DR. KENNETH WILSON
CORNELL UNIVERSITY

Recently I have been in a learning process about technology transfer. The technology that takes place when you are dealing with computers is wholly different in kind and in scale when compared to any other form of transfer that I have been involved with. Let me give you an example that you will hear more about later. This hasn't resulted in technology transfer yet but one can see the possibilities. It's pretty hard to imagine how one would use a superconducting magnet of the kind they have here at Fermilab to make soap. On the other hand, Tom Nash of Fermilab has been analyzing the data analysis track finding programs that are used here. He has found after consultation with the computer science community that what they are doing is data base technology. Even the soap manufacturer has to keep track of how many kinds of soap he has and where they sell the best. This is just what data bases are good for.

This is what I mean when I imply that there really is something different in kind and scale for computing than virtually any other aspect of the technology transfer process.

First I will consider a definition of a supercomputer, then I will discuss the industrial supercomputer market, why it is necessary, the relation of industry and university in that market, and why I think that that market is presently too small for the health of US industry. I will then consider the barriers to progress in uses of supercomputers. Finally I will discuss what is happening in the university community to try to deal with these barriers.

What is a supercomputer? Well, a supercomputer in my definition (everybody has a different one) is a computer with high performance, targeted for scientific and engineering applications, and at least as powerful as the most powerful business data processing mainframe. The standard examples in magazine articles about the competition with Japan consist of the Cray machines which can be 10,20,30 times faster than a business mainframe or the CDC 205. Many of these stories fail to mention that our moderator, Burton Smith, is also the architect of a supercomputer which is in fact much more original than the Cray or the CDC machines. There are a variety of other computers of different kinds that fit the definition that I have given. This is one of the difficulties in dealing with the subject. It is not simply Cray and CDC anymore. For example, the company that I have worked most closely with, Floating Point Sytems, makes what are called array processors. I won't have anything more to say on what an array processor is except I've carefully crafted the definition so their products fit too.

Why are supercomputers important? Let me step back for a bit and let's look at what is happening today. Industry, all the non-defense industry and to some extent the defense industry too, faces shortened time scales for research and development. It's no longer true that one can develop a product and expect most of them to last 20, 50, or 100 years on the market before a replacement has to be designed. There are two reasons for these shortened time scales. One is that the impact of computers makes possible the shortened time scales with computer-aided design and computer-aided manufacturing. Now we even have the computerized

office to shorten the time scale for all the paperwork that accompanies any new product. The other reason is the conversion from regional or national markets to world markets. This is driving the intense competition which is the business climate today. The businessman must take advantage of the shortened time scales to play a role in the market.

One consequence of the shortened time scales and accelerated rate of change is that the organization outruns internal experience with the product, with product design, with materials that go into products. There is not much use having 20 years of experience in how steel behaves if one is switching to a new composite material. When a business is in that situation it has to change the basis of manufacturing rapidly. It is a situation where basic science must often be substituted for experience.

Now one of the ways of using basic science in an industrial setting is through computer simulation, simulation of how a product will behave. This is totally standard when the science is just Newton's laws, the process of structural analysis for a bridge, let's say. No one goes out and builds a bridge to know whether it's going to hold up or not. The analysis is done in advance. The use of computers to do structual analysis is now totally standard. There are a variety of fluid and gas flow situations that are handled by simulation. A lot of the design of aircraft now involve simulation of the flow of air past the wing and other parts of the aircraft. Fluid flow simulations are heavily used in the oil industry especially when they are trying to understand what the oil does when they try to recover it. Not that those simulations are terribly successful all the time. The

area where one would dearly love to use simulation, and it is useable in some circumstances but not all, is for microscopic problems, or problems at the molecular level. This is for the properties of materials, the equations of state of various kinds of substances and so forth. That is in an area where a lot of basic science is needed but there are some things that can be done.

One example where microscopic science was involved that created a lot of problems was the question of a natural gas pipeline through Northern Canada. This had to go through the region of the permafrost. What nobody was able to figure out was how strong the pipe would have to be to withstand 20 years of changes in the permafrost around the pipe and the pressures that would be put on the pipe. That's not something where one can go out and do an experiment, because the pipeline was needed immediately and not in 20 years. That gets into the subject of soil science. It happens that the world's expert on frost in soils is at Cornell. That's how I learned about the problem.

Another example with which I am more familiar in this whole problem of progress outrunning experience in industry is the situation in the high performance computing business itself where the industry presently faces the problem of switching from sequential processing to parallel processing. The need for parallel processing is simple and obvious. It is just that components of computers are becoming extremely powerful because of very large scale integrated circuits, and extremely cheap. However as computing becomes cheaper, while every other aspect of research and development gets more expensive, industry has to put

a larger fraction of its R and D investment into computers. Even though a computer comes down to a cost of several thousand dollars, the industry has to spend many millions of dollars on computers. The most efficient way to spend those million dollars would be to buy lots of 1000 dollar computers and have them run concurrently. But the computer industry has absolutely no experience in how to design parallel computers, has no experience in how to persuade their customers to buy parallel computers, and the users of computers have no experience in how they would use such parallel computers. I should say when I say there is no experience in design, about the sole exception is our moderator here, Burton Smith, which is one of the reasons he is our moderator today. When I talk to industrial computer designers they make no bones about the fact that that is the problem that is preventing them from moving towards parallel processing.

Now what is the role of the universities? I am not going to talk about the role of the universities in the traditional way - advance research which then leads to industrial advance research which leads in turn to industrial research and development and finally a product 20 years later. That is the typical technology transfer process. This was true, for example, for the laser which was invented around 1959 or 1960 and went through all those stages before it became an extremely important part of industrial communications.

But when we come to supercomputers the role of the universities is in the supercomputer market. We have a role in research and development but, for example, when we deal with Floating Point Systems we go through the Vice President for

Marketing. We don't go through the Research and Development Group at Floating Points Systems. That is because we have a role in the market, and, of course, when one has a role in the market the research and development role is subsidiary to that.

What is the role of the universities in the market? The one that is most obvious to everyone is the training of people and specifically the training of people who know what supercomputers are good for and who know what they are not good for. It is perfectly clear in the visits that I have made to industry that they suffer a very serious lack of people with that kind of training. That is, people lack the training to recognize opportunities for the use of large-scale scientific computing in an industrial setting. That is one of the reasons that when you talk about large-scale scientific computing in the business community they have it pigeon holed. There are a few areas which everybody recognizes where one has to use large-scale computing. These include structural analysis, aerodynamic simulation, seismic processing in the oil industry, and circuit optimization. If the application is not on the list, then the assumption is that large-scale computing is not involved. That is nonsense. The problem is there is a lack of people who can identify new areas where large-scale scientific computing should be involved.

It is not only a question of training people. This is graduate training, advanced training. (An undergraduate is probably not going to be very helpful in this kind of problem.) It is also a question of consultants because a consultant could often help industry to figure things out. Unfortunately there are very few people in the universities who are qualified to be

consultants when it comes to large-scale scientific computing, because there is so little experience in the universities with large-scale scientific computers. I must warn you specifically that it will not do to go out and hire somebody with a Nobel Prize and expect they will be able to help you on that problem. Most of the Nobel Prize winners cannot help. You can tell it if you start talking to them about large-scale scientific computing and their eyes glaze over and they sort of look sour. That's not a person to hire. There has to be a revolution in the universities so that we start training not only the students but the faculty in the realities of scientific computing.

There are also other roles that the universities play. Up until now they have received very little credit for these roles. Unfortunately when they don't play these roles the market suffers and it suffers very severely. One important role is the testing and demonstration of new large-scale computers as they come off the assembly line. Universities are better suited than industry to take delivery of model No. 1 or model No. 2 off the assembly line than industry. This is because universities can live with the prototype and lack of software and all the other minor but exceedingly annoying problems that these computers have in their early stages. The way it's handled is that professors decide what is going to be done and students have to do the work.

Now one of the secret but very powerful resources that universities have is their undergraduate population. There is a fraction of those undergraduates who would just love to play with computers. They will sit for hours waiting for something to come back. It doesn't bother them; they don't cost very much; and it

doesn't hurt their morale to sit and wait. They can doodle; it's all a game for them. It's programmers in industry paid $30-40 thousand dollars a year and always under the pressure of deadlines that get disgusted when the computer doesn't work. Of course their managers get disgusted too. What that means is we don't want to think in terms of taking all those Cray 1's which are now obsolete, dumping them into the universities, and considering that a favor to the universities. Rather the newest supercomputers just coming off the assembly line, such as the Cray XMP, or Burton Smith's Denelcor HEP, should be put in the universities. Then the universities will start making use of them early. They will also start the training process on their students so that they will learn what those computers are good for.

I learned about all this because five years ago we bought an array processor at Cornell. We bought it simply because it was a nice, very cost-effective device for research. The first hint that made it clear that this was a different kind of interaction than we had seen before with industry was when potential commercial customers started calling us up and asking us questions like, "We heard you just got the Floating Point Systems' array processor, what do you want it for? Does it work? Is the software any good? What applications do you have in mind for it?" All of these are very practical questions that someone wants the answers to before they go out and buy a computer. In other words, we just acted as an information exchange. And of course they also liked to know what is the competition, are the competitors products any better, or any worse, and why? Because

they are open, the universities are a good resource for this. They are open for someone to call up and ask us questions. Nothing we do is secret and also we talk to all the other universities. For example, if we had the Floating Point Systems device and somebody else has the CSPI device which is Floating Point's competitor, we're likely to know what's happening with the CSPI device. That's just because universities are open and God help us if anybody makes this secret.

The universities are the best place for conceptual software development. I am not talking about software of industrial quality, but demonstrating what kind of software is possible. The classic example is the UCSD PASCAL system which was built at the University of California, San Diego, largely with the help of undergraduates. Some of these people are now fantastic computer scientists and programmers. This system became one of the main operating systems for microcomputers. It was of course developed a lot further when it went to industry, but it started at UCSD. One of the reasons to turn to the universities for this conceptual software development is nobody else has any time to do it. Computing manufacturers are stretched out getting their FORTRAN compilers and operating systems not only out but fixing and fixing and fixing them as people find troubles with them. The industrial users of computing are stretched out just getting their application programs done, so there is nobody left to think about very advanced software development projects. I will come back to that point later.

Finally there is the subject the universities are all about: basic scientific research. Now we don't have to do research in the physics department on Newton's laws, we already know Newton's law. However there are areas of extreme importance to industry which are also very fundamental subjects for basic research. One example is turbulence, turbulent flow. It's not possible to do all the aerodynamic simulations that one would like to do. As soon as the flow becomes turbulent instead of having a nice reliable accurate simulation, one has to go to phenomenological shortcuts which may or may not work. These shortcuts certainly can't always be relied on. Another area is microscopic physics which is just littered with subjects of basic research; problems with phase changes, problems of properties at the molecular level, the subject of quantum chemistry, all kinds of problems in crack propagation in materials and areas like that.

These are the subjects of basic research. The importance of the computer for these subjects is that it makes possible basic research on problems of more realistic interest to industry. In a fluid flow problem one no longer has to consider only the perfect sphere, which is usually not the problem that is faced in industry. Unfortunately as soon as one starts using the computer in these areas, one finds that enormous computing power is needed to accomplish anything. Indeed the closer one gets to nearly basic research, the more the computing power is required to be able to do things that cannot be done in the traditional history of analytic science.

Let me just give an example of why computing power is important. NASA uses reasonably heavy computing power to track its spacecraft. They can do that and get pinpoint accuracy as to where that spacecraft is going to go and that is important. But if someone wants to do weather predictions, simply a problem in gas flow, one finds that weather is not a homogeneous problem. It is not a shell which rotates around the earth. Little regions of air have to be looked at. It is necessary to see what each little region is doing. First one takes a region that is ten thousand miles wide and marks it off into little regions of 100 miles across so that there are 100 segments of 100 miles each to make 10,000 miles East-West. There are 100 segments of 100 miles each to make 10,000 miles North-South. Perhaps there are 100 layers going up through the atmosphere to cover the different wind velocities at different altitudes. Altogether that's 1 million cubic segments of air that have to be tracked. It's like having to track 1 million spacecraft. That factor of 1 million increase basically represents the total gain in computing capabilities between 1950 and today. Unfortunately the problems in modern physics are much worse than that. We would like another factor 1 million in computing capabilities, and we are starting to figure out how to get it.

What are the barriers to progress in the large-scale scientific computing area? First of all, it's important to know that presently in the academic community large-scale scientific computing has the lowest priority. It is a lower priority than equipment for experimental research, it is a lower priority than funds for more graduate students, more post docs and more junior

faculty. Because of this low priority, it has received very little attention from the federal agencies and the universities themselves. In fact there are very few universities which are presently sufficiently organized to take advantage of large-scale computing even if they were given the money to get it. This is because one soon runs up against this barrier of priority.

The reason for that low priority is the experience that was obtained in the 1960s with the computers that were available in the universities. In the 1960s typically a computer mainframe served the entire university and an individual researcher could afford to use only a few seconds of the time on that mainframe. It was not possible to do large-scale scientific computing with one second of time on those mainframes. These were the mainframes of the 1960s, not the mainframes of today. What happened instead was the faculty and university saw their graduate students getting interested in computing but just spending enormous amount of times struggling with all the difficulties of using those computers and not accomplishing anything. That message has sunk deeply into the minds of the faculty. It is very difficult to get that experience out of their minds again. Those feelings are combined, of course, with the problem that the entire society has a reaction towards computers and the changes they bring which is as visible in the universities as it is anywhere else. Of course it is the universities' problem to get that priority raised. However, any help from outside would be useful.

There are some more specific barriers. First of all, there is a communications barrier. In order to make progress in scientific computing, which does not make science any easier but just makes it possible to do more difficult science, it is necessary to have communications. These are communications between different disciplines, because the scientific computing problems are basically the same across all disciplines. These are also communications between different universities, communications with the computer industry. These communications have to be electronic. It must be possible to be able to exchange software and to have access to computers which are elsewhere. In addition, the universities have to be able to take advantage of all the benefits to ordinary communication that computer networks provide.

One of the most important features of these computer networks to exploit is to start networking universities with the research groups in industry. The reason for that is that when industry has a problem that might involve computing, it needs to find exactly the right person to help with that problem. There may be only two or three in the whole country who can help with a specific problem. It is necessary to find those two or three people. If someone else is found they will say that is not their subject.

Now what can be done on a computer network is to have bulletin boards categorized by subject. A request for help is put on that bulletin board. In the network culture what happens is that the experts scan the relevant bulletin board for notes that they can reply to because if they are not replying to those

requests for help they will not be known as experts. Unfortunately, most people don't get the experience and training to be part of the networking culture. The main source of training has been the ARPA NET run by the Department of Defense. They have been very restrictive as to who could be on that network. The people who don't have that experience don't get into that culture and don't respond to network requests. In fact even in the computer science community the older computer scientists who are not trained to use the network culture, don't participate in it. So the networks have to be established and people have to be trained to use them.

One of the most important aids to technology transfer I know of is to get that network culture established between university and industry. That is, of course, not just computing itself, it is technology transfer.

The second barrier is the lack of training programs in universities involving computers. Overcoming this barrier means two things: first of all there needs to be massive use of computers in universities with everybody participating. This is not happening today. Second, in the case of large-scale scientific computing, there needs to be training of students in the management of large-scale software, that is training in computer science. It is not easy training to establish, especially when the programs for graduate students are already full and take too long. Nevertheless something has to be done about it.

Finally, of the barriers to progress, there is one that everybody recognizes as possibly the most critical. That is the software barrier. What I see in industry over and over again is that there is a mainframe bursting at the gills giving poor service and poor turnaround. It is running programs that are ten to twenty years old that have been developed and tinkered with and are now well-established. People have confidence in those programs and they are running them to death. Beside that mainframe is a supercomputer that is sitting idle for two reasons. First of all the programs running on the mainframe won't transfer to the supercomputer, because they are so targeted at the old mainframe that it is impossible to move them. Second they have got lots of new applications which should be running on the supercomputer and driving it to the wall, but their software isn't finished yet. That in a nutshell is the software problem.

The software problem in the scientific case comes down to the Fortran language, the programming language which is used to write that software. In Fortran it takes forever to get the programs up and running. The major reason for that is the nature of the language in which the software is written. The main problem with Fortran, for those of you who have some experience with it, is that when one tries to write up a scientific application and reduce it to Fortran to run in the computer, what's involved is a total scrambling of the lines of thought that go into that program. For example, a computer simulation is usually based on a set of scientific equations. However if one takes a Fortran program and tries to figure out what the equation is that it was based on one finds that the equation has been

ripped into hundreds of little pieces. It has been contorted incredibly to take into account the approximation methods that are used to put that equation on the computer. The equations are now scattered through a 60 page Fortran listing just like pieces of dust scattered around a room. Of course part of what takes so long is first doing that shredding process. But the most time-consuming part is understanding what that program is doing after that shredding process is complete. You can watch programmers leafing back and forth through those 60 pages of listings to try to find out what's going on. That puts a factor of 10-100 in the time scale to get those programs done and working.

As I said the only place where one can really hope to get this problem solved is in the universities, for nobody else has the time to work on the problems at that level. The question is how should scientific programming be done. At Cornell I have a project that is joint between myself and the Computer Science Department working precisely on the problem of a language for writing scientic programs. In this language, the equations would be in one place where they could be read. Once they had been read the programmer would go on to the other parts of the problem such as the numerical methods, the data structures, and the optimization procedures needed to make the program run really fast. This is especially important for these new architechtures on the supercomputer or the parallel processors. At Cornell we call this new approach the Gibbs project.

Let me conclude by pointing out what's happening today in the universities to deal with this whole question of large-scale scientific computing. The most important goal is to get people collaborating within the universities and the universities collaborating with industry, government and wherever else one has to deal with this problem. The collaboration must be across scientific disciplines; it must be between the scientist and the computer scientist. That is especially important. It must involve collaboration with the manufacturers of computers. This is beginning to develop. It also involves collaboration with the industrial users of computing.

In our dealings with Floating Point Systems we have tried to put ourselves in a situation where we act as a buffer between the commercial users of array processors and the manufacturers. For example, the manager of the project at Cornell is presently the President of the Users Group for Floating Point Systems. That's an ideal way of establishing ourselves as providing that buffer. We need to be in collaboration with the users of Floating Point Systems so that when we complain to Floating Point Systems about the way their products don't work, they listen to us because they know that we will just go to the commercial customers and get some support for everything we have to say. Of course their designers don't like to hear that things that they did are wrong so we have to have some pressure. You need collaboration with industrial users so that they get the benefits of what we learn about the products. Of course at the present time, since our primary need is money, we come out hat in hand for a little help for the services that we provide.

A number of us have been pressing for action at the government level. I was a member of the Lax panel last summer which reported the needs for large-scale computing to the NSF, DOD, and other government agencies across all sciences. The one sentence in that panel report which says that we were in danger of falling behind the Japanese in the large-scale computing area has received a lot of attention in the press. In fact there has been a major turn-around at high government levels between last July when the panel met and today. In the last two weeks there was an announcement from the White House that there would be a major effort within the Government to look at the problems of supercomputer needs and access of university scientists to supercomputers. I hope that that announcement really does get followed up. I would be grateful for help from anybody who can help maintain the pressure on the Congress and the White House and governmental agencies to move and to give higher priority to this whole large-scale scientific area than it has had in the past.

There are a number of hardware and software products in the universities. Some of them are discussed in more detail by the round table participants. These include hardware projects at places like MIT, Cal Tech, Columbia, and NYU. There are also software projects at many of the major universities. One other project I would mention from Cornell is that I am presently organizing all of the theoretical science at Cornell into one umbrella organization called the Theory Center to promote the collaboration across disciplines and to attack our computing support needs in common.

QUESTIONS

Question:

In some sense the world has moved in a direction of distributed processing rather than the supercomputers in part to tackle the multiproblem situation where a common data base is needed. Isn't that a more intelligent way of approaching the large volume of difficult problems rather than the alternative of concentrating on parallel processing?

Wilson:

What you hear at all computer conferences is about the micros that are being distributed in personal computers by the millions rather than centralized super main frames with lots of computing power in one place. Now, in fact, for large-scale scientific problems we need both. Just to give an example of the way we plan to proceed at Cornell is to start with a distributed network of super minicomputers. There will be one for each group of theorists. This might be a group of three or four faculty members and a number of students. All of these super minis will be networked together, with a typical local area network. We will hang array processors on that network or attach processors as though they were like the FPS 164 which is a general purpose number crunching engine at a reasonable cost. This costs about $500K for a nicely loaded system. We will simply add more and more FPS 164's as this demand develops. If we need 16 of them, we will have 16, if we need 32, we will have 32, so that nobody suffers from delays in turnaround. We do not want the productivity of our 500 theorists to go down just because the computer system is filled up.

We will then provide access to all kinds of very high performance computers, supercomputers of various kinds, but with the idea that those are for single jobs that have outrun the capability of our distributed network. These are for the jobs that must be like the tracking of a million space craft, which we cannot do on the 164. Maybe we can only track 100,000 on the 164. We will make our users compete for access for longish periods (weeks at a time) on whatever supercomputers we can get a hold of. We will not distribute that time democratically. Instead we will find the best of the projects that need extrodinary amounts of computing time and we will let them take over the supercomputer all to themselves like a personal computing system. And of course, when the supercomputers become cheap enough, we will just have enough of them for everybody.

Question:

As we go to larger computers, would it be possible for these computers to interact among each other in such a way that we don't have to go in there and make the necessary corrections? Is the time going to come when one of the systems is larger than one human being can possibly handle?

Wilson:

The important trend in artificial intelligence is the trend to provide assistance to users who are in charge of the programming process. There's a project which I like very much at MIT called the Programmer's Apprentice. There they are not trying to do artificial intelligence in the sense that you tell the computer, here's my problem and an hour later $e=mc^2$ appears on the screen as the answer to your question. That's all a pipe

dream. What can be done and I think the artificial intelligence community is doing a lot of good work in this area is to make it easier for the human programmer. This can be done by giving him assistance in places where one can't figure out how to give assistance so that the programmer is not working almost at the machine language level when he is writing programs. And, of course, at a higher level what is happening is that people are now packaging programs. For example, the structural anaylsis programs are very heavily packaged. One goes to Swanson Associates or to Nastran where the program is already packaged and learns how to run the package. It isn't necessary to read the 300,000 lines of Fortran that lies in Nastran; nobody could do that. As we learn how to do work at the micro structure level, there are again going to be packaged programs. There will be companies that will swarm just to do that packaging and make them available so that the industrial users will turn to these packages rather than having to do a totally do-it-yourself operation.

Question:

What is your view on the question of artificial intelligence?

Wilson:

Well, this is a country of 200 million people. First of all even if one decides that artificial intelligence has a higher priority, and one neglects the large-scale scientific computing especially the Japanese national project addressed specifically to large scale, super-speed scientific computing, it is still extremely dangerous to give the large-scale scientific computing

such a low priority that nothing gets done. As I say we have 200 million people and they don't all have to be doing artificial intelligence. The other thing that I would say is there is a lot of promise and not much performance in the artificial intelligence area. There are particular kinds of artificial intelligence which are extremely important. In our own Gibbs project (the software productivity project) we interact mostly with members of the artificial intelligence community because that is where a lot of the ideas we are going to use come from. On the other hand, it is clear there has been a big oversell in the last couple of years about artificial intelligence. You should be wary of this.

Question:

Would you care to comment on the impact of the MCC consortium?

Wilson

I believe the MCC is potentially extremely important. At the present time, I feel that it has far too little funding and far too little force behind it to have that much interaction. The critical questions are how it will grow, how fast it will grow, and how adventurous it will be in carrying out it's mandate. If it sticks to the statements as presently made, that it's just doing proprietary advanced research, then it won't really break out of that mold, especially in its interaction with the universities. Then it would be a rather minor force. If it does become aggressive in its relations with the universities and starts addressing issues that are not strictly proprietary

research issues but some of the issues that we have been dealing with in the universities on a global scale then it can be a powerful force on the scene. At the same time it must build support so it isn't a $50 million operation but a $500 million operation. I wish Bobby Inman the very best of luck in steering that corporation in the right direction and I hope he succeeds because it is important.

Question:

What do you see as the impact of IBM not being involved in MCC?

Wilson:

Let me describe my strategy with respect to IBM. I apologize to the IBM representative here. First of all, I myself can obviously have no impact on IBM. I talk to people at IBM and there are people who are on my side and people who are not. The people on my side are happy to talk to me and I do whatever I can with them. On the other hand what IBM responds to is larger forces than just one person. IBM responds to the market and the needs of the market, so the really important objective is to build up the large-scale scientific market so that IBM is motivated to respond to that market. The way we build up that market is by working with faster reacting smaller businesses. This includes organizations like Floating Point Systems, Cray Research, and Denelcor. We can try to build the culture of large-scale scientific computing around these projects and thereby build up the useage to the point where IBM reacts to that market. I expect that by 1990 that market will be there. I am

sure that in 1990 IBM will make more money from the sales of supercomputers than any other manufacturer, domestic or foreign. Perhaps they will blame me for some small fraction of that profit.

Comment:

Let me just assert that you don't have to convince top IBM management that this is of essential importance, they are convinced of that. The problem is that the IBM company is very large and its filled with people that don't know what they are doing, who are trying to push wrong actions in the name of scientific computing. IBM recognizes the need to keep people trained, the issue that you stated before. IBM needs to be salted more broadly with people who are skilled in understanding scientific computing so that they make the right decisions, so that they don't go rushing simultaneously in 90 wrong directions.

Wilson:

The problems I described they face too. But what I am saying is when it becomes an important business decision they find out how to do things right. I don't believe they really do things wrong when it has to do with business data processing. They don't do things wrong in personal computing because that's a big enough market so they have got to do it right and they do it right. I think they will do it right in the scientific market when the market is large enough. I don't know how they will do it; they may do amazing things in order to do it right and they may do all the things I said that need to be done, but I am convinced that when the market is big enough IBM will address

that market and will do it in a sensible fashion. So far it is not really large enough for them to do that yet.