(12)

INTERACTION AMONG TECHNOLOGY, PRODUCTS AND USERS

C. Gordon Bell, Vice President - Office of Development
Digital Equipment Corporation
Maynard, Massachusetts

I have had a very close association with DECUS because we share the same Engineering budget. Since we always have more things we want to build than we have money for, there's going to be pressure from me to not expand DECUS at my expense even though I believe it is a very necessary part of our environment. For its own good, DECUS had probably be moved elsewhere within DEC for funding.

I am happy to come here for a dialog with users and to discuss the process of determining and building a product. This is, in fact, what I'm responsible for and also is my main interest. I gave a talk earlier in the BASIC session and somebody said, "What's the punch line?" I put the abstract out internally and the Marketing people didn't want me to part with the punch line. The punch line was a question - "Does anybody care if all the world's implementations of BASIC are non-standard and you can't transport programs across systems?" - a question I felt would be important to users. It certainly concerned me as a recent BASIC user.

I'm fascinated by the whole business of machine evolution because, due to the exponential nature of our technology. In contrast to others, it is substantially faster than real time. Said another way, one ages very rapidly. Even though we've had only 25 years since the first machines were implemented, we've experienced about 8 orders of magnitude change - depending on the indicator. For example, the automobile has changed by a factor of 10 over a 50 year time period using horsepower as a base.

The way I characterize the process of computing is shown in Figure 1. It is a multi-stage pipeline process in which one starts with research - or at least the research people think that the world starts there. (I don't know where they get input.) In fact, there are some ideas that come around very often. I guess they come from 19th century math and physics. Phenomena starts it (e.g., the transistor), and understanding such as context-free grammar. This goes into an advanced development phase where people work with these ideas, although to many who compute it still all gets lumped into the notion of research. Then we start building a product with technology as a collection of activities called architecture, implementation (which is really the hardware), and the operating system. These all go together to determine the machine. These should really happen in parallel and there is much interaction. Most of you perhaps think that it's serial, that is, somebody builds the hardware and then somebody builds an operating system. By that time several years have gone by and you end up with something no one can program. That happens too. Actually there isn't that much interaction, as you take architecture and build languages on it. There probably should be more.

Then, of course, applications occur, given a language. At each level we are creating another abstract machine along the lines of a set of concentric rings. Each ring takes maybe a couple of years. Maybe by this time we've made such a good system that it only takes a year to apply something. Then the user finds out that it doesn't do what he wants it to. That input goes through a stage called Sales, which is the interface, then there's Marketing and finally it comes back to the designers. This is really a fast response, taking only a month or so. The feedback is highly noisy and biased.

Then, of course, there are other components that get into the act. There's a group here - let's call it government/standards. They're sort of trying to affect and control all of this, usually standardizing on whatever IBM did a few years ago. For example, they want to standardize on the memory-processor interface and somehow I think we've standardized already on the memory-processor interface as a UNIBUS. There are many, many companies making add-on memories for PDP-11s. Standards do encourage competition and I can't imagine anything that's more standard and that's more competitive than peripherals and add-ons to the PDP-11.

The other characteristic of this standards area is that it is the wrong level. A few people believe they can standardize on some signals. This turns out to be probably the most difficult thing to do, because you're trying to specify rise and fall time of a bunch of wires and there really isn't a good method of specifying that at all. They could be useful by getting standards at a much higher level, that is, really getting some of the languages standard, so that, in fact, people can move programs across machines. Now that there is an alphabet, mainly the ASCII alphabet, we could standardize on message protocols. To this extent, we are proposing that the DDCMP be accepted as a standard for asynchronous and synchronous message transmission across machines, because we see that as a problem as opposed to starting out with new hardware and making everybody get new hardware for machine-to-machine communication. This has the beauty of being able to do it today, rather than waiting.

DECUS to me is the main user interface. I value it from a development standpoint, and am happy to support lots of development people not developing for a week while they find out what they should be developing. We have a number of indirect kinds of input; e.g., can we service it; manufacture it; can we build it; can we sell it? All these act as boundary conditions, and there is not a sequential process as shown in Figure 1, but a two-way interchange. To a certain extent, a key ingredient of design in many cases, is that the actual people building the systems have to use them. For a time we didn't have that situation. For example, in the early days of the PDP-11 software, most of that software was done under the PDP-10 and one used a simulated environment of an 11. Pete VanRoekens is to be congratulated for getting his people to use their own products.

Another input is a cultural background. The easiest and best thing to do is to transfer people, to get

knowledge of applications. More exchange occurs this way than any other way, so, if for example, you have somebody who really knows your application, don't be surprised if we try to raid them, because hopefully we will do a better job for everybody. Dave Cutler's experience in process control made the RSX-11M operating systems the best available. We go on the other side and have the person who really has been deeply involved in applications work on the development of the next lower level "ring". A large source of our input comes from just transferring people and many of those people come from the user population.

In terms of getting an overall model of where we've been, I wanted to give you some indication of what has happened in the past and what may be the improvement of technology. For example, in semiconductors, it really isn't the computing industry per se that's driving things so rapidly. A lot comes from a very vigorous technology program. For example, the memory density (in bits) has doubled every year since 1962. Similarly, disk recording density over the period of 1962-1974 has grown 30% per year over about a 20 year life in terms of cost performance. Tape has even improved and that's been like 23-29% per year. Power supplies - well, they've only lost about 3% per year.

One has to trace these gains back to the production functions, and learning curves. The price comes down as more products of a particular type are built, and this has been observed on everything from light bulbs to liberty ships. As the volume is doubled, the cost per unit comes down somewhere between 10-20%, depending on what you're learning, and if there's been no history of doing it before it comes down quite rapidly. But, if you take a mix of technology, then there's a collection of things that determine that learning function. For example, packaging which also follows a negative 3% curve, depends on inflation and labor costs. Besides, we've bent enough sheet metal in our lives that there's not much to be learned in bending another piece. We've probably bent several giga boxes by now. The same way with things like automobiles - you don't get the rapid learning. Because several 100 million automobiles have been built, adding another 10 million that year doesn't really add to the cumulative experience, so you don't have much chance for improvement.

In mini's, in fact, the PDP-8 has seen about a 35% decline in the unit price. (See Figure 2) People say there's a mini-computer revolution - it was earmarked by somebody who, in fact, made the first market survey after the mini-computer was a long time established. It can very well be predicted. I only predict the past with my curves too, as we must be very careful with exponentials. Exponentials eventually hit some physical limit-micros, minis, midis - 4, 8, 12, 16-bit machines all behave alike.

The other thing that happens is that as we move down the cost curve we reach various thresholds and establish "machines". We then go along to produce machines at a constant price level. DEC as a whole is characterized in Figure 3. The PDP-15 has been a relatively constant price; the 8 has trailed down; and the 11 has gone both up and down in price. The 10 is the more traditional constant price machine which has, in fact, gone up in price. The actual machine size has increased with the 10. Essentially this is much more the conventional computing syndrome where one gets locked into a conspiracy between the builders, marketing, and users. The price tends to migrate up. If you look at a given machine and ask, "What is that machine price going to do?" people in Marketing make their forecast to sell a constant price machine because all the overhead structure is there to support it.

Marketing looks to see what the users really want at this time and they find for example that a user wants more memory in the same boxes, so that in fact the memory price or memory size per machine tends to increase over time. The 10 goes up in price - that's sort of the inflation of the salesman's salary because these are the one-big-bang-a-year type people. They're used to selling 1 or 2 machines and all the yields are calculated on that and it would throw the place into chaos if you changed the price. The engineers behave in a similar way - they want to build increased performance machines. The mini is probably the most exciting machine, because many more applications are uncovered as the price goes down, at each level of price many more applications are feasible. At a given price level, an application is feasible or not.

Some people will pay $10,000 for one in their home. I remember the first time somebody said they were going to put a computer in the home to help with the recipes - fortunately it wasn't one of ours. They showed a picture of this machine selling for $10,000. It had teletype with 10 char/sec paper-tape reader on it. I can't think of any way to turn people off to computing than having to use this. Besides there wasn't anything for it to do anyway. I mean, if you put a machine in the kitchen without the substantial amount of software that's required to support a machine in a home, it really is useless. I've had terminal access to a machine for maybe 10 years and really, it all comes back to whether I really have the time to do the programming. I've got a child and a wife that program a bit, but it always comes down to, "Can you help me with this program?" or "Gee, we could do this, if you would only do that," so I end up being the maintainer of several programs. We must work to make programming a lot easier; it has to get a lot easier before it has some utility in the home.

At one point there'll be a computer in a car. Then, at some point, we'll put one on every wheel - when that makes sense from a purely economic standpoint.

I was trying to get some perspective a few weeks ago of what's important in terms of machine use and relative costs. Figure 4 shows this. It is a table of costs per hour for people and the equipment and support costs of computing activities. That's for a human, and if you're in a university environment, they come cheap sometimes. But anyway, you can pick the number here on how much people cost; zero, five, ten, twenty, forty kilobucks per year. If you look at a computer, it's very hard for a machine to be over 1.2 to 2.5 kilobucks per year. Something like the CLASSIC or the 310 that was just announced, the whole machine is of that scale and those are relatively good matches for people, depending on the problem.

The surprising thing to me was the differentials. Very often when we design we have conflict in terms of our designs because we assume that purchasing agents are buying all the equipment (and very often they are). We essentially take everything out of a

system somebody could want and come out with a lower price. Then you get into this phenomena of why it's very low cost, low price, but to use it turns out to be extremely expensive. I think that's something that one finds out with terminals, going from 10 characters/sec to 30 cps really increases cost/performance when considering human costs.

The past has been exciting, we've made many gains as researchers, providers, and users. The technology of the future looks just as exciting and I hope we can continue in the same spirit and determination. We've a fine record.

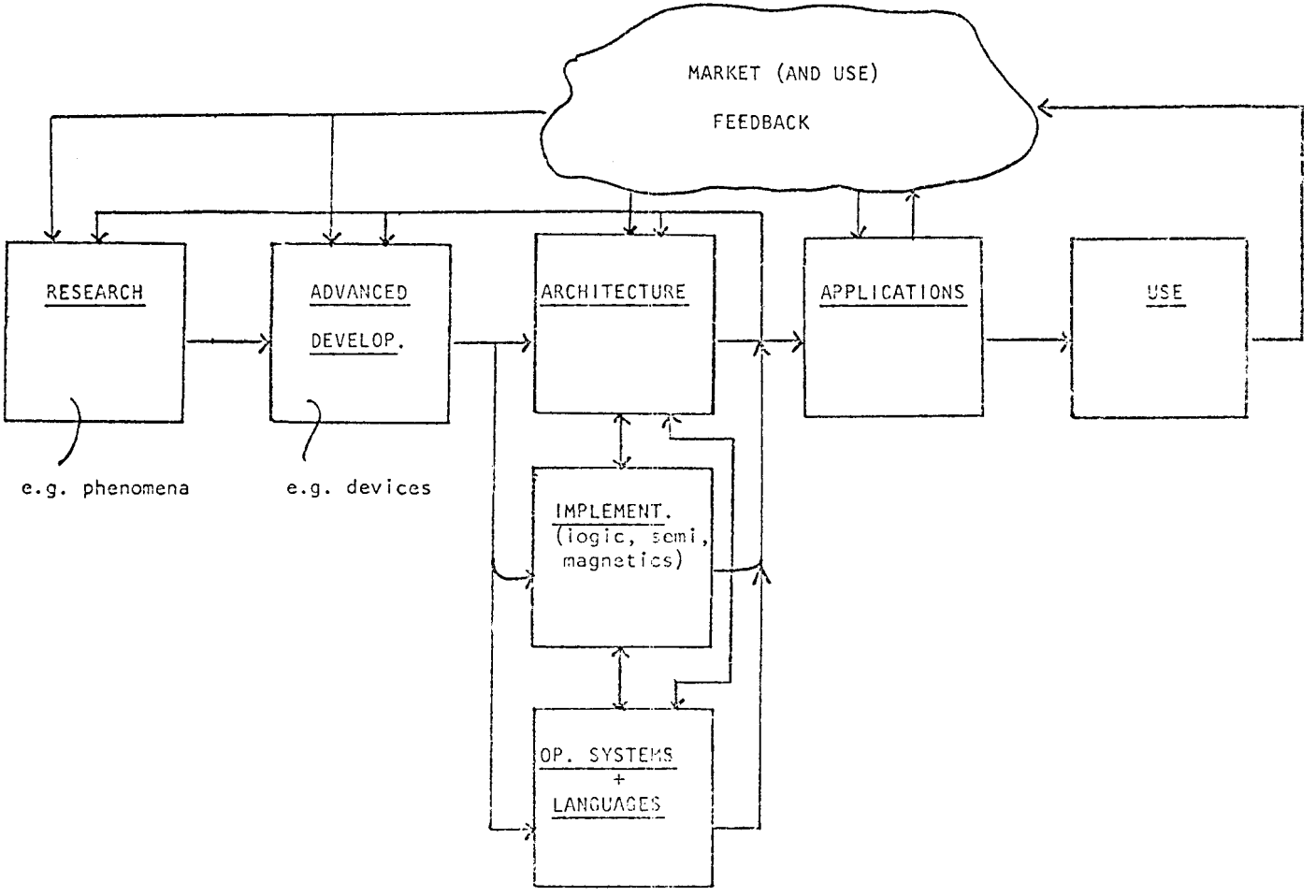

Figure 1. Computer Industry/Use Process

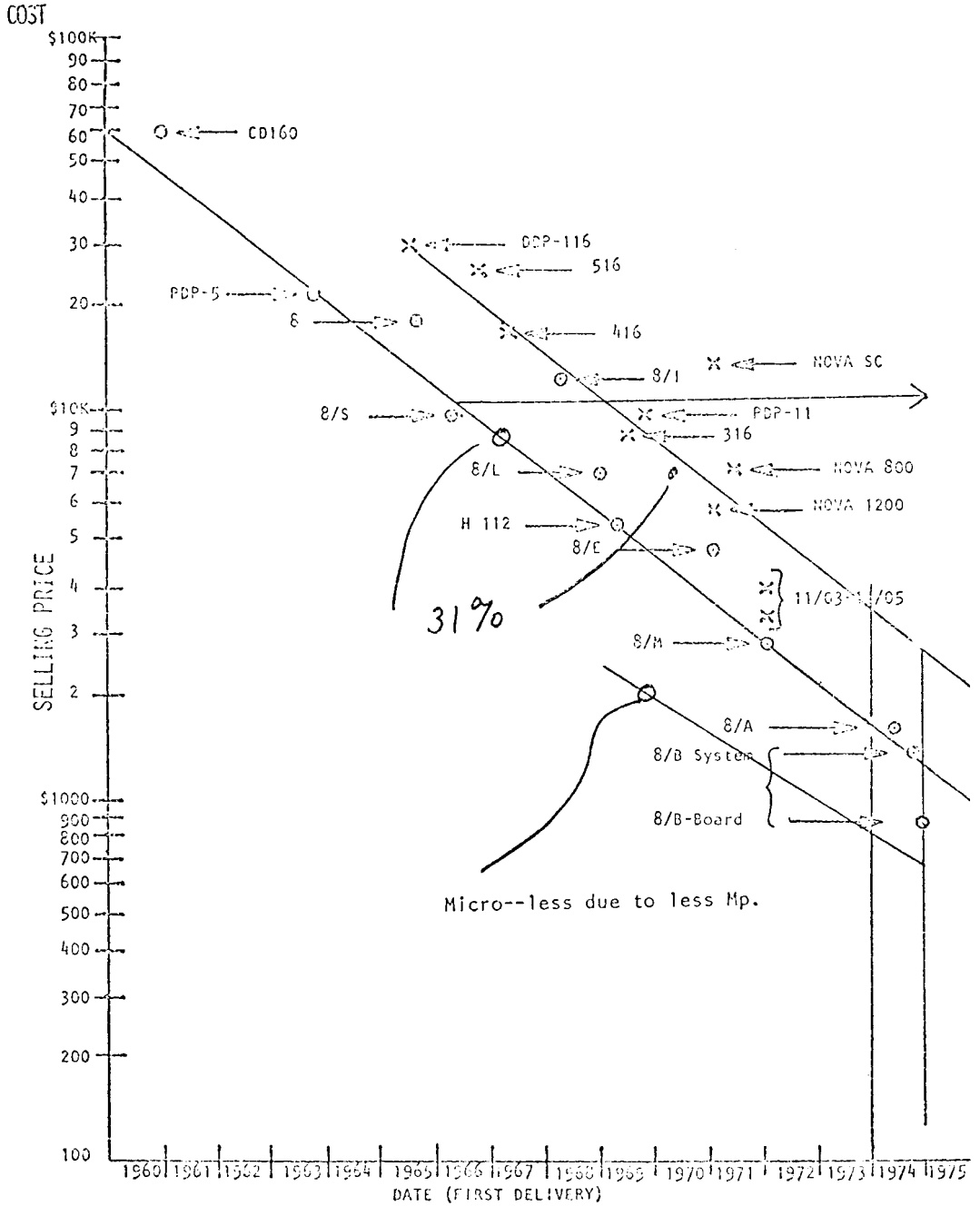

Figure 2. PDP-8 Price Decline

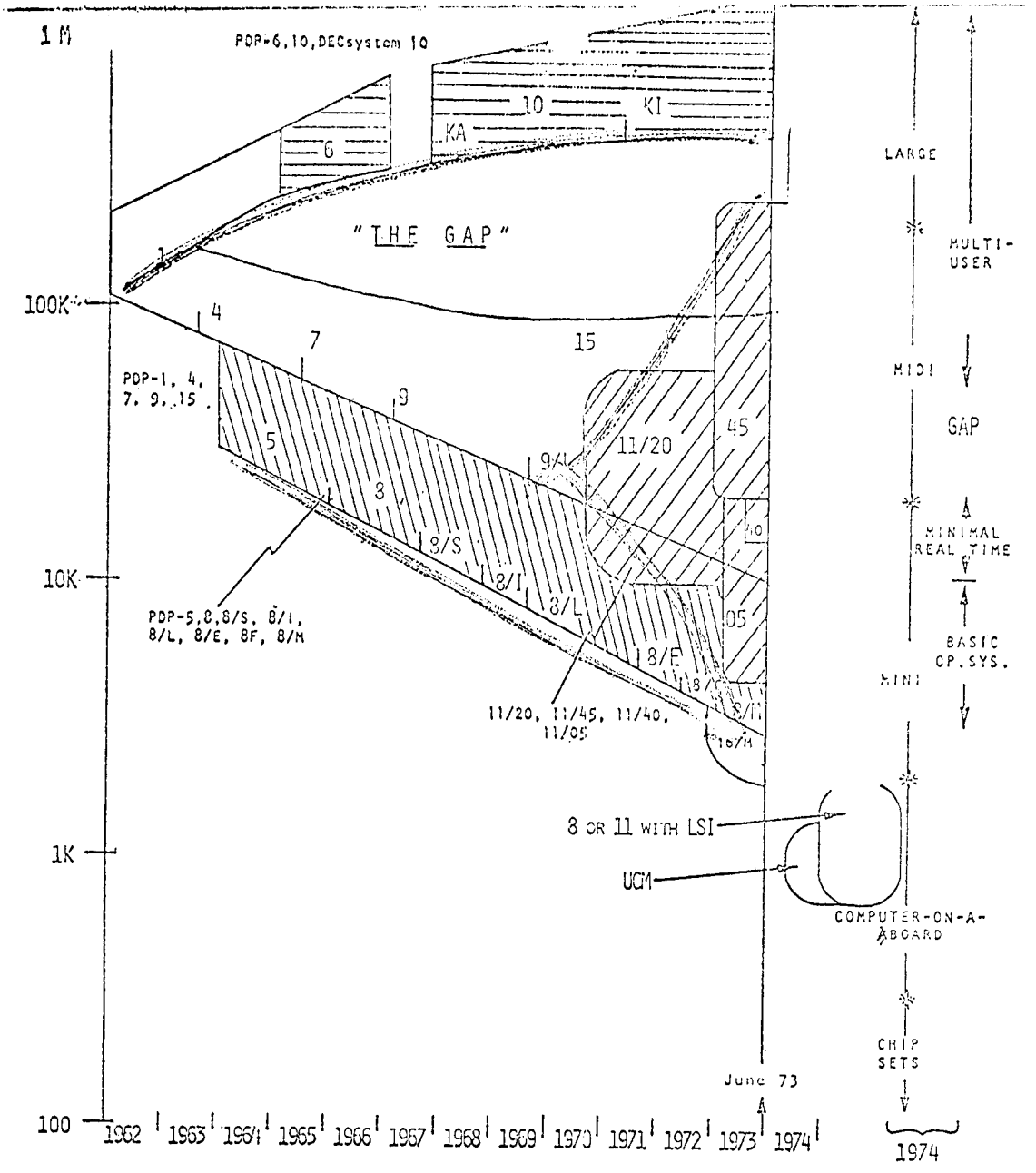

Figure 3. DEC Products vs. Time

OPERATING COSTS FOR COMPUTING

(1K)	COST/Year	COST/HR. @ 2400 HR.
Human	0, 5, 10, 20, 40	0, 2, 4, 8, 16
Computer	1.2 ~ 2.5	.5 ~ 1.
Terminal	.25 ~ .75	.1 ~ .4
Service	.05	.02
Power	.005 ~ .01	.002 ~ .004
Line	0 ~ 2.4	0 ~ 2.
Paper	0 ~ .1	1/3¢ ~ 3¢
Space	.05 ~ .1	.02 ~ .04

Figure 4. Operating Costs for Computing