up over lunch and say "I know you drew what I said, but we have made some changes and we don't do it that way any more." Bingo! The process was being improved as we were charting it. Needless to say we willingly made the chart changes.

After the chart was complete and approved by all the line supervisors, I spent a week with the team and we analyzed the entire system looking for redundancies and improvements. A big chart took a five person team three months to make. Usually the first round of changes paid that back and then some. Killing unnecessary computer processes (e.g. by saving a tape and avoiding a redundant sort) paid off handsomely. Chart analysis was focused, we could tune the process either to reduce flow time, or save money. Sometimes a quality improvement would also be found.

I used this analysis method on parts manuals, chemical reference publications, and a software change system for a major computer manufacturer. I wrote a manual on the charting and analysis process so I could apply it properly whenever an assignment came up. If I was too busy to take on a system analysis job, I sold the client a copy of the

manual, gave his team an introductory lecture, and wished them well. From the feedback I got, the teams could perform satisfactorily without me.

In a couple of cases, senior management wanted to understand what the process was without studying the detail chart. For these requests we produced summary charts which encapsulated the gist of the big chart on a larger scale. These provided the backdrop for discussions transferring responsibilities between divisions of a major corporation.

And into the Sunset

When my wife said I could retire without impacting our standard of living, I did. I was 62 years old and just eligible for Social Security. At the time we were living on a 236 acre alfalfa farm that we had developed in the high desert of Southern California.

In addition to an alfalfa field we leased out, the farm had a fully approved lighted airstrip, a hanger, an energy efficient house of our own design, and an experimental orchard with about a dozen different types of fruit and nut trees.

All of this was paid for as were both cars, a truck, a nicely equipped office, and my airplane. By carefully managing our expenses over the years, by carefully reading the IRS Bulletins, and by having enough money to invest prudently, we set aside a comfortable nest egg.

If you are consulting successfully, spend some time reading tax advice (from both the IRS and commercial publishers). This will keep you from doing anything dumb, and may increase the size of your nest egg as it did ours.

Robert's Reminiscences: A Semi-serious Look at a Curious Life

Robert L. Patrick, April 2002

- In Dallas, at age 7 or 9, I put a penny on the tracks and a streetcar squashed it.

- Randon Reid was a teenager who lived across the street. He was rebuilding a Cub-sized airplane. He taxied it on the street with the wings off when he got the engine running.

- Dad was going to demonstrate his bicycling skill on the hill in front of our house, fell off, and skinned his knee.

- Mother reached into some shrubbery to turn on a water hose. The neighbor's dog had gotten out and it bit her on her wrists. That's when Frances learned to drive. Sis was so short she looked through the spokes of the Pierce Arrow steering wheel.

- When we left Dallas on the move to Reno, the dog Rags was to be left to be ears for Grandma who was deaf. I smuggled him into the car, but he was discovered before too long and we had to go back.

- When we got to Reno I had a semester of grade school left. I was appointed to the Safety Patrol to help little kids across the street. Mother was proud.

- The grade school graduation celebration was at Bowers Mansion. We rode out on the Virginia and Truckee RR and hiked up across the meadow. Dad was on the committee and got a baron of beef from the Waldorf. We kids ate like kings.

- Jr. High was one problem after another. Bobby Landers had been held back several times and terrorized us younger kids. I had a new bicycle and did not handle the envy well. George Gadda was a shop teacher with an attitude. He made fun of the way Dad dressed, and was hard on me. The Worlds Series was a big deal and I don't like baseball. My trombone rivalry with Jack Bastian started and hounded me all through Jr. and Sr. high school. He was a better musician than I was, and took lessons from the school band director.

- Dad found an old hulk of a car and we towed it home. It was a Whippet. I dismantled it and we hauled the parts to the dump.

- Mom and Dad went to a Legion convention in Elko and I taught myself to drive Dad's pickup. I was 14. My second car was a derelict Model 66 Chrysler (1928?). I got it to run and drove a car full of kids to H.S. If one of us was tardy, the attendance officer at school would check the others in my car pool. Sure enough I had broken down again and we all were late. Once I had a racing Model T, but it was not reliable enough to drive to H.S.

- Good math students could go to the Math Club Room instead of study hall. The Club Room had windows which opened at ground level. If you scheduled study hall for last period and were good at math you could escape early. My car was always parked nearby.

- After my crowd got our drivers licenses we would race through the alleys at breakneck speed. One particular alley had a telephone pole and a guy wire on the outside of a corner. With more skill than brains we would corner wide and go between the pole and the wire. One night Wilton Herz (the jeweler's son) didn't do it correctly and wiped out his father's car.

- I was showing off one day and raced Dad's Chrysler up our long drive way at excessive speed. I miscalculated my skidding stop into our garage and moved the garage off its foundation with a big audience watching.

- Jimmy Atkinson was wilder than the rest of us. The judge took away his license, but gave him permission to drive home. He revved the engine, popped the clutch, and dropped the bottom out of his transmission as he pulled away from the Court.

- Our big house was a great party spot. Everybody came. I had lot of dates, but no girlfriend. I was a clod. Sometimes I got invited to teen parties. I never got beyond spin the bottle. I really didn't know what was going on.

- One year the H.S. Band marched in a Memorial Day parade, it snowed. The Band went on a trip to Carson City. We raced all 30 miles. My Chrysler won.

- Mechanically, I worked for a Chrysler dealer, a tire shop, an awning shop, a construction company, and helped rebuild an airplane. I got fired from my first job: dishwashing at Woolworths.

- I was a Boy Scout and later a Sea Scout. The Scouting program helped overcome my shyness. I was tall and slightly built. I didn't do much camping. I was a Life scout, but never made Eagle. I went on a Sea Scout trip to CA and the truck we were riding in flipped going over Donner Pass but stayed on the pavement. It had a stake body and we ended up on the pavement with the truck upside down supported by the stake rack and nobody injured. God was saving us for something.

- We lived in a two story house on the outskirts of Reno. It was built in the 1890s. There were verandas both up and down. I slept upstairs and my parents slept down. Sometimes in my early teens I would go to bed as told and then out to the upstairs porch and climb down a big locust tree and escape. I thought I was so smart, until my Dad confided in me that he did the same thing when he was a teen.

- Virginia Lake was a small man-made recreational lake south of Reno. Early one spring the Sea Scouts put a 16' sail boat on the water right after the ice broke. We rolled over in the cold lake, clung to the boat until rescued, and then the scoutmaster made us jog around the lake (about 1 1/2 miles) in wet clothes to warm up. That kept us from harm.

- I was too light for most sports. I skied a little, rode horses into the foothills (got bucked off once), and sledded down snow covered mountain roads.

- I almost had enough credits to graduate from H.S. in 3 years. I disliked H.S. I took a summer course and went to college when I was 16. Two months later I was 17.

- When I got to the U, I took engineering. I had failed to fulfill the science requirement, so I had to take bonehead science. Therefore I spent 3 yrs in H.S. and 5 yrs in U. of N.

- I joined the SAE fraternity in college. I was busy with engineering and girls and never did really get gung ho. It was OK but I could take it or leave it. After a couple of post-college rebuffs, I became inactive. They still solicit me for money.

- The fraternity held a Hell Week for all new inductees. As part of this we pledges were chained together at the wrist and deposited up in the hills to find our way home. Instead we found our way to a saw mill that had a metal cutting bandsaw they let us use. After cutting all the locks off our wrists, we cut the chain into 6" lengths and returned it to the active members who were initiating us.

- I learned to drink in college. We'd pool our money and buy Seagram's 7 by the case.

- The car racing carried over into college. I was never in an accident, but some of the frat brothers were. We mixed gasoline and alcohol but no one got killed. One weekend Nevada was scheduled to play USF in Kezar Stadium in SF. We all took off Friday afternoon racing over Donner to SF. I was leading the pack when stopped by the CHP. As he was writing out my ticket the whole student body sped by blowing their horns and waving. I was mortified and the patrolman was frustrated.

- One Engineer's Week I loaded several demonstrations on a lowbed trailer and parked it in front of the bank on Virginia Street. We ran demos all day. The most popular was the low voltage climbing arc. It looked like science fiction.

- I went to West Point in July 1947 and lasted 29 days. I had been through Hell Week to get into my college fraternity. Now the Army thought I was going to put up with a whole summer of the same crap. I suppose it was designed to weed out the cadets who were psychologically weak, and the cadets who were physically subpar. However, they also weeded out some who had different plans for their lives. I got back to Reno just in time to register for my sophomore year in engineering.

- Dad had a friend who had some WWII surplus vehicles. I borrowed a halftrack and drove it to school for a week. The transit buses didn't cut me off when I was halftracking. I had portable loud speakers mounted on it and between classes I drove the queen candidates around the campus while they sat on the windshield armor (looked like a shelf when not in combat). The girls wore short skirts and the Military Ball was a success.

- Dad's friend also had a surplus DUK. I borrowed it and took some friends "boating" on Virginia Lake.

- A year later I was advertising manager for the annual ROTC Ball. Two weeks before the Ball I borrowed a pair of walkie-talkie radios and my friend, Bill Hodson, borrowed an airplane. The plane was either a Cub or a Taylorcraft, I don't remember which. As the students passed between their 9 and 10 o'clock classes, Bill bombed the quad with 2000 leaflets. I was on the roof of a building to coordinate the attack. He made two passes from about 500 feet and papered the quad. The Ball was a success and the groundskeepers were livid.

- The college teachers lived and breathed seniority. A drafting teacher from before the war ended up as head of Mechanical Engineering. He was not a good teacher, didn't know the material, and gave open-book speed tests. As engineering juniors we were appalled because we were not learning much in some very important classes.

Somehow a revolt was organized and most of the engineering class petitioned the university president for relief. I ended up speaking for the class and the old man went back to drafting, while some talent taught our advanced courses.

- I was right guide of the band. Once when parading down Virginia Street in Reno, the parade stopped. Instead of just marking time we went into single file (snake formation). Since we were right in front of Harold's Club I led the whole college band (playing, of course) into Harold's, up the up escalator to the second floor, around the bar, and back down the down escalator to the street. It was magnificent.

- Some of my profs had good reputations that did not seem to be deserved. They were O.K., but did not cover the material well, gave tests that were easy to grade, and never counseled. One physics lab taught me several smutty limericks I still remember. Yet somehow I distilled this mess and picked up many engineering fundamentals that I found useful all my life. Go figure.

- During college I commuted across town and had 4 cars: a '39 Plymouth and three Model Ts.

- Many of my classmates were returning war veterans. They carried so much extra baggage that scholastically I was their equal. I had several real good friends but we did not correspond or visit after graduation and the friendships did not last.

- The University offered to degree all students who were called to Korean Service after the mid-point of their senior year. My ROTC commission kicked in and out I went. I went to Hamilton AFB, Lackland AFB, Wright-Patterson AFB, and ended up flying as a junior flight test engineer at Edwards AFB. My Plymouth died in Dallas enroute from San Antonio to Dayton. I traded for an Olds 88 which would pass everything but a gas station.

- After a while at Edwards I was assigned to run a data reduction service to process raw flight test data into charts for engineering interpretation. We worked in a concrete building attached to a big hanger (i.e. a lean-to). I had two adjacent offices and too much equipment and too many people. I went through channels and asked that the dividing wall between my two offices be removed. I waited and waited and the damn wall never came down although the work order was approved. In desperation one Saturday morning several other 2nd Lts and I took it down, threw the scrap out a second story window, and put it in a dumpster. My boss was livid, but we had the space we needed.

- From data reduction it was a natural move into computing. The computer came in a big box; they needed the low-bed trailer it was sitting on. I was instructed to get it off the trailer, inside the chosen building, and put it to work.

- The building had such weak floors we made pedestal plates to distribute the weight. A 1952 earthquake slid the computer off the pedestals. Only one caster punched though the floor. We were down for a week.

- I got good experience in the service, but did not develop socially very much. I also bought and traded for two more cars and ended up with a snazzy '48 Dodge convertible. It cost me $50/month in traffic tickets. I took no leave and after I got out I had about $700 cash, a good car, a college education, and some early computer experience.

- My sisters wanted to entice my aging parents to leave Reno and move back to Dallas. I went along with this and took a job with Convair in Ft. Worth. I was a misfit at Convair. I was successful in getting and accomplishing good assignments but the management and I did not see eye to eye. I matured socially, slowed down, and met Corinne.

- Convair got an early computer. When it was being set up the installation team shut down the computer, but left the air conditioning on. When they came back from lunch there was ice on the inside of the computer room windows.

- I moved to General Motors in Detroit and left Corinne behind. That was a mistake. We were married in 1955, bought a house in Michigan, and lived happily ever after. At the time, we had the only big computer in GM. It was very precious as there were only 17 computers of this size in the world. To protect that machine we held safety classes. For the first one I was demonstrating the use of a huge CO2 extinguisher. (It was on wheels and stood as tall as I.) At the appropriate time I pulled the trigger and it latched ON. I could not get the latch undone. Dry ice piled up in an 18" high pyramid until I figured it out. Except for me, the whole department was amused.

- The computer was installed in the brand new GM Technical Center. Since the computer was not in the plan, they gave us ground floor space in a high bay building intended to support the foundry operation. There were claxtons triggered by the timekeeping system that told union workers when to start/quit work. They were very disconcerting to a programmer buried in a piece of code. So I was standing on an 8' metal partition with a pair of side cutters in my hand, reaching for the claxton high above me. A couple of colleagues and my boss were steadying my feet so I would not fall (Union rules said we couldn't use a ladder.); when my bosses' boss stopped by and said "What're you boys doing?" From 14' high I stared down and said I was disabling the claxton circuit. He said "Carry on." and left.

- On one high priority job they sent a company limo to our house to pick up a suitcase, collected me and a case of punched card data at work, and deposited me at company flight operations on Detroit Metro airport. A company DC-3 then took me and my cards to Milwaukee.

- I was a star at GM and moved up rapidly in salary and good assignments. I even bought 4 new Cadillacs in a row and thought I had gone to car heaven. When I became a pawn in a GM management jurisdictional dispute, I left.

- One of those Caddys became the most expensive car I ever owned. I picked the purplish-blue color from a color chart. When the car came it was a glaring bright purple. None of Corinne's clothes harmonized with that color so she made a whole new wardrobe. Wow, what an expense.

- We sold the house and moved to Washington, D.C. We never had so many family guests before or since. Everybody visited and ate. The teen visitors were hardest to fill up.

- The plant engineers were expanding our computer room and did some welding without a fire watch. They caught the building on fire and we damn near lost the computer.

- Three of us founded Computer Sciences Corporation. We were technically close but did not really get along. We were working in Boston, and my family was still in Washington.

I had a new sports car and Corinne planned to drive up the eastern seaboard, through New York, and spend the weekend with me. I was delighted. The car blew a head gasket in New York City and after she had it repaired, the weekend was gone. I was disappointed, but proud of her.

- From Boston we took our little company to California. The guys flew out over a weekend and went right back to work. Corinne packed the house, saw it loaded into a van, loaded the sports car behind the furniture, and drove the other car to California. She dropped the kids in Ft. Worth with their grandmother and I flew over to drive the rest of the way. We were so pooped when we got to Vegas, we just climbed in bed and went to sleep. When we got to California, I went to work and she found a house, met the van, unloaded our stuff, and acted like all this was a normal accomplishment.

- We three CSC founders agreed to disagree; I sold back my stock, and started consulting. That's how I finally found a job that fitted my personality. Once when returning from a trip, Corinne met me as a blond. They were right, blonds are more fun.

- Consulting was hard challenging work, but it had its rewards. I got to see some stage shows in NYC, traveled 1st class, and ate at some excellent restaurants. I also remember buying a shrink wrapped cheese sandwich to eat after I made a close airline connection. Corinne went with me to Hawaii. She and I worked a weekend in Seattle while the boys went on a Scout hike. She also went with me to England, Holland, Switzerland, and Scandinavia. Several times I was on a tight deadline and she would fly to wherever I was, help me do the write-ups and finish the job.

- In between, she raised two boys (one an Eagle Scout), kept books, did taxes, and invested our money. We paid off the mortgage on the place in Northridge, explored Southern California, bought some acreage in the Antelope Valley, built an approved airport, arranged to have some virgin land put in alfalfa production, designed a solar (more apt term is energy efficient) home, had it built, and lived in the desert almost 20 years.

- I was associated with Datamation magazine for 25 years as author, editor, and information source. During Datamation's heyday they gave splendid parties. They were held at top notch places and they flew in all key employees, associates, and hangers-on from wherever. One year at the annual Christmas party Corinne splurged on a striking outfit. Underneath she wore opaque shorts and top, outside it was all see-thru. It alluded to more than it showed and she was the hit of the season.

- After I flipped my first airplane in a landing accident, she became weather observer, radio ground station operator, and occasionally pickup driver when the desert winds were so strong that I had to land at an alternate airport. Every time I'd land at an alternate, I would have to treat her to dinner.

- Once she killed a rattlesnake with a rifle. I bought her a shotgun. We moved into our new house in the fall, and our bitch dog had a litter of puppies the following January. Everyone should sponsor a litter of pups once. It's too much work to do twice. Being new to the neighborhood, it was hard to find homes for all nine pups. One went home in an airplane, one in a war-surplus ambulance, and one in a brand new Oldsmobile which had only 500 miles on it.

- I had a contract with the Government and a special clearance came with it. On occasion some spooks would visit the ranch to work with me. Eventually, Corinne got cleared herself and became an administrative and support spook.

- She liked birthdays. On one occasion I had a charter pilot clandestinely pick up her sister in Paso Robles and deliver her to the ranch. Corinne suspected nothing until Gus landed and Arlys popped out.

- The ranch was a lot of work, so we sold it and moved to Atascadero. The last harvest at the ranch yielded 3500 pounds of number one Granny Smith apples (in addition to the other fruits, jellies, jams, and nuts).

- I flew VFR too long before I took instrument training. Once I got caught between cloud layers and almost panicked. Another time I turned back due to low clouds only to find if I had pressed on low electrical wires crossed my flight path.

- In my work I flew mostly to airports in the LA Basin. I also flew business trips to Las Vegas, San Francisco, Phoenix, Tucson, San Diego, Santa Barbara, and Oceanside. Departing SFO I was taxiing behind (way behind) a 747. From the rear, each engine exhaust looks like an automobile tunnel. For pleasure, Corinne and I covered most of California.

- One trip was especially notable. I owned a folding three-speed bike which would fit in the back of my Cessna. Thus, if I parked at an airport far from facilities, I would unfold my bike and pedal over. When I was getting ready to go to Stockholm on a consulting job, I needed some additional documents before I left. They shipped them overnight on SAS and held them at the SAS freight counter. I flew my plane to LAX, parked in a transient area, and pedaled over to SAS (a little over a mile) to retrieve my package. The SAS agent thought this was quite unusual.

- Sometimes consulting yielded humor. Like the seriousness of IBM's administration. You could almost tell the rank of a manager by figuring the combustibility index. The more senior the guy, the more inflammable his office (wood furniture, wood waste basket, rug, and curtains on the windows). One time a manager with a wall to wall carpet moved out. The guy that inherited the office was only entitled to an area rug, so they cut six inches off the existing rug all the way round.

- In a big Government agency we had to erase the blackboards to avoid leaving any hint of our conversations. We would always leave a symbol or letter as a "confuser" just in case some spy came around.

- Security was always a hoot. The LA Times newspaper installed a computer on the 3rd floor underneath the cafeteria. The cafeteria floor was penetrated by water and drain lines. A small fire in the kitchen dripped on the computer and shut it down. Hughes had a computer with backup power installed in a building on an earthquake fault. Stanford University had backup power and other protections against student riots, but the fuel fill tube terminated in a lawn and had no locked cap on it. A stock broker had a fireproof vault installed in a high-rise. The vault walls terminated at the dropped ceiling. The overhead crawl space was open all over the floor. I accidentally turned off power in a computer center in Holland. No one knew how to turn it back on again.

- After I decided some of our computer centers were safer than the buildings that housed them I became an earthquake expert. I found that EQ education was appalling and that much was known but little was done (outside of building codes for stronger structures). After giving many lectures, and getting on TV a couple of times, I tried to improve the situation. I contacted big insurance companies about education to reduce loss experience, not interested. I proposed an EQ series to two newspapers, not interested. I contacted the State Seismic Safety commission, not interested. I contacted the Governor and my local pols, not interested. I wrote a book and contacted publishers, not interested. So I applied what I had learned to strengthen the ranch house, moved some of our investments out of state, and gave up in frustration. Now we no longer live close to an active EQ fault.

- To close, I paraphrase a New England joke: A tourist asked a local "Have you lived here all your life?" The local replied, "Not yet."

The Roots of OS/360

Robert L. Patrick, July 13, 2005

The story starts in 1951 when I met my first computer at the SWAC project at UCLA. It was an IBM Card Programmed Calculator (CPC). Typical of the time, IBM provided no programming aids to make the new machine useful. However, Dr. Ev Yowell, and his assistants at UCLA, devised a set of plugboards to control the machine's functions and to provide an interface for engineering work.

Today, software provides this interface function, but in l952 it was done with wires and plugboards. With these plugboards the user could evaluate formulas one mathematical step at a time (programming) and then record each step in a punched card so the machine could be instructed what to do.

Vern Kamm (another 2nd Lt.) and I copied the UCLA boards and set up the CPC installation at Edwards AFB, California. The time was 1952.

In the history of things, the Yowell CPC boards were the ancestor of Speedcode for the IBM 701 and the roots of the FORTRAN compiler at the beginning of the world of engineering computing.

I got out of the Air Force in 1953 and went to work for Convair in Ft. Worth. Convair had ordered IBM 701 Serial No 7. I took programming instruction in NYC and helped with the installation in Ft. Worth. By the time the 701 got to Ft. Worth, John Sheldon of IBM had devised a software system called Speedcode. Speedcode was an interpretive system which provided features similar to the CPC plugboards used by the Yowell team at UCLA.

In 1954 Convair had a quantity of engineering applications which were programmed and tested. The customers in the engineering department needed to run case data through these completed programs to support the design of the B-58 Hustler. In those primitive days programmers were an independent lot, with no formal schooling in programming or computers. All of us had at least one degree, usually mathematics or engineering, plus on the job training. The only common trait was an aptitude for the detail work of programming.

Once a job was tested, the programmer was quickly given another assignment and lost interest (and skill) in his past work. However, the original programmer was still saddled with setting up the case data for his engineering customers and running it on the machine. This was called programmer-present operation as the original programmer worked the computer console, hung the tapes, and supervised the printing of production work.

Not only was this a significant distraction to the programmer developing new code for another job, but quite frankly, many were not very good at production operations. They would forget something so the machine would sit idle while they ran after the program/data/run book they

forgot; or a machine error in the middle of a long run befuddled them; or they were all thumbs hanging tape on a manual drive; etc. I'm sure you get the picture; their programming aptitude got them hired and they weren't very good operators.

I had faced this problem before (at Edwards) and knew how to write instructions so someone other than me could successfully operate using the programs I developed. I applied this experience to the Convair situation and documented my jobs so I did not get (many) urgent phone calls.

Soon my documentation standards became the norm and a bright person with good finger dexterity was hired to be our first production operator. The time was 1954, and the woman hired to run first shift production eventually became my wife, Corinne. She may have been the first production computer operator.

Computer time was at a premium in those days. The goal was not to reduce cost, but to increase raw machine availability. I set up a production second shift for overnight production runs. "Check in the data before you go home, and we will have the output ready in the morning."

Eventually, as the programming staff grew, machine time got very tight. The jobs taxed the reliability of the machine. Further, if you were processing input or printing output the machine was just as tied up as if you were doing heavy calculations. The limitation was in the single sequencing hardware.

Speedcode supported magnetic tape, but it was not popular with the programming staff. I had read an industrial engineering article on time and motion studies (later published in Gantt on Management, A. W. Rathe, AMA, 1961) and found I could do useful debugging work while another programmer was just getting organized at the console. By storing programs and data on tape I could load my job in a few seconds, process a few change cards containing data or programming patches through the card reader, and get the printed output from a test shot without delaying the incoming programmer or impacting the formal schedule.

This tape to tape operation was very efficient for the configuration then in use (small primary memory, equivalent sized drum storage for swapping, and fairly fast magnetic tapes). This experience proved some ideas I had for improving computer production operations.

In 1954 I moved to General Motors Research in Detroit, and IBM had announced the Model 704 to replace the 701. To the customer the 704 offered more reliability, more primary storage, faster circuits, and a pair of offline machines which would read cards and write magnetic tape, and read tape and print - - both completely independent of the mainframe. Suddenly I was center stage as I had been simulating that style of operation with my Speedcode sneak-in tape operation for over a year.

SHARE, the IBM scientific users group, had been formed in Los Angeles and at their third meeting I proposed a new operating system for the 704. Owen Mock of North American had some similar ideas. A project was launched, the work was divided, and in a short time we had an operating system for the 704. (Really two versions were produced because there was insufficient time to fully negotiate on one set of features, but the code in the two versions was 95% identical.)

As we went into operation at GMR with what was called the General Motors I-O system, programmers built test or production decks at their desks and every hour a mail person would pick up the decks from the programmer's desk and then deliver them to the card-to-tape machine.

As the decks arrived and before the cards were read by the machine, the human operator sequenced the batch of programs on external priority. The decks then went Card to Tape in a format called Binary Coded Decimal. The tape was then handed to the mainframe operator in the next room.

When the mainframe was available the tape was hung and the batch was run. There were no run books. The process was non-stop: An input phase converted all the input tape to binary and wrote a binary version of the BCD tape produced by the card to tape machine.

The binary tape was then rewound and the jobs loaded and executed in sequence. If a job hung up or the system crashed, a standard button sequence dumped the offending job and started the next one from the input tape. As jobs produced output it was written in binary onto an output tape by the originating program. When the input tape was exhausted, the compute phase was over and the output tape was rewound.

In the third phase, another conversion program was automatically loaded and it converted the binary output tape to BCD tape which was then handed to a Tape to Print operator in the next room. Printed output was physically matched with the original card input and both were returned by messenger to the programmer's desk.

In addition to getting more jobs completed per day with the operating system, George Ryckman (later SHARE President) designed and built the first time-of-day clock. Using this clock, the system printed an advisory invoice as the last sheet of each job showing the resources used. The programmers exercised their self-discipline to use machine resources wisely. Later, this same information was used to bill projects for "resources used or denied to others."

If all went well, two operators ran batch upon batch without working too hard. The programmers got desk to desk service, and the CPU ran non-stop as long as there was work to do. Test and production work ran intermixed. Trouble was handled by standard procedures. The time was 1955.

Internally the system was efficient because the system programs for input and output translation were much larger than the size of the usual job. Thus the translator programs were loaded from drum memory only once per batch and processed a whole tape before relinquishing control.

As in earlier experiments, standard input card formats, standard deck setups, and standard calling sequences for printed output provided a way for efficient programmer-absent operations by operators who knew the system but not the job stream. I seem to remember that some variant of the NAA/GMR operating system software was used in 43 installations.

The next IBM machine in the scientific sequence was the IBM 709. In addition to a bigger primary memory and more reliability, the 709 offered I-O channels. A channel, despite the IBM parlance of the day, was nothing but a special purpose computer which shared the memory bus with the mainframe. On the other side of the channel controller one attached all of the electromechanical input/output devices: Tapes initially and later readers, printers, and punches. Still later, additional devices could be attached such as a very large disk drive, and eventually through a channel-to-channel connection, another computer.)

In the days of the 704, IBM offered a little software, but the user community did their own thing (led by the user groups SHARE for the scientific community and GUIDE for the data processing community)]. Sometimes an installation used IBM software, sometime software developed by the user community, sometime home-grown code, or a blend of all three. While there was some standardization within an installation, there was less software standardization between installations serving different corporations.

IBM saw this chaos and decided to support, as a manufacturer, software development across a machine type serving a variety of similar users. Thus the IBM-SHARE software project was born. It was a committee made up of people who had successfully done things and a horde of IBMers who wanted to pick our brains. Owen Mock of NAA and I were both members. The effort produced the SHARE Operating System (SOS), a FORTRAN compiler, and eventually a whole bag of other programs.

However, before the first committee meeting I got an engineering assignment from GM. Overnight I resigned from the SOS committee and was assigned to support a big missile project. From there I went on to manage a big service bureau and then help found the Computer Sciences Corporation.

After I left CSC I started consulting. While I was helping out at the Aerospace Corporation, we discovered a fluke in the pricing of the IBM 7040. You could buy a 7040 for about the same as a couple of 7094 channel boxes. We needed more throughput and scheduling flexibility so we designed the Direct Couple extension to IBSYS (aka Attached Support Processor).

Several things about the Direct Couple were interesting. First it never would have existed if IBM had not goofed on the pricing of the IBM 7040. The 7040 could service all I/O devices of interest in a large scientific installation, could communicate channel-to-channel with the 7090,

was compatibly programmed, and was cheaper than a couple of channel boxes. Once the users discovered this, IBM got a lot of channel boxes back for junk, built a lot of new 7040s, and made less money.

Another interesting thing about a 7040-7090 combination was the internal scheduling we built into the 7040 software. Jobs were entered into the 7040, either remotely or directly, and stored on disk. A scheduler looked at the external priority, the current workload, and the characteristics of the job being scheduled. A scheduling algorithm on the 7040 called for tapes and when the tapes were mounted passed the job to the 7090 for execution. As output was produced by the 7090 it was passed to the 7040 and stored on a disk. When a printer was available the output was converted and sent to a printer by the 7040. All the while the 7090 was churning away on some heavy calculations. All of this is reminiscent of early three-phase operating systems except no human operators were involved in the scheduling.

However, if printers were available, and no output had accumulated, the scheduling algorithm ignored external priority and used job characteristics to send jobs with more output and less compute to the 7090 so the printer capacity would not go to waste. Thus it was not unusual for jobs to exit the system in a slightly different order than their priority would dictate.

Other options in the system provided direct simulated online device support if your external priority was blistering high (as if supporting a missile launch). A different algorithm entirely was used for overnight service which attempted to optimally exploit all available resources since the customers would not pick up their output until the next morning. All of this was an early form of adaptive scheduling.

On top of all this, the machine room had conveyor belts behind each printer so the operators were not tempted to queue completed output behind a printer since they no longer had to walk over and place the output in the appropriate bin for customer pickup.

Later, when I was consulting with IBM, I worked for Fred Brooks as part of the cast of thousands that produced OS/360. My assignments did not involve OS architecture, but I drank martinis with the guys that were responsible. Deliveries were late and the early code was slow. Early manufacturing and hardware deliveries had to be scheduled around software availability. Much tuning was required by early users.

When North American installed a 360 Mod 40-65 to replace a highly tuned 7044-7094 (an upgrade from a 7040-7090 Direct Couple), the 360 configuration could not do a day's work in 24 hours . . . very embarrassing.

I got into data base software at the Space Division of North American (builders of the Apollo space capsule). I had been working at Space on general computer consulting work for some time (they had 7044-7094s and were replacing them with Model 40-65s). Uri Berman and Pete Nordyke came up with a piece of software (that they had previously written for the IBM 7010) which allowed the description of a file to be encoded in a table and a general processing

program to use this encoding to manipulate files of information (the 7010 online application was a list of current parts for the Apollo capsule). On the 360 this was called Data Language/I. I was on the development team and did some of the architecture to wrap a control program extension around this package. The result was IMS.

There were several innovations from this project. The online software ran as an application under OS/360. This made us independent of OS maintenance. The software would support several simultaneous online applications, and if one failed the others continued to run (as long as the OS survived). The entire configuration was measured and tuned (both hardware and software configurations) for the workload expected. During the latter stages of system test one Mod65 simulated a load of terminals and drove another Mod65 running the test software. For transactions with the characteristics found at Space, 4000 terminals could be supported. And last, but not least, R.R. Brown (one of the managers at Space) coined the term "data base manager" and hired the first one.

IMS was the first commercial data base management system. It was a resounding business success for IBM and after many years of continuous development is still in use today. Berman got a cash award from IBM. It was said that a $50,000 check was attached.

Three Vignettes

The following are three vignettes that Robert Patrick prepared to attach to this oral history.

Early SHARE

Los Angeles aerospace corporations were the hot bed of engineering computing in the late 1940s and early 1950s. Northrop had spawned the Card Programmed Calculator (CPC) and in 1949 Cuthbert Hurd of IBM sponsored an early (perhaps the first) user conclave in Endicott. The CPC was an externally programmed computer.

The stored program concept attributed to John Von Neumann was then state of the art. The Bureau of Standards built two experimental stored program machines, one in the East, and another on the UCLA campus. In this same era, Willis Ware constructed the Johnniac at RAND.

IBM, sensing some business (or perhaps embarrassed by Eckert-Mauchly), launched the 701 project with a whole string of people who would later become notable. IBM thought the market small and planned just 17 machines (but later made an 18th out of spare parts).

I used a CPC at an isolated outpost in the California desert, and visited RAND to get some invaluable polynomial curve fits that Cecil Hastings had done for the CPC. After I got out of the USAF, I went to work for Convair in Fort Worth who had 701 No. 7 on order.

While I was working in Texas; RAND, North American, Douglas and Lockheed met, designed, and built a set of program preparation software for the 701 called PACT. The friendships and relationships formed in this pioneering effort became SHARE. I was not part of this initial effort. From memory I recall that Paul Armer, Jack Strong and Frank Wagner were among the ringleaders.

The formal SHARE organization came together under the auspices of Blair Smith, an IBM Rep who had a big job and a big expense account. I missed these kickoff efforts and the open-bar that accompanied them. About the same time as SHARE was being formed for laudable technical purposes (with a focus on IBM iron), the Digital Computer Association (DCA) was formed in Los Angeles for inter-manufacturer information exchange. All the local SHARE attendees also attended once-a-month DCA meetings to let their hair down. We were all in our late 20s or early 30s and were "Hail fellows, well met."

I met many of these early pioneers at SHARE III in Boston and fell right in with the rowdies. Early SHARE attendees fell into three classes: IBMers, Administrators, and Technicians. I was

a technician. The administrators managed their installations, ran SHARE meetings, and attended every one. We technicians attended meetings only if we had something to learn or say, or if we were being rewarded for some job well done.

At SHARE V in Chicago the SHARE suite was very popular after the formal sessions were over. Mario Juncosa (a Ph.D. and a very reserved RAND mathematician) and I were competing for the title of "Last Man Standing." When time came to close the suite, we were perplexed about several half empty bottles of booze. There was too much left to drink, and we did not want the cleaners to have a party on us. There was a spinet piano in the suite so we lifted the top and hid the booze in there. Before the session next morning, I visited the suite to see if the booze survived the night, and met Mario doing the same thing.

In 1956 I worked at General Motors Research with George Ryckman (later a SHARE President). I was entitled to buy one new car a year at a discount so I bought a Cadillac Coupe. When the Dallas SHARE meeting came along we drove. My wife and I extended the back seat with suitcases and made a pallet for the boys. We were motoring along under a canopy of stars and had just crossed the Oklahoma-Texas border cruising at about 90 mph when I picked up an escort. After we stopped and explained ourselves, the cop looked in the back seat and found two little boys asleep. He let us go with a warning. He said that we fit the bootlegger profile. Somehow that seemed appropriate since we were going to SHARE VIII.

Operating Systems

In 1954, we needed more computer time on the IBM 701 than was available. Additional computers could not be ordered (even if Convair could afford them) and our computer was already running three-shifts.

These were the days when unions objected (sometimes rightfully) to management sponsored time-and-motion studies and efficiency experts. One of my engineering magazines mentioned a new book, about the history of Industrial Engineering which contained a section on time and motion studies. (I think the author was Gantt.) I got the book, transformed his analysis of manufacturing layouts and materials handling into an analysis of machine operations and throughput. I ran some experiments using the programs I was developing for the airplane design engineers I was programming for, and found I could dramatically increase the throughput (measured in jobs per 8-hour day) with some software and some machine room organization.

The next year I moved to General Motors Research and got an assignment to organize the work flow through their (new) IBM 704. I did a design; presented it to the third SHARE meeting in Boston, and a software project was born.

North American Aviation (aka Rockwell) and GMR wrote the software, the results were as planned, several other SHARE installations picked up the software and the field moved forward with the first stacked-job operating environment.

Over the years, other systems were influenced by these developments: IBSYS for the 709-7090; the Direct Couple extension to IBSYS; and OS/360. This was all due to the analysis methods of the industrial engineering triumvirate (Taylor, Gilbreth, and Gantt).

Commercial Compiler

In 1959, Dick Clippinger of Honeywell contracted with Fletcher Jones, Roy Nutt, and I to produce a natural language compiler for the H-800. Clip thought that an English language compiler would broaden the appeal of the H-800 and pickup additional market share.

Jones, Nutt, and I formed Computer Sciences Corporation and setup shop in Los Angeles. Jones was the salesman/administrator, Nutt was the principal designer, and I was in charge of the development team. Clip's name for the product was FACT, for Fully Automatic Compiling Technique.

Jones and a team of SHARE volunteers had produced SURGE, a report generator for the IBM 704. Nutt was famous for SAP, the Symbolic Assembly Program for programming the 704 in machine language. And I had architected a stacked-job monitor for the 704. We were all about 30 and this was the opportunity of our lifetime.

Sometime over the last New Year, some ignorant journalist denigrated COBOL as providing programming capability using a form of pidgin English. Well, the story really starts in the period between WWI and WWII. International trade was building and groups of linguists on both sides of the Atlantic were striving to define a language that would promote World trade. The competing offerings were Esperanto and Basic English. I found a book called "Basic English" by C.K. Ogden (1930) which offered a simplified sentence structure, a restricted vocabulary, and contained a sample copy of Lincoln's Gettysburg address using the proposed system. I thought it would serve our purposes and we adopted and adapted it.

I wrote the initial users manual for FACT and included in it constructions in source language. Honeywell supported Grace Hopper's effort to produce a common language for business data processing (and level the playing field with IBM). Nutt was the Honeywell/CSC representative on the COBOL committee. COBOL adopted some of FACT. Some of my examples were carried over too. 20 years later I looked at a COBOL manual, and my examples were still there. "Pidgin English" was a mature linguistic development. That's why it lasted so long.

Milestones in Computing

Robert L. Patrick, February 2006

Disclaimer:

This is the first draft of a proposed Timeline of major descriptive milestones in the civilian history of computing excepting: imbedded military systems and computing elements providing special functions in civilian systems where computers are only a minor component, e.g. watches, automobiles, and appliances.

Milestones in Computing is proposed as a series of single page presentations called “plaques” which encapsulate and dramatize the major events in computing in time sequence. At this writing they are all from my memory without any validation or research. When you read them, see if the series distills the progress of the field into understandable discrete units. If the series appears worthwhile, then comes validation, fact checking, and finally a juried panel of experts to say what's in and what's out.

Some plaques contain names of individuals. These should be verified when other facts are checked. If other individuals are found to be the principal contributors when the fact checking is done, their names would be woven into the appropriate text. Note the plaques are meant to chronological. When facts are checked, the sequence that follows will change if key dates change.

Milestones in Computing --- Contents

Milestones in Computing Alphabetic Index

Milestones in Computing

Introduction

Since the modern computing age started in the late 1940s, there have been a few hundred computer designs, tens of thousands software packages, and a few million application programs. Some of these combinations of hardware/ software/applications were innovative, some were incremental improvements on previous packages, and some were 'me too' copies produced for the competitive marketplace.

This series, *Milestones in Computing*, will try to illuminate the combinations that shaped the field. Not to denigrate those creative individuals, companies or institutions that tried hard or produced good systems for a specific need, but to recognize the combinations which, perhaps at the confluence of several ideas, made an outstanding contribution to the (then) state of the art.

To begin this review of computing history, note that the field advanced in eras. In the late 1940s the punched card era was at its peak and several pioneering electronic computing machines were installed in 20 or so locations throughout the country. After a few years of transition, the era of "big iron" was born. In this era computers built of discrete components and weighing a few tons were the norm. This era started in the 1950s and ran 30 years through 1985 (or so). As the big machine era stabilized (it is still with us today), it was forced from center stage temporarily by the mini-computer and then all activity became focused on the desktop personal computer. The communications revolution followed the PC and produced the networks of today (2006).

Each of these eras had many innovative developments. Some were immediately successful, some took a while to mature. The key innovations that I believe shaped our world are highlighted in the series.

One further note about the format of what's to follow. Each milestone contains a bit of background, the system being highlighted, and eventually could contain some footnotes to learn more about the era, the companies involved, the individuals who made major contributions, and other similar systems of the time. The files of the Computer History Museum contain an abundance of historical information, much of it available via the Internet.

1. Card Programmed Calculator

The peak of the punched card era occurred in about 1947. Until that time a punched card customer got a machine, several blank plugboards, a stack of manuals detailing the hardware functions available at the plugboard, and several pounds of plug wires of various lengths and colors.

The customer had to define his job, design the card layouts for his data, and prepare plugboard wiring which would cause the machine to read one or more data fields from a card, perform the necessary calculations, and then punch the calculated result back into a card (usually the same card, but sometimes a different summary card).

This machine system was designed to treat (calculate) every member of a large punched card file in the same way, and to run through a large deck of cards from start to finish without interruption.

The setup of a new application involved all the above, plus a test deck to check the plugboard wiring, and companion plugboard setup to read the result cards, complete the processing, and print out the results on a separate machine called a tabulator.

The rate of processing for each machine was about 100 cards per minute. A large deck of 10,000 cards took over 1 1/2 hours to calculate and list.

In 1947, some engineers at Northrop Aircraft cabled the calculating punch and the tabulator together. In 1948, IBM held a symposium and discussed a set of general purpose plugboards which would read commands from a punched card, perform the desired calculations, and retain the results in mechanical memory awaiting the next command from the following card.

For engineering work, involving complicated equations and a small amount of data, these general purpose plugboards were an enormous step forward. No wires to move, no boards to change, just build a deck of cards containing the correct sequential command stream, insert data cards in the proper place, and run the job. For short jobs of medium complexity the preparation and setup time went from days to an hour or two. Further, the skills required were easily learned (by engineers) and the production runs could be done at any time of day by machine clerks.

This was an externally programmed calculator. It may have been the first instance of software (although the term was not then in use) making a machine user-friendly, increasing productivity, and reducing the skill level of the typical "programmer".

2. Service Bureaus

After the dawn of the industrial revolution, small companies found they sometimes needed occasional access to the same expensive tools, facilities, and skills as large companies in order to compete. The independent job shop was the result. A job shop was sometimes a cooperative, and it sometimes was independently owned and contracted its services.

Work load flows to a job shop to get processed and the results are returned to the source. In manufacturing, machine tools too expensive to install and let sit idle, were "shared" by many small companies to obtain a needed service at a reasonable cost. Sometimes the services obtained were skilled personnel, sometimes machines, and sometimes a combination of both.

ADP (originally founded under a different name) was established in New Jersey to process punched cards and run payrolls in1948. It provided a necessary service to companies too small to run their own punched card shop, companies which lacked the financing to have their own installation, or companies which chose to use their funds for other business purposes.

Over the years ADP converted their customers and their services to computers and greatly increased their volume of business. In the nineteen-fifties and nineteen-sixties many other service bureaus were established around the country to supply punched card and then computer services that were in short supply. Some, like C-E-I-R also offered analysis and programming services that supplied a steady stream of jobs to be run on their service bureau machines.

These early facilities, and the ones that exist today, have had a volatile existence. They must keep up by offering the latest technology and yet continue to offer the older services that are part of their business base. Time has proved that the service bureau business is rewarding and tough.

3. 701 Speedcode

In 1953, IBM delivered an electronic computer whose program was stored inside the computer just like data (a true Von Neumann machine). The stored program concept, so popular for the last 50+ years, allows the internally stored program to be modified by the ongoing calculation. Just this one feature provided the programmer more flexibility, made for shorter programs, and delivered a remarkable increase in speed. Not only were the circuits of a 701 faster than those in the CPC calculating punch, but that speed could be exploited after the job was loaded, because the speed of processing was no longer paced by the 100 card per minute card reader.

However, the 701 was difficult to program. As in the punched card era, the 701 came only with plugboards, wires, and manuals. The user had great flexibility, but it came at a high price. The machine language was binary. While some programming aids followed the machines into the field, they were not standardized or integrated. Most were separate contributions from individuals.

It frequently took days/weeks/months for a programmer to reduce a problem statement to binary form and test it. Very rare was the situation where a small job could be programmed and tested in a day. Further, if one programmer had a very high priority on the machine, which made the machine unavailable for the rest of the programming staff. It was not unusual for a shop to be virtually shut down by a high priority job. On the other hand, a well designed and checked out binary program used machine resources efficiently in production.

Enter Speedcode. John Sheldon and a team of programmers from IBM designed an interpretative program for the 701 which programmed a lot like the general purpose setup for the CPC. It used standard plugboards and a command interpreter to make the 701 user friendly. The time was 1954. With the Speedcode package loaded into a 701 and residing on a high speed drum, the programmer could forget binary and program in decimal. Checkout aids were integrated into the system and magnetic tapes were supported for large reusable files.

With Speedcode loaded, a job of moderate size could be programmed, tested, and run in a day. For jobs involving only a few sets of case data, Speedcode was a dream. However, if the job was large, or ran many sets of case data, the execution inefficiencies of Speedcode were eclipsed by a well programmed job running in native (binary) mode.

Fortunately many engineering jobs were small to medium sized, changed frequently, and ran little case data. For them Speedcode gave the customer results when he needed them.

Unfortunately, programming management of that day never exploited Speedcode as the first step of a programming process that involved: Speedcode to prove the problem statement and get early results while a more efficient program was prepared in binary. Times changed so rapidly that a translator program was never prepared to translate a tested Speedcode program into binary language.

4. Bank Paper

In the mid-1950s U.S. banks were drowning in paper checks. The Bank of America and General Electric (later joined by Pitney-Bowes) launched an innovative program to handle checks that had been honored and needed to be returned to the originator.

The heart of the system was Magnetic Ink Character Recognition (MICR). A special font was designed which could be read by humans and processed at high speed by machines with a low error rate. Machines were then designed which would move this paper without damage and pass the information read from the MICR characters to a dedicated computer for further processing. To complete the equipment set, small machines were designed so a human operator could view the check and use a special keyboard to encode its amount on the bottom of the check. Special checks were required which were preprinted with the owner's home bank and account number in this special font.

Perhaps the biggest success of this project was to sell this system as a standard for use by the Federal Reserve and all member banks in the USA.

After a few years of use, the system settled down, all the banks complied, and batches of checks clearing an entry bank were encoded and automatically balanced. Further, back office processing was dramatically more efficient and canceled checks could be efficiently returned to their originators.

5. Tool Control

In the mid-1950s designers of high performance fighter planes for the USAF found that wings milled from big blocks of aluminum were lighter and stronger than traditional wings built up from individual components. Further, the integrally stiffened wings could be fitted with fuel cells which were less prone to leakage.

Kearney-Trecker, an established maker of large machine tools, designed a milling machine which would take enormous blocks of aluminum and produce finished wings for fighter aircraft. The milling cutter was controlled in three dimensions by a dedicated special purpose computer.

A human tool engineer read the engineering drawings for the part and prepared a series of commands for the tool head. The setup had a carousel of tools available so the machine could select the proper tool for the cut to be made.

This pioneering work formed the basis for other numerically controlled machine tool systems, and eventually today's assembly robots.

6. Formula Translator (FORTRAN)

In the mid-1950s IBM was preparing a replacement for the 701 which was faster and more reliable. It was designated the 704. A team of programmers, led by John Backus, produced a program to accept formulas, stated in a stilted but easily fathomable language familiar to engineers and mathematicians, and produce running binary code for the 704.

While FORTRAN did not eliminate the backlog of the programming jobs facing users, it was wildly successful and made a big dent in it. FORTRAN was so well designed that it even

supported early users who were doing commercial work manipulating bits and parts of a word in storage (work not normally thought of in terms of engineering equations).

As a result the programming backlog was reduced, programmer training could be focused, and more computers were sold. Competing vendors rushed to produce similar language processors for their platforms and other language translators appeared. Several attempts at language processors for commercial applications were produced before COBOL, and academic communities prepared processors for their own needs such LISP.

7. Operating Systems

In the late-1950s there was a terrible shortage of computer time. The machines were used mostly for program development, the customers were calling for production runs, and a micro-analysis of machine room activities showed a lot of operational inefficiencies and idle time. By applying the industrial engineering analysis techniques devised by Laurence Gantt in the early 1900s, the inefficiencies could be enumerated. In these early days a programmer was free to choose his own operational techniques, his procedures for run preparation, and to operate the machine him/herself. Some were very efficient, and some were . . . lousy. The machine schedule was assigned in blocks of a few minutes and if a test run crashed or terminated early, the machine sat idle.

At a SHARE meeting, Bob Patrick of General Motors Research and Owen Mock of North American (nee Rockwell) decided to implement a batch operating system for the 704. There would be standardized procedures for input and output, a standardized job control language (although this phrase was unknown at the time), professional operators, and features to aid in salvaging runs that crashed. The operating system would be in sole control of the machine and if the human operator sensed a crash or a runaway error he could terminate the job and the next job would be loaded automatically. A variety of input translators were available via the job control language including decimal to binary data conversion, binary pass through (for previously prepared program decks), a symbolic assembler for program translation, and a FORTRAN compiler.

On the output side, cards could be produced, data saved on tape, and custom formatted reports printed. As part of a companion development, IBM produced standalone machines which would read cards and prepare magnetic tape, and read magnetic tape and print reports. These peripheral machines were typically installed in a nearby room so the main computer ran in a virtually dust free environment and was devoid of programmers. Some shops provided desk-to-desk pickup and return messenger service so programmers could concentrate on their specialty - devising problem solutions and translating them into machine readable statements.

In an engineering/scientific installation, performance/throughput/operational productivity was measured by jobs per day. With block time scheduling and programmer present operation, 6 jobs per hour were scheduled. A smooth running tape-to-tape system which delivered 10 times that quantity was not unusual, with no increase in machine rental dollars.

As part of this pioneering effort, George Ryckman, later a SHARE president designed and manufactured a time-of-day clock. This allowed the operating system to log jobs on and off the mainframe, allowed the operator to compare actual running times to the programmer's estimated time for the job, and allowed advisory invoices to be prepared and printed at the end of each job so the programmer could know how much machine time he had used during his run.

Later IBM machines had evolutionary improvements to this initial operating system and other manufacturers devised their own methods of increasing the productivity of the computer room.

8. Programming Houses

In the late 1950s the young computer industry was exploding. Users had discovered computer applications they needed to compete in the market place and while the then dozen or so manufacturers could supply equipment on a reasonable schedule they had no trained people to offer.

The colleges and universities were grappling with the same problem. The schools had not yet added computer (later called information systems) courses to their curriculums, and there were no trade schools on the horizon.

Meanwhile, large companies that had computer experience were not managing their work force to keep the talent they already had. Limited wage differentials between the talented and the merely competent tended to encourage the talented to go elsewhere. Some of the talented had entrepreneurial feelings, and some of the talented saw an opportunity to physically move to a more desirable geographic location without fear of unemployment.

Out of this milieu came consulting opportunities, programming contracts, and (at higher risk) the development of generalized program packages which could be repeatedly used by and later sold to a variety of customers with similar needs.

Computer Usage Corporation came first. In 1955 four talented individuals with early computer experience established CUC to provide consulting and programming services in the New York area. Sometime later, in 1959, Computer Sciences Corporation was established to prepare software for one manufacturer's latest hardware announcement.

Many other companies followed these first two. Some were outgrowths of service bureaus and contracted for analysis and custom programming, some supplied only trained manpower (body shops), some concentrated on packaged products, and some offered all three basic services. The business was hard, like the service bureau segment that preceded it. In the late 1960s companies offering timesharing appeared.

Economics drove some out of business, and some of those who were able to adapt still remain.

9. Medical Systems

In the early 1960s two physicians, Drs. Weil and Shuban, installed a process control computer to support the nursing staff in a shock ward. After a patient experiences shock, he/she is very unstable and requires constant observation; because a second incident could result in death if not immediately treated.

With a grant, a computer as big as a Cadillac, some programming, and the development of a set of special sensors for bodily functions (measuring urine output was the hardest challenge) resulted in a patient recovering from shock being placed online where monitoring was continuous and alarms were automatically sounded.

All was not bliss however. As installed, the nurses thought the computer system was just as demanding as another patient. Enter Tom Rockwell, an electrical engineer from a medical family who made the system stable and nurse friendly. Next time you have a serious exam or undergo an operation, note all the computer assisted (electronic) monitors which watch over you and help the medical staff. A small part of their contribution to your wellbeing is owed to Drs. Weil, Shuban, and Rockwell. (After Tom stopped programming he got his medical degree.)

10. Single-Site Networks

Single-site 'point-of-sale' networks were popular in the early 1960s. IBM designed and manufactured terminals that had only specified functions and were suitable for installation on the shop floor of a big manufacturing/warehouse operation. These terminals were connected to a central computer by dedicated copper (telephone) wires. The wires terminated at a control unit about the size of a piano which in turn was channel connected to a dedicated medium-sized computer.

The computer was programmed to log all incoming transactions (for restart purposes) and update a set of related files maintained on giant disks. The disks of that era were not removable and the diameter of a single platter was about 30 inches.

A person at a remote terminal could ask a question about the status of an item, and quickly get a response on his terminal. Movement reports for parts between workstations could be entered so the location of a part/assembly was always known to the computer. At the end of the day the computer would put activity and summary information on a magnetic tape which was then passed to the mainframe computer for overnight processing.

When the Apollo capsule was being built for its trip to the Moon, a computer system at the Space Division of Rockwell held a file of active parts. Whenever any engineer needed to reference the latest drawing for a specific part, he/she interrogated the system to see if the drawing in his/her possession was the latest edition, or whether a fresh copy had to be obtained.

Two programmers, Pete Nordyke of Rockwell and Uri Berman of IBM, had devised a program package which divorced the definition and maintenance of the Apollo data files from the dozens of programs that used/added/ processed data to/from those files. This package of code, later prepared for the System/360, became Data Language-I, the heart of IMS/360.

11. Geographically Dispersed Systems with Central Processing

The geographically dispersed system required a major step forward in the management of the system. This management usually crossed several time zones, several communications companies, and many locations. In contrast with single-site systems, experienced technical talent could not jump in a golf cart and buzz over to solve the problem. They had to work through local maintenance personnel, and in many cases, operators without any technical skills whatsoever. Further, the early systems did this with hardware barely reliable enough for the task.

The pioneer system was Sabre, developed by a team from IBM and American Airlines on 7000 series equipment, and then redeveloped in 1966-68 to run on an S/360 platform under OS/360. Once Sabre showed the way, reservation systems followed for banks, hotels, cruises, concerts,

stage shows, libraries, etc. The hallmark of these systems was dispersion with a central data base - - many terminals with restricted function, high reliability, and a body of precious data on which the business depended.

Processor malfunctions when updating the main file were particularly onerous. Using twin mainframes, modifying the operating system so that either mainframe could seize the primary load in an instant following a fault while maintaining the integrity of the data files, was an extremely difficult task.

To get ultra-reliability from equipment designed for average conditions, these systems were first put together on equipment from a single vendor. Later as understanding and techniques improved, systems with equipment from several manufacturers were blended together. DEC computers were frequently used for communications services in later developments.

12. Dispersed Systems with Dispersed Processing

Managements of large corporations have long worried about survivability. They buy parts from two or more vendors, negotiate carefully with unions, and where manufacturing volume allows, assemble their products at two locations so one strike, one hurricane, one big snow, or one earthquake can not severely impact the bottom line.

Once the geographically dispersed systems proved reliable and economically feasible, senior management became aware of (and worried about) having all of the company's precious data stored in just one place. As their understanding grew, they then worried about all communications coming through one central telephone office.

It was a big technological step to move from single site processing to multi-site and then on to distributed processing. Computers were cheaper and smaller, but the technical challenges were bigger. Staged backup, systems and procedures designs to hold original data until the transactions were completely processed and the files successfully updated, system security, alternates for likely outages, and communications encryption were formidable challenges even after a system was up and apparently running smoothly.

The Bank of America pioneered with two massive, dispersed, linked, processing centers for the State of California. If one system suddenly went down, the other center had the programs and the files to seamlessly take over the whole load.

Fireman's Fund (American Express) installed office computers which supported several onsite insurance terminals and then contacted the central system for access to the main file set.

The Internet made routine communications easier, but many important problems remain which require careful analysis and design.

13. OS/360

In the early 1960s, IBM was awash with the number and variety of systems they had to support in the field. This inventory of unrelated equipment and software was inhibiting their further growth. In a series of legendary design sessions, led by Drs. Brooks, Amdahl, and Blaauw, the hardware architecture for a family of bit-compatible computers took shape.

The software for this family was another matter. The first software design was followed by a second. This second effort, led by Fred Brooks, Dick Case, Scott Locken, and Don Gavis, was delivered as Operating System/360. The System/360 was announced in 1964, and some of the smaller machines were actually delivered that year. But the larger machines needed the advanced OS which was not available until 1966. Initially, OS/360 lacked both function and performance, but by 1967-8 most of the problems were solved and it went on to be the most successful software operating system ever (until Microsoft's PC software came along).

The story of OS/360 is well known, but the record fails to emphasize some of the innovations that system delivered. The hardware was bit-compatible, i.e. a checked out program would run on any machine without change if the configuration was correct. The software also had compatibility as a goal, but due to differences in configurations of hardware and software, some minor adjustment was usually required to move a production program from one platform to another (typically larger) platform.

In addition to compatibility, OS supported a growing variety of I-O gear. There was the usual menu of readers, printers, punches, tapes, and disks; and there also were communications controllers, custom terminals, photocomposers, and channel to channel connections.

Running software systems were built by an automatic process called SysGen. It took the vanilla factory system and customized it to run in a customer's unique configuration. The customized system was placed on a disk and resided there during operation. The advantage of disk residence was, of course, uniformity of access - - all parts of the operating system were equally available.

OS/360 was designed to run multiple programs simultaneously. The compilers produced object code which was relocatable and the loader bound the remainder of the addresses before execution. The system kept track of the resources used (or denied to others) by each user. The hardware had boundary registers to keep each user in his assigned space (and away from the operating system). The OS maintained these fences and detected any violations by runaway programs. Thus production and test jobs could be intermixed. Jobs were not batched, but introduced when ready to run. Hardware interrupts from the communications controller were handled immediately by setting the background job aside and allowing the foreground job to perform priority processing. When the system was installed and settled down, an online terminal job frequently ran all day in the foreground while a series of batch jobs ran to completion in the background.

After delivery, a data base system was added so both foreground and background jobs could access common files containing current data. After a few years, general purpose programmer terminals were supported so programmers could remotely create, change, and submit jobs for execution. A set of vital changes allowed the entire job-set under execution to be protected so if one job failed, the others continued to run (almost) without interruption.

Initially, the system was big, not totally checked out, somewhat slow, and not totally secure. As bugs were found they were fixed, performance improved, and security got better (but remains a community-wide problem yet today).

14. Point of Sale

In the late 1960s, IBM devised a system of barcodes which could both uniquely identify a manufacturer and a unique product from that manufacturer; and be read by a reader that was

almost bullet proof. All that remained was to sell the system concept to the American Association of Manufacturers, get each and every company that manufactured or packaged a product for retail sale to adopt it, and to build and maintain a manufacturer's register so the codes were unique.

After that was done, the building of barcode printers, point of sale scanners, and onsite computers to control the scanners was routine. The retail systems we know today are tied into cash registers; and also have a local data base for prices, daily activity reports for site managers, and communications capabilities to a central computer to track inventories, report sales, and schedule shipments to restock the shelves.

Point of sale systems were initially installed in grocery stores, but have been adapted to and adopted by other retail outlets. With the addition of hand scanners for warehouse control, barcode systems are used for inventory control in manufacturing, pharmaceutical, and package tracking systems.

15. Graphics

In the early1960s, Stromberg-Carlson made an output device that would interpret a magnetic tape full of graphic commands and produce a spool of 35mm film. Before this tape-to-film machine was available, the results from the simulation of physical processes were often presented as a full box of numeric listings. Page after page of formatted numerical listings were very hard to digest. Sometimes simplifying analysis reached conclusions which were wrong.

When 35mm film output was available, the results of computations could be shown as a movie. Then if interesting, surprising, or abnormal results were visible, the related numerical data could be reviewed in detail.

About the same time, large workrooms of draftsmen (and women) were the norm when designing large or complicated projects. The rolls of paper produced were unwieldy. Further, great difficulty was encountered when an engineering change affected more than one drawing.

The styling sections in car companies used 1:1 full sized etchings on giant aluminum plates which were dimensionally stable. Once again change was a pain, and the plates, typically 6' high and 24' long, were hard to handle and file.

Computer graphics changed all of this. Some special large CRT consoles, a computer, special programs, and a family of output devices growing in function and size changed all this.

Boeing was a leader in what eventually became known as CAD/CAM (Computer Aided Design and Computer Aided Manufacturing). A person with drafting skill and special training, working at an input console, could create a drawing on the CRT tube. Designs for large parts were worked on in sections. The resulting graphics could be stored in digital form, or produced as output on a growing variety of plotters. Eventually, the computer files were translated into the languages used by machine tools and parts were manufactured, sometimes without paper. A landmark book on this subject was published in 1960s.

16. Photocomposing

In the late1960s, output devices which used electronic commands to create text images were a rapidly developing art. Small dedicated standalone systems replaced hot metal linecasters. Magazines were among the first to experiment with this technology.

Soon the manufacturers of printing presses caught up with these output experiments and it was possible for computer output to become page images which then were made into etched plates which ran on conventional presses.

Suddenly there was a lot of publishing activity aimed at newspapers and book publishers. For several years writers had been sitting at computer terminals to create their copy. Once created, that copy could be edited and stored for publication. When the output devices supported multiple fonts, heads could be added to stories, stories read and approved, and pages could be made up, all electronically.

Publication systems are still evolving today. An author at a PC can create copy and transmit it to the publisher via the Internet, where it can be retransmitted, if the editor and the publisher are not co-located. After it is finally approved for publication, the story can be displayed as a galley, made into finished pages, and sent to the printer for printing and shipping.

Incidentally, photocomposing is a misnomer; it should have been called computer assisted publishing.

17. Flexible Communications

In1960, only IBM and seven other companies (colloquially known as IBM and the 7 Dwarfs) had serious commercial computer offerings. But by 1970, after one recession and fierce competition from the System/360 family of computers, IBM dominated the market. That left many trained computer engineers available for other pursuits.

The midi-computer came and went, and the mini-computers found a niche and held onto it. However, the big action was in digital communications. Computer terminals were basically digital internally. It was unwieldy and slow to take their output, convert it into tones, and transmit those tones across an analog telephone line. When the telephone system went digital (to get more logical circuits over the same physical medium) communications between computers followed.

Gone were the analog to digital modems and gone were dedicated point to point circuits. As the telephone plant modernized, a physical circuit could be shared between several uses (and users). Several logical circuits could terminate at a small dedicated digital communications computer (mounted in a rack or simply sitting on a closet shelf). That computer talked to a mate some distance away (on a dedicated circuit). The mate recognized a header code at the beginning of a message and knew where to send the remainder of the message. Soon switching computers were connected to other switching computers and a primitive network was born.

All this drove the chartered communications carriers wild. But after they saw the handwriting on the wall they gleefully joined in. Soon messages could be sent from one computer to another if you knew the communications address and the routing of the computer you wished to connect with. Your message would be sent over a combination of dedicated and switched circuits of which the sender was unaware and unconcerned. The days of the leased and dedicated physical circuit were then numbered.

18. Local Area Networks

The large universities pioneered local networks in the late 1960s. Frequently a large college campus would be spread over many acres of land. Departmental computers, often funded by volatile government grants, were used for both funded research and student problems. Each university department had parochial control over its computer, its programming languages, and the services it offered. One large Western university had a main computer center, an administrative center, and 26 departmental computers.

If the grant funding a departmental computer ran out, the administration was under heavy pressure to pick up the tab, because instructors and students had made a significant investment in that computer system for their course work. This 'freedom' and lack of standardization was usually not to the benefit of the students. Further, computing experience in this environment did not mean much on a resume.

The local area (campus) network (LAN) began to bring order out of this chaos because the switching computers demanded a certain amount of standardization. Further, the high cost of maintenance became visible. When languages such as BASIC (good for stating the solution to small problems) were proven, communication circuits were run into dormitories and the whole campus became wired with facilities available 24/7.

Large industrial firms also had LANs installed, but their adoption and adaptation was easier, since standards were more prevalent and the variety of computers to be connected was smaller.

19. Personal Computers

What was eventually known as the personal computer originally came upon the scene as a hobby kit. By the late 1970s, chip and disk technology advanced to allow commercial products to be offered.

After a wave of college graduates entered the field with some computer training in the mid to late 1960s, there was a growing trend toward self expression. In computing's first era, full-time trained programmers received problem statements from users and meticulously transformed these statements into code that produced answers. There never were enough programmers to meet the demand - there always was a backlog. Programming priorities where someone lost and someone gained were frequent.

As new graduates came into the field with experience with Fortran and Basic, the central computer shops could no longer defend their programming monopoly and 'open shop' programming became the norm. The small jobs in the backlog soon disappeared as the problem owners became familiar with the services and procedures offered to them. Most computer centers were out-maneuvered by their clients, and their workload, prestige, and budgets dropped. The users resisted all standards as unnecessary interference, and while answers were being produced faster, the quality of the results dropped and occasionally errors crept into finished reports.

While the user community battled the computer center, the mini-computer came on the scene. A mini could be installed without extra air conditioning or other special facilities and a large user could show enough past expense to justify one. This justification was helped by Federal

contracting rules which allowed computers to be classified as a *necessary* tool, and user personnel to be classified as *direct* labor. Politically, the weakened computer centers lost again and departmental centers became popular. Without standards, transfer of personnel between departments became more difficult and since the computer configurations were dissimilar, excess load on one computer was hard to transfer to another computer on the same campus.

Frequently the same old arguments were used by strong project leaders to separate their work from departmental computers. The manufacturers fed this debate with bigger and faster mini-computers which would calculate rapidly, but lacked strong input/output capabilities.

The personal computer, with its advanced technology, benefited from these political debates. It was suddenly possible to give each researcher, engineer, or student his own machine. The costs were considered direct project support and the jurisdictional arguments moved to the financial organization.

As the PC matured, the variety of software increased and PCs could support text processing. The same engineers and scientists who could not wait for their jobs to be programmed suddenly became typists. The ratio of secretaries to direct project personnel went down as documents were keyboarded by their creators and cleaned up by the remaining typists and editors.

The acceptance of the personal computer, which provided calculation, word processing, and financial planning (spreadsheet) capability to the desktop, was initially rather slow. When Apple provided the friendly "graphical user interface" (which allowed a mouse to point to icons instead requiring a user to type calling sequences), followed later by Microsoft's Windows, and then the series of books for "Dummies" was offered, the desktop computer suddenly became very popular. Although the costs of support services frequently went up and quality sometimes went down, nevertheless a quiet revolution took place among white collar workers.

20. Games

In one sense modern electronic games can be traced to the pinball machines of the 1930s. In the late 1930s the Link navigational trainer was produced for training fledgling pilots during WWII. The Link contained analog electronics and enclosed the pilot in a more or less real life environment while the instructor, standing outside of the capsule, gave instructions and followed progress on a robot plotter.

In the 1960s the pinball machine turned electronic with Pac-Man. The original presentation even looked like a pinball machine and they were installed in arcades and initialized with a coin.

As the electronic revolution delivered the personal computer and the communications controller to business and industry, it became possible to make game electronics in small packages. Enter new talent which did not grow up using computers to support academic/industrial processes. These people had knowledge of human interests, psychology, and play. They created a whole new leisure-time world for active single and multiple user entertainment. Some educational scholars joined in and personal education and training were added to the marketplace.

21. Animation and Entertainment

Take a stable full of artists and animators, add digital graphics to reduce the repetitive nature of the process and solve the file/recall problem, and digital animation was born. Engineering graphics proved that images could be stored and transmitted. Inexpensive home playback devices were available. The market was ready for simple stories with sound.

Hand drawn animation predates the movie camera. Animation with sound went along with the early 'talkies'. The editing of digital images is actually easier than with film. The new equipment was timely and much improved and the studio demand was there.

However, lifelike art needed an advance in software technology; but when faces could be shaded and images adjusted within an outline on a frame, the results were dramatic.

Digital movies were a natural technological step. Sometimes the original scenes were shot digitally, and sometimes analog images were digitized in the studio. Contractors to the movie industry allowed small studios and individuals to experiment. When editing with special digital equipment, sound was easier to dub, and titles were easier to add and modify.

Computers were part and parcel of early signal switching. Not only did they allow incoming broadcasts to be sent to the transmitters, they allowed the broadcasts to be recorded for rebroadcast at a more convenient time. It was only a short step to frequent interspersed commercials. A necessary byproduct of electronic switching was an accounting tape which recorded what and when for offline billing purposes.

Computers for production are now commonplace. Commercially, the country is going digital to get bandwidth and additional channels (or so they say). Expensive digital home TVs have been selling more slowly than anticipated due to their stubbornly high cost. But eventually the process will be end to end digital, controlled by computers all along the way.

22. Networks and the Internet

Telephone transmission technology went digital first. In the 1960s, the analog signals from your phone were converted to digital whenever you made a long distance call. Then it was only a short step to bypass the converters and provide digital service to business and industry.

After portable telephones (with short radio links) were allowed, some visionary individuals devised a system of radio links to connect portable telephones to regional antenna towers and (digitally) switch the signals to landlines or other cell phones that were in range. Further, their arguments and their politics were sufficiently strong to get the system financed and built.

The Federal Government has funded a lot of computer innovation over the years. A communications network of campus computers, to allow the interchange of big files was proposed, funded, built, and only partially successful. However, researchers and computer scientists associated with the system found it handy for personal communications. Again visionary individuals saw an opportunity to connect unrelated computers into a switched network by the addition of more sophisticated software and a centralized digital directory to allow calls to be initiated by name instead of having to know the routing and the receiving address. Again, their arguments and their politics were sufficiently strong to get the system financed and built.

From its academic roots, the Internet designers consciously left out accounting and content controls. Anonymity was the winning campus concept. Today the content over the Internet is still open and uncontrolled.

23. Portables

The era of portable computing can be traced to the hand-held calculators produced by Texas Instruments in the 1960s. Although they were externally programmed and had only a couple of words of storage, they offered an impressive variety of mathematical functions and would fit in a briefcase.

Over the next 30 years, calculators got more internal storage, solar cell power, and their functions were specialized for specific audiences. The financial hand-helds of the 1970s were particularly noteworthy.

In the 1980s Compaq came out with a PC in a case and called it portable, since it had a handle on top and could be carried by a strong man.

These two trends have continued through this day. The calculators are more capable and the full function PCs are smaller. When the revolution in communications occurred, it was warmly embraced as business travelers and students were always on the go and wanted to be connected. Almost unnoticed were the legions of tradesmen (and women), outside salespersons, and workers in construction, pharmacies, etc. who needed computing and communications in their normal activities.

The use of portable computers for entertainment has almost obscured these more pedestrian uses which will be with us a long while. Time will tell whether games, gossip, and personal entertainment will survive the next economic slow down.

