Oral History of Charles W. “Chuck” Walker

Interviewed by:
Gardner Hendrie

Recorded: June 9, 2011
Sonoma, California

CHM Reference number: X6276.2012

Editor’s Note: Additional commentary and clarifying notes were supplied by Chuck Walker after this interview took place. These instances are denoted by square brackets.

Gardner Hendrie: We have with us today Chuck Walker, who has graciously agreed to do an oral history for the Computer History Museum. Thank you very much, Chuck.

Charles W. Walker: More than welcome.

Hendrie: I think what I'd like to do is start out with your career at 3C (Computer Control Company), which is the part that we're particularly interested in, because it was a very early minicomputer company that was pretty successful. Then, we'll go back to your early history; and then, we'll finish with your career after you left 3C, if that's all right with you.

Walker: That's fair enough.

Hendrie: Well, I guess probably we ought to go over the story of how you got recruited at 3C. Maybe, we ought to start where you first met Ed Griffith, since he keeps coming up, and do a little of your pre-3C history, just so everything's a little bit more in perspective.

Walker: Well, Ed certainly enters into a lot of it. Ed and I worked together at Rocketdyne in a data-processing group, doing data reduction for rocket-engine tests. A very good learning experience, because IBM (International Business Machines, Inc.) was in there pretty thick, and I think Rocketdyne had a standing order for two each of whatever IBM came out with. Anyway, we started writing data-reduction programs in FORTRAN (IBM Mathematical Formula Translating System), in a data-processing group.

Hendrie: Well, maybe you ought to go back and say what you originally did at Rocketdyne, and how you introduced yourself to FORTRAN. I think that would be interesting.

Walker: All right. I started out doing data reduction, using some kind of desk calculator, and had been doing that for a few months. I became aware of the computer at Rocketdyne's Canoga Park facility. I was up on the hill at that time, Santa Susana Mountains.

Hendrie: This is where they did the actual test firings of rockets?

Walker: The actual test firings, right, were done up in the Santa Susana Mountains. [They] collected up

data from these firings—Dynalog recorders, strip recorders, all sorts of things like that—and I had to read out the information on them, and enter the data manually into data-reduction forms, or data-collection forms, more properly. I was reading the charts, doing calculations on desk calculators, and entering the results manually into test record notebooks.

Hendrie: Now, were the recordings fundamentally on strip charts, or were they digital?

Walker: No, they were all analog recordings at this time. They were either strip charts for some recordings, or they were circular charts. I think they were Dynalog charts, if I recall.

Hendrie: Okay, but they were all pen plots.

Walker: All pen plots, all analog, so I had to read these analog traces, and extract data from them, and calculate data that was of interest to the design engineers. It was rather tedious, and we were always way behind on things, because it just took so long to do it manually. I became aware of the computer facility at Canoga Park, which was kind of the headquarters for Rocketdyne. They had gotten an IBM 704 at that time, and the 704 had a program called FORTRAN that was used to write programs in. Rocketdyne encouraged anyone that was interested in it to work with it. I got a manual on it, and started writing data-reduction programs while I was up on the hill. It was faster, it was much better than hand-calculating the test results. A supervisor in a data-reduction group down in Canoga Park heard about what I was doing up there, and recruited me to come down and work for him in the data-reduction department.

Hendrie: All right. We're back on.

Walker: Okay. I went down to work in this data reduction group with a fellow named Ed Griffith. He and I were writing data reduction programs for supporting the rocket engine tests. [Even though a computer (now an IBM 709) performed the actual calculations preparation of input data was still a manual process. Clerks in the data reduction group would read the data from the analog recordings and transcribe the data of interest to data entry forms. The data entry forms were then sent to a keypunch group where the data was punched into Hollerith cards. These cards were then read by the data reduction programs that Ed and I wrote.] Ed left not too long after I went down to join the data reduction group and I continued on writing data reduction programs. We were, at that point, collecting some data that was digital recording tapes, and between the digital recordings and the analogue recordings, at the time, it was very heavy data processing. [One of the things that Ed and I developed was a simple data collection system that recorded test results on magnetic tape. This made it possible to do statistical analysis of test results without having to manually reenter the test results. It’s been so long ago I don't remember how we

organized the data, but in any case it was not necessarily serial. Recording on tape is always serial so to get the data in the desired order two tapes were required. Let’s call the tapes “current” and “new”. The current tape contained all of the test results to date and the new tape was blank. The current tape would be copied to the new tape up to the point where the new test results were to be inserted. The new test results were then written to the new tape after which the remainder of the current tape was copied to the new tape. At the completion of processing the new tape was saved to become the current tape for the next run and the no longer current tape saved as a backup.]

Hendrie: But, fundamentally, the data and then the calculations that they wanted to see on that data were all...

Walker: All on mag tape.

Hendrie: All on mag tape and all associated with each other.

Walker: Right.

Hendrie: In sequential blocks.

Walker: The thing that would happen is you'd get several tests run over a weekend when there was no one there except the computer operators themselves. [FORTRAN in those days was very unforgiving of I/O errors–any I/O error such as an alpha character in a numeric field would terminate execution. With all of the manual steps required to prepare the data for input such errors were not uncommon. When an error came up in one of the programs it would abort the program at that point, not copying the rest of the tape. The computer operators could not tell that the program terminated abnormally and would make the incomplete new tape the next current tape so we were losing data because of the errors in the input data. I had the system people change the exit calls to call an exit trap routine. The exit trap routine in the library would simply call exit so unless a program included a special exit trap routine the program would abort normally. I added a special exit trap routine to our programs] so that we were able to get hold of the program again before we were taken off the machine and at least finish copying out the tapes so we weren't losing data. That was probably the most significant thing that I did there.

Hendrie: You came up with this idea.

Walker: I came up with this idea to catch the system before you were actually taken off. I think it wound up in FORTRAN ultimately as the error equals clause that showed up in I/O statements—I don't know if

that came because of me. Anyway, it's the same effect. You could give the program something to do in an error other than just exit.

Hendrie: Yes. If people wanted to exit their programs, they could [still just] exit.

Walker: Right.

Hendrie: If they had a programming bug, you could add a subroutine that would run. [If they had a program that needed to do something special on an I/O error they could add a subroutine that would run when an I/O error occurred].

Walker: What was done is the routines that detected the error were changed to call a program called TRAPEX. The TRAPEX routine added to the system library, which was loaded by default, did nothing but call exit so, unless you put a TRAPEX program of your own in there, the system appeared to work exactly as it always did. But, with a TRAPEX program in there, we could catch the system and at least copy the rest of the tapes. So that was my first programming invention, maybe.

Hendrie: Good.

Walker: In any case, the man I was working for that headed up the data reduction group there left Rocketdyne and went to work for General Electric in Philadelphia. He recruited me to come back to General Electric, which I did, and we got back there with the idea of doing data reduction programming for GE only there was some political shuffling within GE after I got back there...

Hendrie: Now, what was GE doing at this facility?

Walker: GE was processing data from reentry vehicles.

Hendrie: So they were working on reentry vehicles there.

Walker: Right. [Data analysis anyway.]

Hendrie: And this was telemetry data for reentry vehicles, probably?

Walker: Not so much for the reentry vehicle [itself. They were analyzing reentry signatures]. They had recording apparatus stationed out in the Pacific, I think Quadulan or whatever the receiving end of the Pacific Missile Range was and data would be sent back from Quadulan on tapes to be processed. So it wasn't a live capture that we were working with at GE. In any case, the politics and things changed there and the group that I had joined in GE was not going to be doing any kind of programming work, which is what I was interested in and Ed Griffith materialized. He was then working for RCA (Radio Corporation of America) and convinced me to go to work for RCA because I could do some real programming work there. So I left GE after about a year and went to work for RCA. This was in Van Nuys, California. One of the first things I did was complete an Assembly program at a division of RCA in Riverton, New Jersey, the BMEWS (Ballistic Missile Early Warning System) prototype test site, which was kind of interesting.

Hendrie: That would be fun.

Walker: That was very interesting. In fact, I was there at the time of Cuban Missile Crisis, which was very interesting. They wound up turning this test site on the skies over Cuba to keep an eye out for any missiles that might be launched from Cuba.

Hendrie: Oh, really?

Walker: Yes. That was kind of an interesting side light to the sojourn to Riverton, New Jersey. Anyway, I spent a couple of months in Riverton, New Jersey, finishing off an Assembly program for a special version of an RCA computer; it was designated as 4103. After finishing that, I went to Van Nuys, who I was reporting to all along. That's where Ed Griffith was and that's where I got recruited to. So I moved back...

Hendrie: Okay.

Walker: I moved back across country to Van Nuys.

Hendrie: Now, who designed the 4300 series? Was that...

Walker: I think Dave Woods...

Hendrie: Bill Woods.

Walker: Bill Woods, I'm sorry, Bill Woods was involved in that design. In fact, he may have been the principle designer on it. There's another engineer named Dave Schwartz. I don't remember if he came in after the fact but he worked closely with Bill.

Hendrie: Well, a side bar is, of course, that Bill Woods eventually ended up at 3C and did the Logic design for the 516, 416s machines.

Walker: Right.

Hendrie: Yes, but anyway.

Walker: Very tangled paths.

Hendrie: Yes, very tangled paths.

Walker: In any case, I moved back out to Van Nuys, which is a suburb of Los Angeles and I was [going to do] software development for the 4102. As kind of a side bar thing, before starting on software for the 4102. [The first computer I got involved with there was called the RCA 110 which, oddly enough, has a history reaching back to Gardner Hendrie.]

Hendrie: Yes, okay.

Walker: I don't remember if I did the whole assembler now or just finished off an existing assembler but the first project with RCA [in Van Nuys] was this 110 assembler. [After finishing the 110 assembler] I moved over to the 4102 project. The 4102 was kind of the parent processor of the 4103. The 4103 was an extended word length processor of the 4102, otherwise logically the same. [The 4102] was an interesting processor, my first exposure to real interrupt processing. That's the only sort of [I/O] processing it had. There was no kind of standard, as I would have called it, I/O [input/output] at the time. You had to do everything as an interrupt process. So I got my feet wet in interrupt processing on that guy. I wound up writing an assembler for it. I don't recall now what all other software I did but I think I/O drivers and the assembler. I don't remember it having an operating system, per se. In any case, the fortunes at RCA shifted around a bit and their EDP [Electronic Data Processing] division decided that anything having to do with computers should be done there...

Hendrie: And that was back in New Jersey.

Walker: That was back in New Jersey.

Hendrie: Yes, in Cherry Hill.

Walker: I did not want to go back to New Jersey for RCA and so, again, Ed Griffith enters into things. He'd left RCA at that time and wound up at a company called Computer Control Company. Ed contacted me to see if I'd like to go to work for Computer Control Company (3C), which was in Massachusetts. Well, another cross-country move was in store there so I went over the pros and cons with my wife. We decided we could go back to Massachusetts, which is where 3C was located, and work on the computers that 3C was developing.

Hendrie: Now, do you remember approximately what year this might have been?

Walker: This would have been about 1963 or four, middle '60s, '63 or '64.

Hendrie: Was I there?

Walker: I don't think you were at 3C yet. I think you joined 3C not too long after I did.

Hendrie: Yes, I joined in March or April of '64 so that makes sense.

Walker: Okay.

Hendrie: So it might have been late '63.

Walker: Late '63 perhaps.

Hendrie: Now, did you go directly back to Massachusetts or did you work awhile in the west coast division of 3C...

Walker: I worked for awhile, a short time in the west coast division completing an assembler for a DDP-(Digital Data Processor) 19, which was the predecessor of the DDP-24, which is a 24-bit computer. The DDP-19, interestingly, was a 19-bit computer.

Hendrie: Yes.

Walker: They were very creative in their naming.

Hendrie: Right.

Walker: I spent two or three months in the west coast division finishing off an assembler for the DDP-19 before going to the Boston area—Framingham, specifically—for the job at 3C. I knew that that move was coming but had this stand in...

Hendrie: They had something they needed to finish up.

Walker: They needed something finished in the west coast office, which I did. Then I moved up to Framingham—seems to me that was in the fall of the year, I don't recall now—and went to work on software for the DDP-24, which was the 24-bit successor to DDP-19. I wrote an assembler for the DDP-24 plus other operating software, I/O drivers and that kind of thing

Hendrie: Was there an operating system for the DDP-24? Or nothing really?

Walker: Nothing really. That was up to the end user. We had utility routines, I/O drivers and things like that, that would help out in writing code for it but there was no operating system for it at the time I went back there.

Hendrie: Okay.

Walker: Early in the DDP-24 project, there were DDP-24s sold to Link for use in the Apollo mission simulator. These were used to simulate the [moon] orbiter and lander on the moon end of things. They made use of features in the DDP-24 that had not yet been built. They were designed but it was a famous first for shared memory use. As I recall, there were three DDP-24s sharing two memories along with a private memory and...

Hendrie: Oh, my goodness, all right.

Walker: The shared memory part of it had been designed but had not been built yet.

Hendrie: Oh, my goodness.

Walker: So I went through a fairly hectic period there of developing the system for Link, building the three DDP-24s and the shared memory. I wound up having Ed Griffith on the software side and myself and two hardware engineers that were covering the system 24/7, which turned out to be one hellacious project.

Hendrie: Yes. Who were the hardware engineers?

Walker: The engineers, as I recall, were Ed Thayer and Frank Brinkerhoff.

Hendrie: Okay.

Walker: I believe those were the names. Ed and Frank on the hardware side and Ed and I on the software side were covering the system test of this new system and it was literally 24/7.

Hendrie: Now, were there diagnostics? Did you write test programs?

Walker: We wrote some test programs on the fly to test some of this. There were some diagnostics in place just for the standard 24. We had to write what we needed to test the shared memory and that kind of thing. So we wrote those as we went.

Hendrie: Now, did you modify the assembler to deal with shared memory?

Walker: I don't recall any modifications of the assembler.

Hendrie: So that was up to the programmer to...

Walker: Up to the programmer...

Hendrie: ...figure that all out.

Walker: I don't remember whether it was separated by addressing space now or how you got into the shared memory versus private memory. I think they were in separate addressing space. That is my recollection. It's too long ago to...

Hendrie: To remember exactly, yes.

Walker: ...remember the details.

Hendrie: Yes, I understand.

Walker: In any case, my recollection is this went on for maybe a month, the final test on this system working seven days a week, 14 or 15 hours a day. So Ed and I had a little overlap on each end and then we'd basically work 12 hours and with the overlap turned into, like, a 14 or so hour day. This went on for about a month, as I recall. At the end of all of this, we had made a successful delivery to Link and the system worked out pretty well. The company, 3C, threw a tremendous party for all of us, not just the four of us who were monitoring this but all the people who worked on the project, the hardware people, the software people, and everybody. Big dinner at a—I don't remember the name of the club there in Framingham. It was pretty well known at the time but they took us all there for dinner, for a very nice evening, small...

Hendrie: A small thank you.

Walker: Small thank you for a hellacious project. Certainly Ed and I and the two hardware guys weren't the only ones putting in horrendous hours. Just about everybody on the project was. The bottom line was it was very successful.

Hendrie: Do you remember who the project engineer was on that?

Walker: I do not. The name Ed Spignazy comes to mind but I don't know that Ed was the project engineer on that or not.

Hendrie: It could be. He certainly was at 3C but he did the high speed arithmetic unit, the multiply and divide unit on the DDP-116.

Walker: It may have been one of the principles in the company that was kind of the project engineer on this stuff, Frank Dean possibly. Might have been Frank...

Hendrie: Or maybe Syd Galen or one of those people.

Walker: I do not recall who the project engineer was.

Hendrie: Okay.

Walker: In any case, it was a lively period there at 3C. Things calmed down a little bit after that was delivered to Link. Somewhere in the sequence of all this, Gardner joined the company and we—collectively, the company—decided to build a smaller computer to be competitive with some of the small computers from DEC (Digital Equipment Corporation). Gardner was the project engineer. I recall many sessions talking about tradeoffs in word length, thought to be somewhere between 12 and 18 bits. This was about the time the ASCII [American Standard Code for Information Interchange] character set was coming into play. [For cost reasons 14 bits was leading candidate.] It was my belief that we ought to be at 16 bits, but 16 bits carried with it certain cost penalties, cost per bit in the system and all that. After lots of haggling around on the architecture of the system, we settled on going to a 16-bit computer. I believe that was the first commercial 16-computer, if not the first 16-bit computer period. DEC's equipment was not 16-bit at the time, it was 12.

Hendrie: They had an 18-bit line and a 12-bit line.

Walker: Yes. So I think we were certainly the first commercial 16-bit computer. I think that it was a very successful computer family. The 116, the initial computer in that 16-bit series, spawned the 416 and the 516. The 416 was kind of a curious offshoot of the product line. I think it saved money by not having [two general registers (A and B)] or an index register, if I remember right. Otherwise, it was...

Hendrie: Yes, it was a crippled machine. It didn't have all the instructions...

Walker: Right.

Hendrie: I don't think it had index registers but I think it didn't have a B register. It only had an A register. The DDP-116 machine did have an A and a B register as well as an index register, if my memory is correct.

Walker: Right. The rest of the...

Hendrie: Primary accumulator, secondary...

Walker: ...16-bit family did. [Since the 416 did not have the same compliment of registers as the 516 and only a subset of the 516 instruction set it was very difficult to use code written for the 516 on the 416.]

Hendrie: Yes. Do you remember whether the assembler or the software was written so that it would protect you against—if you wanted to write compatible software programs that could run on either a 416 or a 516, that the assembler would catch you if you tried to execute a 516 instruction?

Walker: I don't remember.

Hendrie: Don't remember any...

Walker: It's probable that we would have done that but I don't recall. I wrote the assembler for the 16-bit family, which, again, was, I think, another inventive thing. I came up with that...

Hendrie: Oh, yes. Tell me how you dealt with the sector, the addressing modes in the 116. They were a little...

Walker: Awkward, you might say.

Hendrie: Well, they were odd for straightforward machines. They were used by many, many machines later.

Walker: Right. The memory addressing space was divided into 512 word sectors. It was a word addressing machine. This was before the prevalence of byte addressing. The base sector or Sector 0 was addressable from anywhere in memory by any instruction. The sector that the instruction itself was in was the only other sector that was [directly] addressable so it created...

Hendrie: In a single word instruction?

Walker: In a single word instruction. [The address field in an instruction was nine bits long, giving a range of 512 words. The upper six bits of the 15-bit address were either zero or the upper six bits of the program counter.]

Hendrie: Yes. If you had an indirect addressed instruction, which took two words, then you could get anywhere. Right?

Walker: Right. The indirect address was a mechanism for getting to areas outside [the sector the instruction was in.]

Hendrie: Yes, [you could indirectly address] 32K words, I think.

Walker: 32K words. This [sector addressing] created some problems developing relocatable code for this system so I came up with a scheme to generate what I tag named extended object code. That was the assembler generating code as if 32K was all directly addressable and the linker took care of figuring out if the address was within the current sector, the [512 word] sector that the instruction was in or was outside it. [If it was inside it, the linker used the lower nine bits of the 15 bit address in the instruction.] If it was outside it, [the linker] automatically generated a link, an indirect link to the 15-bit address that was stored in the base sector which, of course, was addressable from anywhere. [The nine-bit address of the indirect link in the base sector was used in the instruction in place of the 15-bit address generated by the assembler.] So it made possible the generation of relocatable code for the sector addressing machine, which, again, I don't know of that having existed beforehand anywhere. It worked pretty well. So we had relocatable code for this hard-sectored machine.

Hendrie: That was a really cool thing to be able to do.

Walker: It worked out pretty well. Again, that was one of these things that were done in great haste. The person writing the assembler for the 16-bit family didn't see this concept and had generated an assembler that would work well if you had a machine that had sector zero and sector one only [a 1K machine]. It wouldn't work very well if you had a larger machine, 4K memory machine or larger [and the base machine had a 4K memory]. So, in one hectic two or three day period, I modified the assembler to generate extended object code. That had to be delivered to I think it was TRW [Thompson Ramo Wooldridge, Inc.] at the time. We had a very tight delivery date on this thing and that created a minor crisis trying to get the assembler to work with this extended object code for TRW.

Hendrie: Yes, so you didn't start out on that project but then you came—you were recruited to go fix it.

Walker: Recruited, I guess, you would say, or trapped perhaps. [Actually I recognized the problem and just made the fix so you could say I volunteered.]

Hendrie: Perhaps, yes.

Walker: I enjoyed that, even though it was pretty hectic and a lot of work—in fact, I wrote an article about [extended object code that that got published in Datamation.]

Hendrie: Oh, really?

Walker: ...that showed up, yes, sometime afterwards. My only publication ever.

Hendrie: Your only publication. Well, I think we have a pretty good Datamation archive.

Walker: Do you?

Hendrie: Yes, so it's probably there.

Walker: It should be in there if you have an archive of that. Does Datamation magazine even exist any more? I don't think it does.

Hendrie: I don't think it does but it was published for a really long time.

Walker: Yes, it was the only magazine for quite a long time that covered computers that way.

Hendrie: Yes, exactly. Well, that's interesting. So you really became sort of a specialist in Assembly language, the assemblers...

Walker: Right.

Hendrie: ...for machines, it sounds like...

Walker: Yes.

Hendrie: ...you did a lot of those.

Walker: Well, we were involved in hardware diagnostics, too. I was involved in a lot of that.

Hendrie: Oh, really?

Walker: Yes. Hardware diagnostics and assemblers and all the supporting kind of software, I/O drivers and interrupt handlers and things like that.

Hendrie: Yes, things that were directly connected to the hardware.

Walker: Right.

Hendrie: Very related to the hardware. Do you remember any stories about diagnostics or things that you did in the diagnostic world that you had to write something special to find some thing that nobody could figure out why it was doing this? Which is our classic diagnostic stories.

Walker: No, I do not have any recollection of anything [special that] I wrote. It's highly likely that I wound up doing some of that but nothing stuck with me. I'll probably think of it tonight at midnight or something like that.

Hendrie: Yes. When you got to 3C, do you remember, were there other programmers there? I mean, Ed Griffith was there but were there any other programmers?

Walker: I believe, at the time, it was just Ed and me and we were trying to hire. We did hire, in fact, a few programmers relatively right away but I believe Ed and I were it when I went to work there.

Hendrie: Wow. That's pretty—a computer company with two programmers. That's pretty interesting.

Walker: Well, it was pretty early in the whole game. There might have been someone out in 3C west that could program but I don't recall.

Hendrie: There might have been applications programmers there doing stuff like data analysis for Point Mugu. I know they had some contracts to do that sort of work.

Walker: I don't remember anyone out there that...

Hendrie: Did systems programming?

Walker: Did systems kind of programming.

Hendrie: Do you remember any of the people that you brought on after you and Ed got there?

Walker: Let's see, John Thron comes to mind as a programmer that came in. Connie McLarney. Ron Gendrau, he wasn't in programming. I think he was doing operation support work. Boy, I can't think of other names that were there at the time, again, they'll probably pop into my head at midnight tonight.

Hendrie: Yes, exactly. That's what happens to me.

Walker: I can't think of any—John Andrusko was another name. Boy, nobody else comes to mind right now.

Hendrie: Okay.

Walker: 3C, of course, got bought out by Honeywell in the middle '60s, later '60s and Honeywell changed the complexion of the company drastically, brought in a lot of their people from the Philadelphia area, I guess, Fort Washington. They were people who were involved in process control, if I remember right.

Hendrie: Yes, I think that is correct. That's what I remember.

Walker: Yes. They brought in a number of people. It just really changed the way 3C looked and worked. I think, at the time that happened I was the manager of software development. I got shuffled around a little bit, wound up working in [another] group. Lowell Bensky was there at this time. I wound up working for Lowell but I don't remember at this point exactly what his title was. But, anyway, I was kind of shuffled aside from the manager of software development to a person brought in from Fort Washington.

Hendrie: Was that a John Nesbitt?

Walker: Yes. That name was—yes, John Nesbitt. In fact, it's not flattering; we used to call him John Nebulous. <*laughter*>

Hendrie: He wasn't too clear or precise.

Walker: Yes. Or seemed to be very indecisive. That nickname for him pops into mind.

Hendrie: Yes, okay.

Walker: Anyway, after Honeywell bought 3C, things just kind of went downhill and it just didn't look too promising to me. I wound up leaving 3C and coming back out to the west coast for a small consulting company named Compata. I met one of the people from Compata—who was doing some consulting work at 3C—a fellow named Bob Hooper.

Hendrie: And that's how you got...

Walker: That's how [I got] my connection to Compata and so I moved back out to the west coast going to work for Compata. Compata was a computer consulting company, did both hardware and software design, although the bulk of the company was on the software side.

Hendrie: Before we get into Compata and move on in your career, I had a couple of other questions about the 3C period.

Walker: Sure.

Hendrie: Did 3C have a FORTRAN compiler when you were there and, if so, could you tell me the story of where that came from or how you got it?

Walker: [3C did have a FORTRAN compiler.] The compiler was developed by an outside software company located here in California. In fact, I'm trying to recall their name now. It went out to bid to several companies, Programmatics [and Phil Hankins were two] of the companies that bid on it and I'll be darned if I can remember the name of the...

Hendrie: The one that actually did it.

Walker: The one that actually did it. But I know that it was a small company located in the Orange County, California area. They developed this compiler for FORTRAN for the 16-bit family. I don't remember if we had a compiler for the 24-bit family or not. We perhaps did. I don't remember.

Hendrie: But that contract was definitely written while you were still there?

Walker: Yes.

Hendrie: So it was in the era of the 116 and 516?

Walker: Right. Right.

Hendrie: Not the future machines.

Walker: No, it was in the 116/516 era.

Hendrie: Okay.

Walker: I'll be darned if I can think of the name of that company. I know they had people on site doing checkout work for quite awhile.

Hendrie: Okay.

Walker: The compiler was basically quite successful.

Hendrie: Did you get involved in that project that you can remember?

Walker: Not directly in the creation of the compiler. At that time, my involvement would be more on the management side of things. I didn't write any code for the compiler.

Hendrie: Okay.

Walker: One of the things I do recall creating code for now, it's kind of a side thing, the only way you could get a program started, get started on things, you needed a little, a sneak on loader for the 16-bit series. You had to read from—we had only paper tape of one sort of another and that brings to mind another interesting thing. But, anyway, John Thron and I had a little competition to see who could create the tightest...

Hendrie: And smallest.

Walker: And smallest sneak on bootstrap loader.

Hendrie: That you had to key in, right?

Walker: You had to key this in.

Hendrie: Yes, this is what you keyed in, which then brought in a real loader, which then brought in the real programs.

Walker: I wound up winning that competition by one instruction.

Hendrie: How long was yours, do you remember?

Walker: It was about eight instructions, as I recall. Of course, it had to restore itself so you didn't have to reenter it every time you wanted to load something so, once you entered this thing into memory, you were good unless you clobbered it in some way or another. This was back in the days of core memory and you could put something in memory and it would stay there.

Hendrie: Yes, just as long as you didn't shut the machine off—the electricity. Well, even then...

Walker: Oh, you can shut it off completely and it would still stay there in the core memory indefinitely.

Hendrie: That's true. Yes, okay.

Walker: So that sat in the highest eight locations in memory. It brought the real loader in right underneath it and [reset] itself to be ready to go next time around. So that was kind of an interesting little competition there to get the smallest...

Hendrie: Oh, that's fun.

Walker: ...sneak on loader.

Hendrie: That's a good story.

Walker: Something else came to mind now.

Hendrie: Yes, something that brought another story to mind.

Walker: Yes. Now it eludes me. It perhaps will come back at an inopportune moment. Oh, gosh. Yes, I don't recall what popped into my head at that time now. I think old age is setting in.

Hendrie: Well, maybe—let me see whether I can jog your memory, not necessarily with specific incident but if you think about things that you worked on at 3C or in that period, are there any things that just come to mind that you remember you really felt a sense of accomplishment from doing? You know sometimes you do something and it just feels good.

Walker: Well, the little sneak on loader would be one in that category. That was kind of a nice thing to have accomplished. The whole concept of this extended object code—allowing more general development of relocatable code. I thought that was...

Hendrie: That was a big contribution.

Walker: ...satisfying...

Hendrie: Yes. And it was very satisfying that you figured that out.

Walker: Yes, yes. Nothing pops into mind with the magnitude of those two.

Hendrie: Do you have any examples of something that you thought, when you started it out, was a really good idea and it just didn't work out?

Walker: That's what I was thinking about. I remember the original 116 I/O device was an ASR (Automatic Send and Receive) 33.

Hendrie: Yes.

Walker: Which was an unwieldy beast to say the least.

Hendrie: But, boy, was it cheap.

Walker: Right. The only way we could [print and read or] punch paper tape was on the ASR 33. [The paper tape punch could only be turned on or off manually and when on would copy (punch) all characters as they were read or printed.] Of course, anything you punched would drive the printer as well so [when the generated object code was punched it would also be printed. Pure object code is binary in nature so could contain characters that when printed had undesirable effects on the printed listing. To prevent this I invented what was called the invisible object code. Whether it's a good idea or not could be argued but, in any case, what I did was map the object code into the non-printing characters in the ASCII set. There are enough non-printing characters in there I was able to do this. To get a classic side by side listing required positioning the carriage at the start of the line for the printing assembly code with this. You had to pre-position the carriage as I recall or maybe the assembler did it for you but, in any case, the idea was the text of the assembly line [the source line] was printed offset from the left margin enough to allow printing of the generated instruction code starting at the left margin on the same line. The source code lines were ended with a carriage return and a line feed. So the assembler processed each line up through the carriage return but, not the line feed. The carriage would return and we'd print the object code to the immediate left of its associated source line so it looked like a regular assembler listing that you would do on a line printer or something like that. The next source character read would be the line feed associated with the line just processed which would take it down to the next line positioned to read and print the next source line. Well, it seemed like a neat idea.

Hendrie: Yes, it does.

Walker: It did work pretty neatly but the downside of it was [the size of the object tape]. Because you were going through the ASR 33 with no way for the computer to turn the punch on or off the object tape included a copy of the source tape as well as the printed machine instructions and the generated invisible

object code. The invisible object code [conveyed only four bits per character so each16-bit instruction generated required four characters. The loader ignored everything but the invisible object code but the whole nine yards had to be read at the blistering speed of ten characters per second.] It didn't work too badly if you had a high speed paper tape reader and punch but, with the ASR 33, it was one of these things that sounded like a neat idea and it worked...

Hendrie: Yes, it did what you wanted it to do.

Walker: Right, did what you wanted it to do.

Hendrie: You got a tape that you could then—and you had a loader that loaded invisible object tape.

Walker: Yes. That was the problem. You had the [punch] on all the time so it wound up you'd get the source code embedded in the object tape as well as the object code in the object tape, which made something extremely slow to load again. So you'd have a nice printout out of it but you got a monstrous object tape out of it and you spent a lot of time loading this invisible object code because you had to read the whole tape.

Hendrie: Oh, yes, of course. And on the tape is what you printed and then the object code. So you had the source and the object code.

Walker: Exactly.

Hendrie: And reading it in a ten character per second reader on an ASR 33, it wasn't such a happy thing.

Walker: It was slow, to say the least. But it did work. I don't remember how much that feature got used or whether we went away from that. I think most of the serious systems probably went out with a paper tape reader and punch and didn't depend on just the ASR for I/O.

Hendrie: Right.

Walker: But it did work if you had the ASR only. You got a decent looking printout out of it and a monstrous object tape out of it.

Hendrie: But it could be loaded and it did work.

Walker: Could be loaded and it did work.

Hendrie: So it was one of those characteristics that the software did work with a minimum system but you probably weren't that happy with and so 3C would sell some more equipment.

Walker: Yes, I guess that's the way it worked out. [In retrospect, the invisible object code itself was not a bad idea. It was just a way to cope with the operational characteristics of the ASR-33.]

Hendrie: Yes. That is pretty interesting. Let's see, of the things that you remember doing at 3C, what do you think was the most difficult? Maybe not in terms of stressing your body but the most difficult project that you had to work on when you were at 3C?

Walker: Well, I kind of lump everything all together. I don't remember anything that I thought of at the time as being...

Hendrie: This is really hard.

Walker: Yes. I think everything just kind of—I was so into things there that nothing seemed really hard.

Hendrie: You might have to do it in far too short amount of time but it didn't seem hard.

Walker: That happened. That happened.

Hendrie: That happened frequently, it sounds like.

Walker: Frequently enough. But I don't recall anything that I thought, "Gee, this is bordering on the impossible to do." Again, I really enjoyed so much what I was doing that it made everything seem easy.

Hendrie: Why don't we just continue with after you finally left 3C? It just wasn't as much fun any more.

Walker: Yes.

Hendrie: You sound like you'd sort of gotten shuffled aside a little bit.

Walker: A little bit, yes. I didn't really agree with the way things were headed at 3C at the time with the Honeywell crew in charge of things.

Hendrie: Ed Griffith left already or not?

Walker: Yes, Ed had left already, along with an assortment of other people I thought highly of at 3C. I don't remember names now but, before I left, there was a fair movement out of 3C because there was so much displacement by the Honeywell people and the company just wasn't the same as it was before the buyout. In any case, I left 3C and joined Compata, which was a software consulting company in southern California. I think I mentioned earlier in our discussion that my connection at Compata had been Bob Hooper, who was back consulting at 3C. I think Ed brought Bob Hooper in, as I think of it now. I think Ed knew Bob from somewhere. That's how Ed Griffith pops up in everything. In any case, I went to work for Compata as a consultant. The company had only, I don't know, half a dozen people at the time I went out there. It was a pretty neat operation. Just did all manner of different consulting there, aerospace oriented stuff, did a lot of work for Hughes Aircraft in Fullerton, got into some processing. Compata bought a company called Basic Four, which was a company that developed application programs in Basic. They'd had a bunch of contracts for different programs. I wound up working on a couple of those and wound up having all kinds of forms that had to be created and the operator, the person using the system had to fill in the blanks so I created a thing that, today, probably would be called scripts. I created a system where you could lay out the—this was all done with the CRT (Cathode Ray Tube) interface, and you could lay out the form on the CRT and then traverse the form, you know, sequentially, just with an enter key after every entry, it would walk down the form and the script would specify where you went for the next entry. You could divide the page into paragraphs so you could use the tab key to go from paragraph to paragraph. So if you had a paragraph with half a dozen entries in it but they didn't all need to be filled in and you wanted to go to the next paragraph, you could tab and it would take you to the start of the next paragraph.

Hendrie: Ah, yes. If you didn't need it.

Walker: Right. So you didn't have to go through every entry on the form and you could go backwards the same way. I think shift or control was used with tab, probably must have been shift, to take you backwards on it. So you'd scroll up and down and same thing with the enter key. So I developed this script which was used for creating a large number of forms for a property management company in San Diego, the name of which I've long since forgotten. That aided greatly in the development of these fill in the blanks forms.

Hendrie: Yes, right. And then made it much easier to enter all the information.

Walker: Right. It was easy to enter the information and it was easy to develop the forms themselves. I don't remember if I created a piece of software to do it or you just did it longhand now but, in any case, it was pretty simple and...

Hendrie: Wow.

Walker: ...you could create these forms.

Hendrie: This would be a commercial piece of software today.

Walker: It would.

Hendrie: People would sell it as an application, very generally useful.

Walker: Odd you should say that because, while we weren't thinking about selling it, we were thinking about expanding its use within Compata, but, unfortunately, Compata sold out about that time and things just kind of disintegrated. That's when I went on my own as a consultant. So this script mechanism kind of died there. I was pretty annoyed at that because I thought it was something that was a real useful tool and could have gotten some mileage out of it at Compata but just didn't go that way.

Hendrie: Yes. I think people were beginning to do packaged software for minicomputers. This was in the '70s?

Walker: Right.

Hendrie: Are we to the '70s now?

Walker: Yes, we're in the '70s, early '70s.

Hendrie: In the early '70s, yes.

Walker: '72, '73, somewhere along in there.

Hendrie: Very interesting. Oh, well.

Walker: Yes.

Hendrie: An entrepreneur. You needed to hook up with an entrepreneur. You write it and he sells it.

Walker: Yes. That's my life story. Invent neat things and never be able to capitalize on them.

Hendrie: Yes. So you went off on your own. What sorts of things were you doing?

Walker: You name it. It was all...

Hendrie: You name it, you do it?

Walker: Yes. Not much in the way of applications kind of programming, mostly systems kind of programming. A lot of work in diagnostics. I wrote several special purpose compilers for the test industry, wound up consulting for some period of time for a company called Xincom, which was bought by Fairchild and became the Xincom division of Fairchild. Anyway, I did a lot of diagnostic programming work for Xincom, memory test oriented programming and something else flashed in my mind...

Hendrie: So Xincom built testers?

Walker: Xincom...

Hendrie: Clearly if Fairchild bought it...

Walker: Yes.

Hendrie: ...it was for the semiconductor industry.

Walker: Memory testers.

Hendrie: Ah, for memory chips.

Walker: Memory chips.

Hendrie: Ah. Okay.

Walker: [I need to describe the test environment. Up to eight Xincom memory test machines were attached to a host computer via communication lines. The host computer was a Data General Nova. Test programs were generated and stored on the host computer and downloaded to each test machine via the communication lines. Each test machine was controlled by an embedded Fairchild F1 microprocessor operating on the downloaded test program. Test results generated by the test programs were uploaded to the host computer to be processed by the data reduction executive (DREX).] Test programs were written in XINTOL (XINcom Test Oriented Language). I developed a special purpose XINTOL compiler which ran on the host computer that generated for code for memory chip testing. It was kind of a—it was halfway between an assembler and a compiler. It was an assembler that could generate multiple instructions [for controlling the tester, including Test Pattern Computer (TPC) instructions and data structures that could be operated on by the test program and were recognized by DREX.]

Hendrie: A programming system.

Walker: Programming system. I didn't, myself, write any of the data processing programs.

Hendrie: So one of the engineers for a chip thought maybe what we ought to do is try to put all their net ones and zeros in this row and then, in the next row, put the reverse pattern.

Walker: Right.

Hendrie: Your compiler would help him do that, is that what you're talking about? Or don't I have the right...

Walker: No, you got the right idea. The compiler would actually translate TPC instructions to do this.

Hendrie: The programming system.

Walker: Yes, the programming system would certainly do that. The testers were actually two computers. There was a [very high speed] pattern computer embedded in the test computer and the pattern computer had its own programming language. You could tell it to generate words of...

Hendrie: Whatever pattern.

Walker: ...ones and zeros...

Hendrie: I got it, yes.

Walker: Various sorts of patterns.

Hendrie: Stuff that might make the chip fail.

Walker: Right. That was the idea. Stuff that would make the chip fail and so the compiler-like program could generate these test instructions but it didn't do much of anything automatically. It didn't know that you wanted to generate alternating ones and zeros, for instance, and you could use the TPC instructions for that.

Hendrie: But it was part of the overall software system...

Walker: It was part of the overall...

Hendrie: ...for the...

Walker: ...software system.

Hendrie: ...for a Xincom...

Walker: Right.

Hendrie: ...tester.

Walker: I didn't do too much on the host end of things. Most of my work was on the tester end of things, the special language compilers for the tester [(which did run on the host)] and various routines that could be used. It was part of a test program library kind of thing that would be called in. I wound up on an offshoot of that for Xincom as well. They were going to develop a bubble memory tester. Bubble memories never really materialized.

Hendrie: So they didn't sell very many testers.

Walker: Very few bubble memory testers but I did develop a bubble memory test compiler as well as the solid state memory test compiler.

Hendrie: Okay.

Walker: Interesting thing that never happened.

Hendrie: Yes, right. Exactly.

Walker: I spent, gee, consulting at Xincom almost five years.

Hendrie: Really?

Walker: As just a...

Hendrie: So it was a very long...

Walker: Very long.

Hendrie: You were essentially a contract employee sort of.

Walker: Basically a contract employee.

Hendrie: Yes.

Walker: Yes. They tried to hire me on more than one occasion but I didn't want to. They were happy with the arrangement as it went, I guess. So it just worked out.

Hendrie: Yes. Very good.

Walker: Back at the time I left there, a surprising number of people in the company didn't know I wasn't an employee.

Hendrie: Wow, okay.

Walker: Yes. The company grew enormously in this period of time. When I first went to work to consult at Compata, there were probably 30 or so people; fairly small company.

Hendrie: At Xincom, yes.

Walker: Yes, at Xincom, I'm sorry. Then, by the time I left, it had grown to a few hundred, I don't remember the number exactly but it had grown considerably. Of course, I'd been there all this time and all these people that came in were unaware that I was not a...

Hendrie: Yes, you showed up at work every day just like everybody else.

Walker: I was there pretty much like everybody else.

Hendrie: Very good.

Walker: That was an interesting period. That was the longest consulting job that I had in all the years I consulted.

Hendrie: Okay.

Walker: I wound up with other companies doing similar kinds of things. At Teradyne, I worked on a special translator for a language developed by IBM for describing programs. IBM had a—what the heck did they call it? It was a memory test system. They had a serial stream of bits they could put into their [chip] design and then clock it in such a way that it would cause whatever they wanted to happen in the system and then you'd get a serial stream back out. So they analyzed this serial stream coming back out to see if it looked like what they expected from all these changes that took place when you clocked the input serial stream throughout how ever many clocks that it needed. So it was a pretty ingenious thing. So I wound up writing a special purpose compiler for that language. It had some interesting things. The first company that bought the system from Teradyne wanted to use this system before the serial stream system was actually completed. So I got the compiler to be able to generate strings of test instructions that duplicated what the serial stream would do and then, of course, they couldn't get the serial data back out quite the same way but they could cause the system to go through the test and so the compiler then wound up being able to generate either way. You could generate the serial stream for the end purpose but you could use it on a standard tester that didn't have the serial stream capability and it just sent streams of test vectors out.

Hendrie: Oh, wow. So this was, again, a product for testing chips?

Walker: Chip.

Hendrie: Yes, it was chip testing, a Teradyne chip testing product.

Walker: Right.

Hendrie: Wow. So did you work with Teradyne in Boston?

Walker: This was Teradyne out in Agoura, just outside of the San Fernando Valley, here on the west coast.

Hendrie: Ah, okay. So it was a division of Teradyne? Because I know their home office...

Walker: Yes. The home office...

Hendrie: Is in Boston.

Walker: Is in Boston, right, but this is a division in Agoura. That's where this work was done.

Hendrie: Well, that's pretty interesting.

Walker: That was a fun project.

Hendrie: Challenging.

Walker: A challenging project and fun. It took the data—I was able to take the data that would have been put in through the serial steam and map it into test instructions to accomplish the same thing. So that could be used on these testers that did not have the serial...

Hendrie: Did not have this internal—yes, this is something designed into the chip, right? To help it be tested.

Walker: Yes.

Hendrie: Yes, I remember when that was something that people did a lot of.

Walker: Yes, if you had a really big chip, it was...

Hendrie: It was, yes, a really good way to...

Walker: Right. I'll be darned if I can remember what that process—there was a name for this.

Hendrie: Yes, I can't remember it, either. I remember what you're talking about. Did other people besides IBM make chips that had this in it or was this mostly...

Walker: I don't know. I mean, this was developed by IBM. The concept was developed by IBM and the actual user of this was a company in Texas. I've forgotten who had the testers but IBM was contracting to this company for chip production.

Hendrie: Oh, okay.

Walker: There was something, it seems to me they were in the Dallas area but I can't remember...

Hendrie: Mosstek or somebody like that?

Walker: It wasn't Mosstek but it would be somebody...

Hendrie: It wasn't TI (Texas Instruments, Inc.), of course?

Walker: It wasn't TI. I can't remember the name of the company. Anyway...

Hendrie: Okay.

Walker: Anyway, the user of the...

Hendrie: They were the user of this equipment and of this software?

Walker: Right.

Hendrie: Got it.

Walker: Where it was used beyond that, I don't know.

Hendrie: That's pretty interesting.

Walker: That was an interesting project, yes.

Hendrie: Yes. Wow.

Walker: I did any number of different kinds of things. I was a consultant for just a whole host of companies. I consulted for 20 some odd years, 22 years or something like that so there was a fair sprinkling of different stuff going on.

Hendrie: Yes, exactly.

Walker: I did some work for Bendix Ocenics division. They were building dipping sonar systems and I developed some code for that, some classified and some not.

Hendrie: Yes.

Walker: That was kind of interesting. This was Bendix out in Sylmar, California. Gosh, what other... nothing standout seems to come to mind. A lot of ho hum kind of...

Hendrie: Yes, just they needed something done and you weren't booked so you signed up and did it.

Walker: Right. Those tended to be more short-term kind of things, just get the job done and move on.

Hendrie: Okay. Well, that makes sense. I'd like to switch back and get a little bit of your early background now if we could do that, sort of switch gears. So can you tell me about your early life, where you were born and brought up, what your parents did, siblings, just sort of general background when you were a little kid.

Walker: Little kid, okay. Well, oddly enough, I started out life as a little kid. <*laughter*>

Hendrie: Okay.

Walker: I was born in Van Nuys, California.

Hendrie: Native Californian.

Walker: Native Californian, one of the rare. I lived my entire early life in the southern California area. I grew up in a community called Sherman Oaks, which was in the San Fernando Valley. Went to Van Nuys High School, the nearby high school there.

Hendrie: Tell me about your parents. What did they do?

Walker: Well, my dad died when I was four. He was an insurance salesman and I have about zero recollection of my dad. My mother never remarried. I have one sister that had Down’s syndrome.

Hendrie: She older or younger?

Walker: Younger. She was about two years younger than I am. She died several years ago now, been 20 years, I guess, since she died. I think that may be the reason mother never remarried, she didn't want to burden anyone with a child with Down's syndrome. My mother devoted a sizeable percentage of her life to trying to make life better for my sister.

Hendrie: What did your mother do? She was a single mom.

Walker: Single mom. Worked for the post office.

Hendrie: Worked for the post office.

Walker: Her entire professional career was at the post office. She went to work shortly after my dad died and stayed with the post office until she retired. In the end, she was—I don't know exactly what her title was but she was secretary assistant to the post master at Van Nuys Post Office.

Hendrie: But she hadn't worked before that? She just had to go to work when your dad...

Walker: Yes. Well, she—I honestly don't remember. She may have worked early in her marriage some.

Hendrie: Before she had kids.

Walker: Before she had kids.

Hendrie: There we go. Up and running.

Walker: Well, now all I got to do is get the—turn the thought back.

Hendrie: Oh, well, we were talking about whether your mother had worked earlier. You don’t remember,

of course.

Walker: Well, I don’t remember if she did or not. I—She probably did some work shortly after they were married, but she did not work any time prior to my father’s death in my memory.

Hendrie: He died relatively young.

Walker: Yes.

Hendrie: Did either of your parents have an opportunity to go to college?

Walker: No.

Hendrie: Both of them worked.

Walker: Yes. My mother may have gone to some kind of business school, but she didn’t go to college. She was fluent in things like taking shorthand and stuff like that.

Hendrie: Oh, so she probably...

Walker: She picked that up...

Hendrie: Somewhere.

Walker: ...somewhere at some business school. My father, I honestly I don’t know. I don’t believe that he did, but—I don’t think that he did. In any case, I grew up in Southern California and...

Hendrie: What are your earliest memories of what you thought you wanted to do when you grew up?

Walker: <*laughs*> Oh, gosh. At one point I know that I wanted to be in the forest service. In fact, I wound up going to school at Oregon State for that purpose. They had a major forestry college there, and so I was going to become a forest ranger. I went up to Oregon State. Spent a couple of winters of working in the woods with all the rain and decided maybe this *isn’t* what I want to do. <*laughs*> Pretty wet

up there. I’ve always been interested in the, I guess you’d say the engineering side of things, and I think somewhat creative for most of my life. Anyway, I decided, well, forestry wasn’t a thing. So, I switched majors into what was called agricultural engineering. It was kind of an offshoot of mechanical engineering oriented toward farm equipment, designing tractors and combines and all that kind of stuff. So, I got into that. Then, the money just ran out to go to school. I was up there trying to—my mom couldn’t keep it all up and...

Hendrie: You could earn some, but not enough to...

Walker: Not enough. So, I wound up in the middle of my junior year and dropped out and went to work at Rocketdyne and...

Hendrie: So that was your first job.

Walker: That was basically my first job at Rocketdyne other than trivial stuff that you work as a kid when you’re growing up.

Hendrie: How did you manage to get that job?

Walker: Well, interestingly, Carol, my wife, was working at Rocketdyne at the time and...

Hendrie: Now, when did you get married? In college?

Walker: No, it was post-college. It was ’57. I can’t think of the—for sure the year, 1957, I guess. It was about a year after I dropped out that we got married, and she had gotten a job at Rocketdyne. We had a mutual friend that worked at Rocketdyne who might’ve been the lead-in to her job there, and she heard about the need for—well, actually I started out working—I've forgotten what the title was. I was working on a test stand hanging motors to be tested and stuff like that and hooking them up and that kind of thing.

Hendrie: Yes, on the test stand. Getting motors set up for test firings.

Walker: For tests. Right. The stuff I was working on were called vernier motors. They were on the Atlas missile, and they were little, tiny motors, 1,000 pound thrust motors that helped stabilize the missile on a [launch]. So, I wound up doing that and monitoring these little vernier test motors, actually, running tests.

The tests were set up by a test engineer, but actually doing the operation of throwing the switches and running the tests. I did a lot of that. In the meantime, Carol had been working—this is all up on the hill, but she had been working in one of the other offices on the hill. I don’t remember exactly what her group did. It would’ve been related to data processing. She heard about an opening in the data processing section that I went into up on the hill, which is how I got over there. Oh, I did this test stand work for a fairly short period, a couple of three months, and transferred over to this division. I was doing data processing, data reading, reading charts and all that kind of stuff. That got me into the FORTRAN hookup and...

Hendrie: And then you discovered FORTRAN and you read it.

Walker: I discovered FORTRAN and I was hooked.

Hendrie: You taught yourself. Did you take a course in it or not?

Walker: No course or anything, just from the manual.

Hendrie: You just read the manual.

Walker: I read the manual.

Hendrie: Tried it.

Walker: It is something that came to me fairly naturally, I guess.

Hendrie: As software clearly has. <*laughs*>

Walker: Yes.

Hendrie: It’s proven that your brain is wired to do this.

Walker: It is wired to do software, right. So, anyway I got into that, and from there I wound up— <*phone rings*> Excuse me again. I hate to keep interrupting.

Hendrie: That’s okay. <*background noise*> Oops! Oopity. <*interruption—phone conversation*>

<*inaudible*>

Walker: Okay, I guess we kind of cycled through the early history a little bit.

Hendrie: Yes.

Walker: Up until the point of going to college in Oregon State <*cough*> and coming back and going to work at Rocketdyne as a test handler. I don't remember what the official title was, but it was the lead in into the programming world for me.

Hendrie: Yes, that's pretty interesting though, that you just decided—you picked it up and decided, "This would be fun, and I'm going to try it." And so you just submitted some programs? Some FORTRAN programs?

Walker: Basically—

Hendrie: To them, sent them down there and said, "Try these?" Well, you knew—

Walker: No, what they did is—it was everything on the individual. They had this set-up. They had a courier service that would go from the hill down to the processing center, and they had this sequence so they could write up instructions for the processing center. You compile this, run this, put this tape on, that kind of thing.

Hendrie: Yes, right.

Walker: And so I just—initially, there were no tapes involved or anything. I was just doing what you could do on a desk calculator without having a computer do it. And so fill out forms. Once you wrote a program, they could put it in the library. You didn't have to compile every time or anything like that.

Hendrie: You didn't have to key punch the program over and over again, or any of that.

Walker: No, you don't have to do that and the key punching operation is pretty slick. You had these forms to fill out, and they take care of key punching and sorting and going through all the steps of actually creating and running a program. Then once you're satisfied with a program, you could put the program in a library there. At the time, the library was all punched cards.

Hendrie: Oh, okay. <*laughs*> Yes.

Walker: So you have the object program in punched cards in the library and he'd send down whatever he needed to run with this and say, "Well, run this with this program, XYZ."

Hendrie: They pull the cards from the library and put them in the card reader—do all that.

Walker: Do all that.

Hendrie: They do all that, yes.

Walker: You generate a printout and send the printout back up on a courier to the hill, so it's the way the thing worked there. It just seemed like a good way to—

Hendrie: To do your job.

Walker: To do the job.

Hendrie: So that's what you started at, just doing—instead of doing the manual calculation sheets, you wrote a program, sent it down <*laughs*>, and had your answers come back instead of wearing your fingers out on the—

Walker: There were things that were too tedious to calculate by hand, that maybe something involving iteration or something like that, that was, of course, a natural for the computer. So I wound up probably expanding a little bit on what we were computing.

Hendrie: Okay.

Walker: That was just a lead in into the real programming job down the hill, which is where I met Ed Griffith, who has been <*laughs*> a factor in so much of this.

Hendrie: Right. Well, when was the last time you had any contact with Ed?

Walker: Well, Ed came to work for Compata, believe it or not.

Hendrie: <*laughs*> Well, now that's a turnabout.

Walker: Yes, that's a turnabout. I was there first, and he came to Compata. His arrival at Compata was by none of my doing. I don't remember how—again, he knew Bob Hooper and Bob Hooper was still at Compata, so that may be how that came to be. But that's the last time I've seen Ed. That would have been at the time I was still at Compata, which would be in the early '70s.

Hendrie: Oh my goodness.

Walker: Wherever he is, or even if he's still alive, I don't know.

Hendrie: Yes.

Walker: I ought to try to find him.

Hendrie: Yes.

Walker: He might put me to work somewhere.

Hendrie: No, I'll do an oral history of him. <*laughs*>

Walker: Yes, he would have an interesting oral history.

Hendrie: Yes. Now he was sort of a character. Do you remember any Ed Griffith stories from 3C days or other times? I just remember that he was a real character.

Walker: He is a real character. Gosh, I'm trying to think if there's anything that really stands out as a story. Boy, I haven't thought about Ed for a long time. Gosh, nothing comes to mind.

Hendrie: Nothing comes to mind? Well, can you think of any area that we didn't cover that we should have?

Walker: Well, I've condensed a lot of history into a couple of hours of conversation.

Hendrie: That is true.

Walker: So there's probably something missing in there. Gee, nothing comes into my mind that says, "Gee, we really ought to cover this." I don't know, so much of my life seems like it's been just kind of routine, that hardly anything stands out.

Hendrie: Well, you said something to me, I think on the phone, that your time at 3C was probably your best job that you ever had.

Walker: Yes, I would agree with that.

Hendrie: Talk to me about why. Why do you say that? Or can you figure out why that's how you feel?

Walker: Well, I think it's a combination of things. A lot of satisfaction just in what I did at 3C. A lot of satisfaction in the people I worked with at 3C. People like you and others that were at 3C at that time, I had very high regard for. Lowell Bensky, for instance. I'm trying to think of some of the other names of people that were there. Just in general, it seemed like there was a great deal of camaraderie in the people that were at 3C. We had kind of a common cause, and it seemed like as a company, we were really getting somewhere. I think at the time I was at 3C, we were probably about maybe a half step ahead of DEC in what we were doing, the technology and just general business sense. I think we were probably outselling DEC a little bit at the time. I think we were pretty much neck and neck, but my recollection is that we might have been just slightly ahead of DEC.

Hendrie: Well, I've actually looked at that, and I'm not wonderful with numbers, but what my recollection is, is that the year before Honeywell bought us, that we had almost 50 percent more sales than DEC did. And we were significantly more profitable than they were.

Walker: That sticks <*inaudible*>.

Hendrie: Yes, we were—

Walker: <*inaudible*> higher numbers.

Hendrie: We were doing quite well.

Walker: We were doing very well. I think there's a good feeling about having a successful product. I think that the 16-bit family was definitely a successful product, and I remember at the time, not too long before I left, we were starting to think of 32-bit families. In fact, John Thron and I—or you and I went out to see John Thron. He had a house way the hell out the Cape in Truro or someplace.

Hendrie: Yes, he did.

Walker: And you and I drove out to have a skull session out there.

Hendrie: We did?

Walker: Yes, we did.

Hendrie: Okay, all right.

Walker: We went out there. We were trying to put things in order for a 32-bit family of computers.

Hendrie: And, of course, that eventually did get built. There was a 6/32 and an 8/32.

Walker: Right. I remember those numbers, 6/32 and 8/32.

Hendrie: But they may not have been completed by the time you left.

Walker: I'm sure they were not. I'm sure.

Hendrie: Yes.

Walker: Yes, I remember driving out there. You had a BMW. Was it, 2002, was it?

Hendrie: Yes, I had a BMW 1800.

Walker: Oh, 1800.

Hendrie: Yes, it was before the 2002s came out.

Walker: Yes, I remember riding out with either the—

Hendrie: Really?

Walker: I drove back part of the way there. It was my first exposure to driving a BMW. I think [I got] kind of hooked on them. I've owned a couple or three since.

Hendrie: <*laughs*> It is fun, isn't it?

Walker: Yes. You saw my wife was driving an X5 down there.

Hendrie: I didn't notice, but okay.

Walker: Yes, I think that the thing that made 3C so good, in my mind, is that the—everybody in the company seemed to be working toward the same goal and there wasn't a lot of clashing of people in there. Everybody was doing their share of the work and that kind of thing. Up until the time Honeywell bought it, that's when it sort of disintegrated. But I think that—well you look at the caliber of people that were at 3C and where they went and what they did post-3C. People always talked about Harvey going to Data General and being a major force. The guy who joined at Prime and started Prime. You starting Stratus. There are a couple of guys that went down to Interdata, whose names I don't remember now.

Hendrie: I think the guys who started General Automation out here, they originally worked with 3C, for a while at least, I'm not sure they were long-term 3Cers, but Larry Goshorn.

Walker: Oh, I remember that name. Yes.

Hendrie: Larry Goshorn and Burt Yale. Burt Yale was a salesman.

Walker: Yes, he was a sales guy.

Hendrie: He was a sales guy and Larry worked briefly for 3C. Before that, he had worked for the DSI [Data Sciences International], which eventually then became Varian Data Machines. But he didn't go to that. He started General Automation with Burt Yale.

Walker: <*laughs*> I’ll be darned.

Hendrie: So there were a lot of people that sort of fanned out and did a bunch of different stuff.

Walker: It's clear that the staff at 3C was at the top end of the scale. Just look at what the people did afterwards. Well, the success of 3C, you could see it in that. Everybody had a common cause there. Just the whole thing, I think, came together [and], to me, looked extremely good. My guess is if they hadn't sold that to Honeywell, I might still be there. <*laughs*> But who knows.

Hendrie: I think maybe it was an environment that was more accomplishment oriented and less political oriented—

Walker: That was absolutely true.

Hendrie: Than many organizations.

Walker: What became of Frank Dean? I'm assuming he retired.

Hendrie: He is retired. He lives in Phoenix, and I took his oral history.

Walker: Wow, great.

Hendrie: So, I think he's still alive. I did that a few years ago. I've actually done the oral history of him, of a fellow who became the treasurer, not Bob Michaud. Having trouble with the name. I'll remember it. And Br—one of the other founders who's still alive and lives down the Cape.

Walker: Is it Bill Horton possibly?

Hendrie: No, I'm not sure—I don't think Bill Horton is still alive.

Walker: I remember he had a house out on the Cape.

Hendrie: Yes, I think a couple of them—Brooks. He wasn't there when you were there. He'd already left. And he was the first president, and then Ben Kessel took over from Brooks.

Walker: Oh, okay. And then Ben died in that airline accident.

Hendrie: Right, and let's see, who else? Paul Bothwell, he still comes to reunions sometimes. He lives up in New Hampshire. I've seen him a few times. Bob Baron lives out in Denver and moved into book publishing.

Walker: Book publishing?

Hendrie: Yes.

Walker: <*laughs*> Well, that's a change.

Hendrie: Yes, so some of the founders are still around.

Walker: Well, that's good.

Hendrie: Well, I think we've probably sort of covered most of the bases, so I'd like to thank you, Chuck, for doing an oral history for the Computer History Museum.

Walker: Well, I'm certainly more than glad to do it. I wish my memory was more up to the task.

Hendrie: I certainly understand. I have many of the same problems. All right, well thank you.

Walker: The thing that will happen now is I'll think of all these things tonight at midnight.

END OF INTERVIEW

