LP: ---It went through it's changes for better or for worse, a lot of people suffered to one degree or another from events or from behavior of individuals or from their inability to keep pace or from the divergence between what the company became and what they felt it should be and whatever. A lot of walking wounded and I have contact with some of them, at least periodically and they all come forward with a lot more than you ever wanted to hear in most cases, I am probably as bad as the rest of them, so. Anyhow what should we talk about?

BL: History. The concept of Digital's history existing in the words of those people who made that history. This is a powerful idea and basic. I believe the first book will be published in April or May. But maybe you won't read it. But how you first got into the company and how you became in charge of software.

LP: Let me just ramble then I'll take it from day one on up through the present. I was working as a programmer at Honeywell and a friend of mine was busy looking for a job and he found a position at Digital that he interviewed for and he told me I should interview too and I said, I wasn't really interested because I was reasonably happy at Honeywell and I hadn't been there all that long, one night Mahendra called me to do a reference check on my friend and he said, what are you doing and I said,

basically I am a programmer and he said, why don't you come talk to us and that's how I got hired by Digital in 1963. So Digital at that time was a small company about 500 and some employees. That year they had done 3 million dollars in sales and I came to work as a programmer in charge of software development for the PDP-5 which basically meant me and occasional help from one other person and there was a slightly larger group we were starting work on the PDP-4 was getting some work done for it and I believe the initial discussion became the PDP-6 had begun and since that was a high tech-high innovation content project - the time sharing system- that had everybody's attention, everything else was sort of uninteresting. The things they made the money on were not as interesting as the things that were technically further out and so I started as programmer with the PDP-5 and ---

BL: Was the project fairly long?

LP: IT was totally unstructured, there was no concept of a plan of software or of anything else and when I first started up my boss sort of wasn't there, they couldn't find him, they didn't know where he is, so I hung around for a few days then I decided I'd better do something and one of the then hot-shot sales people named John Cudella dropped in and he was talking to customers trying to sell

some PDP-5 and he and I talked a while about what you might need and we decided it needed an assembler and some other things and that was basically what happened. When I was in for my interview they were looking for Gordon Bell, I came into the interview by Gordon Bell, of course no one knew where he was, so I hung around for half a day and then I made another appointment to come back the following week and I came in and the same thing happened, Gordon Bell just never showed up , nobody knew where he was, so they handed me off to Allen Cottrac (?) and Allen was busy showing me for the fifteenth time showing me around the mill and he said, where are you from and I said oh, some little town in New Jersey and he said, where in New Jersey and I said, well you never heard of it and he said tell me and I said Vineland, New Jersey and he turned to me and said, you're hired I'm from Vineland too. So that's a true story.

BL: Did you also go to MIT like him?

LP: No, no not at all, I'm not an engineer, I have a degree in Business administration and I got into programming by virtue of being interested in programming at a time when there were damn few programmers and no formal training that anybody knew of for software development.

BL: Did you have extremely strong mathematics in your background?

LP: No, not really. I took an initial set of courses at the [INAUDIBLE NAME] University night school in [INAUDIBLE] County and then I went to work for an insurance company in Philadelphia that was as a programmer-analyst trainee they were contemplating the purchase of a computer and they contemplated and contemplated so I basically learned on the job, which is the only way in those days. And I grew up with the craft as it were. So I ran the PDP-5 development such as it was and I was kind of aware of but not really involved in the evolution of the PDP-6 software and the project got progressively in trouble and at some point in time they asked me if I would take responsibility and see if I could bail it out, so I picked up the software development for the PDP-6 development and then ultimately picked up responsibility for all of the software engineering and that's the history in a nutshell and over the course of the years I was basically always responsible for software engineering and over the course of the years I picked up other areas of responsibilities including founding the software services organization and I ran that for a number of years till it turned into the P&L center. I had responsibilities in many ways for various and sundry of Digital's product lines including

price-setting for some period of time. I had many bosses, there was a time where a piece of me worked for Ted Johnson and a piece for Hendell (?) and piece for Stan Olsen for various - organization wasn't a big issue in those days. I actually proposed and was responsible over all work for Olsen that became the first commercial product line the first attempt to sell DEC systems for PDP-11 to a commercial environment, which was a traumatic thing because DEC did not make commercial quality equipment DEC made scientific quality equipment which basically said that the users were ever forgiving and the service policies and the quality of equipment and the viability of the equipment wasn't up to commercial standards. They had no concept about the commercial standards. It was all a learning experience. When Gordon Bell came back from Carnegie-Melon, you are familiar with the demise of the PDP-6? At the time the PDP-6 met its demise we did two things, one is we formed a study team to figure out what the foul in the machine was a combination of Allen Cottrac, myself, there were two or three [INAUDIBLE] I believe, there were one or two other people involved whose names I don't remember. To basically - the intent was not to lose those key people. So the bone was go think about what we should do next now that we've gotten this thing out of our system. The other thing we did at the same time is I actually went to the Board of Directors and said even though the PDP-6

hardware has met it's demise we have a tremendous investment in state of the art software in the timesharing system and I would like to continue the development in anticipation of someday putting it to use. So we did and what happened obviously was that the PDP-6 begat the PDP-10 which was basically the PDP-6 done right and the software development had continued apace and so that became the genesis of the PDP-10 product line.

BL: Remarkable, Allen Kotok talks a bit about the 6 of course, and I think described it as an elegant architecture and elegant design for engineers, it just didn't quite work because of the signal integrity.

LP: Well DEC had no idea of how to build complex computing systems. They built simple things which was of the scale dramatically larger than they or almost anybody else had attempted and I shouldn't say that IBM could, this would be the machine but for them this was a major undertaking they had no idea how to design it, how to do the component evaluation and sourcing, how to make it maintainable, how to build it, how to check it out, it was very much a plug it in and let the smoke kind of approach. We spent endless hours trying to get those machines to work, I mean I remember being there six and half days a week until 1-2 o'clock in the morning. I remember the computer we had to use for development was

on the first floor of the building, building 12, on the first floor that's where the computer was initially it was in building 5 in a place where there was a very fancy raised floor structure that cost a fortune that we designed in-house and build in-house probably at four times what it would have cost us to buy it outside and then we moved it down to building twelve and we were there until all hours, we used to take these space heaters and jumper the shut-off switches, tilt switches, lay them on their side and use them to heat pizzas at night. I remember many a night calling Gordon Bell at home and telling him the machine is dead and we had a bunch of people using it, Gordon would come in his bathrobe and pajamas and his toolkit and try to fix the machine and anyhow, ultimately, they decided they couldn't make it work, up to any reasonable standard.

BL: What thought was given in the design _____ to software in the beginning?

LP: Well, I think people basically understood that it was joint effort between hardware and software that certainly was a first for DEC, so there were lots of trade-offs and the hardware designers were very software knowledgeable people, extraordinarily so. Bob _____ and certainly Gordon , so these people are all multidimensional and the classic mistake we made was we

were designing complex hardware and complex software at the same time, so you couldn't debug anything because all the debugging tools were made to work under the time sharing system and the timesharing system didn't work. We didn't have a good concept of building the tools through strapping ourselves, everything was done in parallel in the same emerging hardware base and it was just very difficult to do, working with the advantage of hindsight you understand that you can't build things that way. But we didn't know any better, so. It was a big trauma. At the time the original vice President of DEC, Holland Anderson, Andy, who I remember very fondly and he was a gentleman a really nice guy, he left is some dispute which was never understood by me, but we also noted that he went - if you want anecdotes I have possibly a half million maybe. Independent of what you may use - we reached a point on the PDP-6 when it was really critical we were in serious trouble and all the attention and energy of the company were going to this product line which was busy draining the company and the businesses that were profitable weren't getting any attention, which was ultimately the demise of the PDP-6. But at this period of time Andy had responsibility for the PDP-6 product line and he had daily meetings, how are things going, in Ken's office and one time I remember that we had had a University of Oregon had, we had been contacted by the U. of Oregon and they were interested in

the PDP-6 potentially or something that would inventory their dairy herd management program which was a very big thing and still is I assume. And the problem is that they needed a COBOL compiler to do all their software development and to use the existing software. I was the only one at DEC who even had installed COBOL so they packed me onto a plane and shipped me out to Portland Oregon to talk to the University and I got out there and I got quizzed for two days on the details of the design of the compiler we hadn't even conceptualized yet, but when I came back we agreed that we try and go out and buy a subcontractor-development of a COBOL compiler in order to get this chunk of very interesting business. So I sent out for bids to software firms who were in the business developing compilers and Bob Wayne, who was the marketing manager for the PDP-6 at the time said that he, an old ITT guy from New Jersey- said hey, just as a courtesy why don't you ask ITT to bid? and they were sort of into it, they thought they were in the software business, so I sent them a letter inviting them to bid on the problem of the compiler and they came in at about a factor of 7 when everybody else wanted, they were clearly out of the ballpark. So after we got the bids in I sent out an initial letter to everybody except three final bidders and the letter said, thanks for your bid, we appreciate your taking the time, we have narrowed the list down to 2-3 bidders and unfortunately your bid

wasn't one of them, but it has been a pleasure, whatever... and so we were having this Monday morning, how are things going status meeting and Ken said, how are things going and Bob with a great flair jumped up and whipped open his jacket pocket and took out this letter and said, not very well, thanks to Larry Portner and what it was was an nasty gram from somebody at ITT who had got the rejection and the problem is I had signed the letters and the secretary had put the wrong letter in the wrong envelope so it went to the right company with the wrong name on the letter, so Bob was all set to roast me on the spit, for the demise of the PDP-6 and Holland Anderson much to his credit and to my relief, stood up and said, Bob, that's totally ridiculous, if there were ever a serious candidate for buying anything from us the fact that the letter was addressed wrong by mistake certainly wouldn't deter them, let's not have any of that foolishness. So, here I was a twenty-three year old green programmer in way in over my head in this potential crash of a major business enterprise and anyhow, that was the story.

BL: Did the COBOL compiler ever get developed?

LP: Yes, ultimately we did develop one, it was developed in-house and we hired some people from outside that I had worked with previously at Honeywell actually, All Black

was one and some other people. We did develop it, for a long time we had the fundamental ethic as many companies did that we could develop anything and NIH was a very strong factor, it went with the territory, it was the behavior that the company encouraged - you got some bad stuff with it as well as some good stuff.

BL: On the 6 also, that must have been an extraordinary software problem resulting in the release of timesharing system, apparently from scratch. What models were you using?

LP: Well, the big model of course was the MIT project MACK system and that was the model for timesharing systems and there was a lot of, I wasn't a designer of the PDP-6 by that time I had basically had already transitioned briefly from being a developer into being a manager of development and I can't take any credit for any aspect of the PDP-6 software system other than I'll take a lot of credit for having made it happen at DEC, but that was basically a brute force kind of thing. That was a push and pull and lead and expedite and facilitate and argue and control kind of management style because we were already in trouble. I will take credit for one invention in that space which is the help command. Which is now pervasive, every personal computer in the world had Help - I invented the HELP command, because the

timesharing system from a user's perspective was changing nightly so anytime anybody like myself wanted to get on and do a little work you could never figure out what change, ~~so I said,~~ I can't remember from day to day so I have to get a script that you invoke with a command called help and it was a primitive kind of thing, but basically as it came up it would give you a menu and you would say if you want more information on commands, software availability, whatever, type this in and that ultimately became the help command that the industry knows and loves today. That is the extent of my development involvement. So anyhow, the PDP-6 begat the PDP-10 and the PDP-5 begat the PDP-8. I did a FORTRAN compiler for the PDP-8, in a similar kind of thing, I didn't know what a compiler was at that point in time and we went out and looked for bids and it was initially for the PDP-5 and the best software house in the country said if we wanted a FORTRAN compiler for a 4-k memory of 12-bit words machine and the people we spoke to said we could do it for an 8-k at 12-bit or a 4-k at 24-bit but we cannot do anything for that. So I was sitting in my office bemoaning the fates and my boss dropped by and he said, why don't you do it? and I said, do what, and he said

write the compiler', I said, 'I don't even know what a compiler is' and he said 'hey there is this whole communication with the ACM which talks about a precedence

scan for scanning input and why don't you read it and I'll bet you can do it. To make a long story short, I did it and I did it in six months. The thing actually survived for years and years and years and I met years later a fellow from Dow-Batteshey (?) chemical company in Texas and we were at a DECUS meeting he comes up and says, "I always wanted to meet you, you know we have run our whole plant on your FORTRAN compiler, we have six hundred programs that runs every part of that plant and was all done in the FORTRAN compiler - so I guess primitive by any standards was still adequate. That's probably the single most interesting and useful software development piece that I actually did.

BL: Considering the market as being primarily a _____ engineering machine, FORTRAN was probably the most critical part.

LP: During this time the company just grew, we had a conscious strategy and it basically said - push the technology hard. Listen to the customers and try to understand what they want and then do it, be responsive. Compete with the other mini-computer companies by constantly taking advantage by expanding the playing field, which was - when everybody else could go processing units like we could then you start building mass storage devices and everybody else would build mass

storage devices then you start building printing devices, when everybody else can do printing devices you start interconnecting and when everybody else can interconnect, you start providing software services, so we are constantly expanding the playing field, always trying to maximize our internal efforts, our strength, our capacity, our financial strength and that was the way to be in this play. It was not articulated as a strategy, but defacto, that was what it was, mostly it was done by emotion. The PDP-10 the program the business was killed on many occasions and it always came back because some big customer would call up and complain or some multimillion dollar order would make us decide that maybe we should change our mind yet one more time and in a sense it was an experiment in autonomous management in the company that can also really ran right down to the _____. All key decisions were made and ratified by Ken and the ones that were made by other people couldn't be implemented because Ken would basically play with them until they died. So. In that way Ken could take credit for the things that worked and avoid any responsibility for the things that didn't work and I believe to this day that's the Ken Olsen style. I learned a lot from watching Ken, I really did, a lot of stuff I clutched to my bosom and try and emulate and alot of stuff that I wouldn't touch with a ten foot pole. But in particular, one thing that I do remember that stuck

with me and I think had a good effect is that -are you interested in anecdotes or just history? - there was at one time, we went through the facade that the product line managers really ran the company and there was a huge product line management committee, managers meetings and Ken would let them play with strategy and basically strategies almost always meant budget. So we were putting together a plan for some year, and there was a particular irksome competitive environment I think, SDS or SDC whoever it was was competing with us on the timesharing domain of the PDP-10 against the PDP-10 and we spent a lot of time debating the company strategy which meant which products and what markets and how we going to make the numbers, so Ken sat there and listened for a while then he got up and he said, well, I am going to let you guys show me how smart you are, you figure out what our strategy should be and call me back in when you've got it figured out. So he left and they churned for a while, we spent one and a half or two days hacking away at it, finally we had talked ourselves into something which we felt good about and we called Ken back in and he comes in and stood up and he said, think of it this way in the board rooms of every one of our competitors across the world, there a bunch of guys who are no smarter than you are who are thinking the same thoughts and drawing the same conclusions, you honestly can't win by doing the same thing as everybody else is

going to do, so forget that and start over, the accent on it was, the obvious is probably wrong because it is obvious. So when you build everything you know how to do and lead yourself to what you view is the obvious conclusion, force yourself to throw it out and make believe that God intervened and said, hey guys you can't use that one, try again. And then figure out how to go around the problem, come in from a different approach, looking for an edge to differentiation in your thinking and your behavior and I don't know if he meant that or whether he was consciously training anybody or whether he was just telling us how his mind works, but in any event that made a big impression on me.

BL: That's wonderful.

LP: The rest of the company the rest of the time there had the characteristic that we were opportunistic basically, we grew at a time when the industry allowed mistakes, it was very tolerant, you could screw up to an immense degree and the fundamental demand for computing products was so strong that the pace of technological operation was so rapid it seemed so at the time, that you could really screw things up to an immense degree and be covered because the demand was still there and it applied to development and marketing and quality aspects of the business. So we were fortunately inclined to enter, you

didn't have to be terribly smart you just had to be there with capacity and drive and initiative and the game has obviously changed over the years and at some point in time when it had changed, DEC screwed up in some monumental fashions. I mean truly monumental fashions, out of ego, out of stupidity, out of naivete out of process. And the company became unmanagable at some point not too many years ago. Unmanageable because they let everybody do their own thing, let the good stuff rise to the surface, the good ideas, we'll have a champion that will prove his mettle by driving things through the morass, it stopped working at some point when it became impossible to get anything done, there were too many constituencies, too many product line managers and too many committees and too many nay sayers that pooled together the engineering budget was a nightmare, everybody wanted everything and you could not say no to anything because if you made a decision that looked strategic and it hurt somebody or some business, that's not what I need, they are all competing for the attentions of a finite resource pool. All these businesses trying to differentiate themself by product wanted something different, but you can only get so much out of ----- so you finally say, it doesn't made sense to do that, we haven't got the budget for it and it just seems like it's less important in the grand scheme of things than this. There was no agency to make that

decision so it fell to the Gods who were playing the role of Director, in this case the interior people to say, we can't do it, we don't have the resources and that guy would get on the phone to Ken and the next day Ken would call up and say - gee you really have to do that because Sam is good fellow and he works really hard, and he could maybe go out of business. So things churned and churned and decisions got harder and harder to make. Strategy really couldn't happen, we had a number of embryonic attempts to form strategy committees and what have you and finally someone who I respected in some respect said,

[end of side one = begin side two]

"You are wasting your time", I said what do you mean and he said, 'well, no strategy activity is going to work in this company and the reason is that Ken doesn't want it to work, because if Ken has process that helps the company work that means he is not going to be as fundamentally involved as he wants and needs to be so, forget it', and I think that was a piece of dramatic insight. So, it just got worse and worse - Gordon and Ken were feuding over who had the biggest ego or who had screwed up least and of course as they say the winning generals write the history books - so that was it. The way I characterize it is I spent twenty years at DEC, seventeen of them were the best years of my life two of

them were terrible and one was absolutely awful. That's history.

BL: You had begun saying that there was a lot of listening to the customers in the '70's meaning that you must have been fairly close and understood what they were asking for.

LP: IT was intimately tied into it's customers. There was a lot of contact. The customers were members of the family in essence, the customers were members of the family, there was a strong dialog at all levels with the engineers and the customers and potential customers, DECUS was very active and very effective as a lobbying committee, the product line managers who knew their business were intimately involved with the customer base, the whole laboratory data products business. Clayton and those guys would go out and spend a lot of time listening and being involved and sitting around and - we built what the customers told us they needed.

BL: From the software aspect, what was that? And how did it change?

LP: Well, it was a less a software issue actually than it was a hardware issue. DEC was, for many years, and may even still be today, in its heart of hearts, a

hardware company. So it was mostly hardware, the software followed. It was always viewed as a necessary evil and with the advent of the PDP-11, in fact, the PDP-11 happened without a software plan. The software budget was basically zip. There was initially no money for software for that and when I used to go over and complain to Olsen about how shortsighted that was, he'd say, hey, you do the hardware and the software is free, the software comes from heaven. So --- and at that time the industry was tolerant and we sold to people who didn't make a hell of a lot of difference, we did quite good with those markets, unfortunately the same attitudes that were adequate in those days became a real negative as the industry matured and competition happened and software became more and more the essence. I can't remember the exact year, but at one point in time when morale was really low in the software engineering organization they were all feeling like second class citizens we had a series of meetings in the basement in Sudbury and we decided that the reason that everyone felt lousy in the software engineering organization was that their product, the fruits of their labor had no relevance in the company and the reason it had no relevance was because it wasn't a product. A product was something that you shipped, you invoice for and it brought in money. So we put together something called the Software Product Proposal and the essence of it was to unbundle

software and charge for it and we spent almost a year selling that to the company and there were huge product line manager meetings where people stood up on their chairs and screamed and said, it's immoral to charge for software, software is supposed to be free, it has no intrinsic value and however, in the final analysis we did unbundle software and that's now a multi-billion dollar business for DEC. But at the time it was basically a hardware company immoral to charge for software, despite the fact that it had high development costs, the fact that it had no manufacturing cost, made it zero value and the concept of charge based on value rather than cost never set in. So that spawned the software business, the software services business followed in its footsteps basically it said, hey if you can sell the product you can sell the services and the services are differentiated between product required, which is what we used to know as warranty. We built it, it failed, therefore we had to fix it and we can fix it under warranty or we can fix it for money and so there was product required service and then there was something called customer desired service, which is - you may not have any obligation but we might benefit from some additional work and we would be willing to pay for it. So, that's spawned, again another multi-billion dollar business for DEC, which today - if you look at what held DEC together through the eighties, when the PC revolution happened and DEC did not

participate for a whole bunch of reasons which I am sure you have heard more about than you'd care to. What held it together was basically the VAX architecture, we captured the marketplace with an excellent machine and an architecture that allowed us to follow the technology curve and not induce transients into the customers software existing application base. DECNet which was basically the ability to talk between a variety of DEC platforms and to other manufacturer's platforms. And the services business and in particular the software services business which by now has one of the highest growth components of DEC's repertoire of businesses so that was DEC's strategy. That was what held it together through the eighties. VAX architecture, networking and services.

BL: I would guess that because there actually was a software strategy for a change, but I'll take that back a couple of years to the beginning of the eighties at the PC revolution, how did we get into the situation with between seven or eleven operating systems for the PDP-11, unbelievable amount of products the end of the 8, the new PC operating system, [INAUDIBLE]

LP: It grew like Topsy and since the basic philosophical approach, the fundamental strategy approach was do whatever the customers want and internally it was, hey, if you've got a good idea and someone can carry it

through and sneak it through, that was blessed by the system. Sneaking something through, now you'd better not fail and you'd better not--- if it failed and you got caught you were a bad guy, but if it succeeded it was okay if you snuck it through. There was even a premium associated with sneaking it through the system. So we were responsive to a variety of markets and when we had the PDP-8 and then we decided it was getting long in the tooth architecturally, we invented the PDP-11, but the PDP-8 was still a major business, it still sold a lot and we said, hey, why throw it out, you've got a tremendous investment, let's milk it. So then there was the PDP-8 family and the PDP-11 family happened and because DEC was really a bunch of little fiefdoms, a bunch of little product lines and they all had the right to develop whatever they wanted or to acquire it to meet their customers requirements or desires or differentiate themself or solve their ego needs or whatever the reasons were it didn't matter. They could do it if they made money. So they built all these business that grew out of being responsive to local market opportunities. As the market place matured, and customers became customers for multiple of DEC's products, suddenly they said, hey, I've got one of these and one of these and one of those and now I'd like to kind of share data or plug them together and I can't. Everything is totally proprietary absolutely unique, whatever, so that spawned a real

problem for the company. They got that way for good reasons, they were responsive, maximally responsive to the marketplace. Then as the market matured and the customer base matured it suddenly was a bad thing, so we began to DECNet was the first instrument for allowing us to interconnect this diverse set of products, so one of the major drives of DECNet was to provide a unifying factor across a plethora of DEC systems which were otherwise incompatible. And initially DECNet had no relationship to the outside world, DECNet was DEC-Net and it was to solve the problem of these things that didn't talk to each other. That's where DECNet came from. At some point in time after DECNet phase one sort of was high expense almost failure, Ken dropped by and he said, should we kill it? and I said, to be honest we don't have any choice but to continue development, because at some point in time everything is going to have to talk to everything and if we stop now, we have already paid the price for the learning, now we ought to continue and we did, we did phase two which turned out to be a success, but it was many things, we were lucky because the size of the company and the basic design philosophy which said, lots of irons in the fire and all over the place, so some cyclical businesses are down, something else is probably going to be doing good to compensate and if some countries the economy is poor, some will be doing well and that worked in an environment where the total demand

for products was growing and where the competition was basically not there, we weren't big enough to attract the attention or have to compete with the giants and we were bigger than the small guys and we had this wonderful niche and alot of momentum. And it worked very well.

BL: I wondered about the commercial niche which certainly wasn't tapped for the longest time. You mentioned that you worked on that and tried to go into it on the PDP-11, was that the TRX?

LP: No, that was basically a RISDS business, Ed Marinaro was the product line manager, ED worked for me and I and/or he, sort of worked for Stan Olsen and we were basically selling RISDS systems, the first big customer, big contract, was a company called Formos-McKesson, which in hindsight turned out to be kind of a bad scene because we never could make the equipment work up to their standards, though we installed a fair amount of it, but it was always down and our service policy - the service policies were nine to five, five days a week and you were talking to companies who were trying to run their business off these computers and to be told they weren't going to get a serviceman out for four days didn't sit very well with companies who basically viewed the commercial marketplace as being IBM, who was babysitting them all the time. This was, I guess Irwin Jacobs

ultimately took over that business and ran it out of Merrimac, N.H. and he reported to Julius Marcus and it became an amalgamation. The residue of the first commercial product line, I can't remember what name it had but it basically was selling RISDS systems to commercial customers, ended up being folded under Julius Marcus in Merrimac and when he became vice president for commercial. It included at that time included the telephone company product line and Irwin Jacobs commercial OEM business and a bunch of other stuff. But that was really the first attempt we made to sell to a commercial customer, commercial accounts who had different needs in terms of product, service, fundamental reliability, software and everything else. I'll give you another little anecdote which you probably won't hear from anybody else because everybody wants to take credit for it, but - at one point in time we had two days scheduled when I think it was General Electric, who was a big customer of DEC's, came in to Merrimac and we had prepared, I was sort of working part-time for Julius Marcus as a [INAUDIBLE] consultant at that time in addition to my other jobs, and they put together a big dog and pony show for GE to tell them Digital's strategy and strengths and what-have-you and we got together in the big conference room and the DEC representative got up and started to talk about what the agenda was going to be and the heavy hitter from GE, whose name I can't remember

said, No, I'm sorry we have a different agenda, we didn't come here to listen to you, we came here to tell you what we the consumers of computing products find important and they proceeded to spend a day and a half giving us their version of what kind of computing they needed and they described it, it was called the Cloud, and basically it said, we want to have terminals, smart terminals and dumb terminals and small computing systems and medium computing systems and large computing systems and we want to have them here and there and there in this building, in this office and other offices in other buildings in other countries, whatever, and we want the same keystrokes that you use at the dumb terminal to do the same thing at the big system and we want the same applications to be able to run anywhere in that system and we want the data to be able to move around that system and they drew this thing which was all these interconnecting lines and they said, that's what we want, that's how we want to run our business, we need the computing environment like that to support the running of our business and that ultimately got re-invented at DEC, as the basis for DEC's strategy around VAX for example, the fact that the growth and the architecture, the ability to grow and maintain a single architecture a single operating system and a single mode of behavior across the tremendous dynamic range of computing. That actually became re-invented in DEC as the essence of

DEC's product strategy and it came from General Electric. It came from exactly where it should have come from, which is that people who had a need to run their business talking about the kind of computing environment that they thought was appropriate to enable them to do that. Interesting piece of history.

BL: To create that strategy there must have been a revolution from what software was needed from operating systems to tools and utilities to languages and then finally, eventually, applications . Why? When was the need for applications ----

LP: What happened was that at some point in time and I believe this was in the latter days of the PDP-11 and before the VAX we developed a model, two models actually that were significant in terms of future growth of the company from a software perspective. One was the onion skin model, which basically gave birth to the whole concept of layered products and it basically said, hey if you were like an onion skin at the bottom you've got the fundamental piece of hardware that establishes the instruction set requirements and on top of that you've got an operating system and then on top of that you have what we call layer products which were the compilers, the languages the data manager stuff and on top of that you have the application and the goal was to go from the

innermost core which is what we see that defines the instructive raw capabilities to the outermost core which is where some person who has some set of problems to solve sits a some device and needs to utilize the computer so we concocted this model which shows up all over the place now and I honestly don't know if that was original at that point in time or not, but it was a convenient way for us to think about what we built. That was the model that basically said, somewhere you have to connect to a user and that ultimately gets you into the applications discussion. The other thing that model did is it said that if you standardize the interfaces at each of those concentric circles then you could have lots of development going on in parallel, all the time everything will continue to work because if the interface is frozen and managed, so we put together another group under Phil Keating I believe, that spec'd the interfaces now, I'll get my time scales and my names and my which product line was on, all mixed up because I never kept good reference, but that onionskin model was significant and then of course services were somewhere at the outer layers. Applications were never a strong force at DEC, ever for a whole bunch of reasons, one of which was we didn't have the expertise to build them, number two we decided it was smart to stay away from it as long as we could but then we could sell the material, let the customer take the responsibility. But that was one of the key elements,

the other key element that determined our software strategy at the time was something called Support Monster, and we did a little study and we looked at the proliferation computing systems and the software support we had and the cost of supporting the product and we did some extrapolation of trends and we said, hey in three years the cost of maintaining our software at the rate we are going will eclipse the total revenue of the company so we had better do something to head off - to eat the support monster before it eats us. We had the art department, I don't know what happened to it, put together a great big chart on the wall which was basically a dinosaur and it had the growth and revenue of the company , the growth in the unit volume of installed machines and the growth, projected growth and support costs and you could see where they - we had a big campaign around Eclipse and that led to a change in policy in terms of what our warranties were and a change in our perception of the impact of product quality. The product quality thing we played around for a long time in software and then finally get upon the only right approach which was, you can't add it on, you can't test it and basically you have to design the stuff and build the stuff from day one in a way which lends itself to reliability and quality maintainability. And we sent a guy whose name was and still is, Lou Cohen, who is a really bright guy and he was in the software quality

management group and we sent Lou Cohen off to put together an approach which was based upon doing it right the first time, training the developers how to build software that had a highlight of being good. Cohen went off and he was publishing this and he spent a lot of time lecturing the problem and researching it and he came back with a whole menu of how to build software and that ultimately became the software development process for DEC and it was married to the phase written process which basically said, hey you structure the whole thing and you approach it as an engineering discipline, not an art form, and if you want to make it an art form, that's fine but you always have someone who is the clean-up person associated with the artist, so the artist goes around and does art and the clean-up person makes it commercially viable and that was the essence of it.

BL: Was that successful?

LP: Absolutely. You'd have to ask DEC customers, but I know that our goals at the time were to change the shape of the growth curve into the support cost curve which we clearly did and we started charging for some of the work and we started putting some basic discipline in field organization up to some point in time the software services organization were basically handmaidens of the salesforce and because it was a free resource, it was

always 100% used up and the salesmen could be courteous to the customers by giving open ended quantities of ongoing support to these customers and we finally said, no, you can give away as much as you want, but it's a budgetary item consideration and it shows up as part of the cost of selling. So if you believe that an account strategy for account X is going to be maximized to our advantage if we give them lots of additional support, that's fine, but recognize that that's part of the cost of selling to that account, so when you look at your cost of sales, that's part of that calculation, so you can give it away if you want, but someone has to pay for it. Alternatively, you can sell it, in which case it produces revenue for the company and if you're a good salesman you can figure out how to sell it, not give it away and by the way you'll get bene's out of having sold it. Part of the software services scheme was to basically differentiate organizationally so that salesmen know that while they have the ability to give it away, to establish a budgetary mechanisms to allow them to budget for an amount that they could give away. So they could do the following things, they could hire some number, they could basically pay for, plan and pay for, some number of pre-sale support people, which basically was a local office decision as to whether or not they wanted more software people or fewer software people whatever was appropriate for their local market. So they could plan

and budget and pay for pre-sale support out of the software support organization. They could buy additional economic quantities after the fact or they could sell the service. The goal there from the software support organizational standpoint was to give them a local accountability and a profit quota. And the original goal was - you can give away as much as you want, this was the goal for the software service managers - but within one year the profit from four sales services has got to equal or be greater than the cost of the giveaways, so you can manage either side of that, you can manage the giveaways down to the profit up or both, but that's your goal and they did it in much less than a year. It turned out that they became so enthusiastic about the profit modem that they turned the whole thing around. That was another major breakthrough for the company in terms of being able to sustain profitable future growth. And that is what I know. Somewheres or other if I thought I could find it I even have pictures taken years ago when we used to go in on Sundays and build our own offices out of porch doors and whatever, because whoever used to build it couldn't build them fast enough and so we used to take the program department and they built their own offices on the weekends. So much history.

BL: I appreciate this so much.

[END OF PORTNER INTERVIEW]