CHANGE IN PERCEPTION OF GATEMAKER AS A FUNDAMENTAL STEP IN COMPUTER SCIENCE.

CHRIS ALEXANDER, APRIL 11, 1997

I would say the most serious obstacle to proper understanding between Sun and CES hinges on the perception of the Gatemaker Pilot.

In an initial memorandum (April 8) I have made it clear that the format of Gatemaker is based on extensive work in the psychology of the creative process, and that it is fact-based and empirical in nature, not primarily opinion-based.

The conclusion I draw is that the character of this program has a bearing not only on future generation of programs in the building/environment field, but that all software (including software designed to help developers design software) may, in all probability, also benefit from the kind of sea-change which is anticipated in Gatemaker. One may describe this sea-change, broadly, by saying that there are reasons for thinking that the character of the computer environment of the future needs to become more childish, and more human, if it is to help human beings genuinely extract the best of themselves, in tasks in the computer environment; and that this change may well affect activities which are apparently technical, not only those that one broadly classifies as "creative."

The misperception which exists, and which has existed as an undercurrent of opinion since the Aspen meeting, may be classified as a mistaken evaluation of four fundamental issues:

- **WHAT IS NEEDED TO IMPLEMENT BILL JOY'S PROGRAM**
- **CHARACTER OF COMPUTER SCIENCE**
- **CHANGE IN NATURE AND STRUCTURE OF SOFTWARE, NOT MERELY "ON THE GLASS."**
- **DEEPER CHARACTER OF DEVELOPMENT PROCESS**

I will take these issues one by one.

- **WHAT IS NEEDED TO IMPLEMENT BILL JOY'S PROGRAM**

Bill Joy, at our the meeting with John Gage and Greg Bryant in Berkeley (October 1996) stated his hope for a creative process, within the computer environment, that is more humane, and more like real creation than what is currently possible in the computer environment.

He repeated this wish several times during the Aspen meetings of March 1997. He expressed a hope for a type of program which would allow users to draw the broad

picture, not the details; for a program with an attitude comparable to the naïve use of crude sketching and hand motions in daily life, but realized in the computer environment;

What Bill, I think, failed to grasp, is that the realization of his vision requires deeper changes than he may have anticipated, and deeper changes than he may even feel comfortable with, when he first encounters them.

Thus, for example, he expressed (I believe) a concern that Gatemaker in the early pilot form, was inept and amateurish, that it was developed by people who were not "top programmers;" Mike Clary expressed the similar view that the pilot did not have the sophistication of the work Sun has come to expect from its developers. Dick Gabriel, too, was confused by this issue at first, and unconsciously found himself evaluating the pilot by surface characteristics of its appearance, its apparent childishness, by the lack of sophisticated computation, by the possible lack of sophistication of the code itself (which he guessed at), and by the whole manner and character of its presentation.

None of the computer scientists present at the Aspen meeting fully grasped, I think, that the requirements of Bill Joy's program -- namely, that the human creative process be supported by a more deeply intuitive computer environment -- necessarily imply that the resulting changes are likely to be suspicious-looking, disturbing, even offensive at first to the trained computer eye.

That is of course inevitable, when one thinks carefully about it, since Bill Joy's program arises from a dissatisfaction with the very essence of the present accepted computer software environment. The reason Bill has expressed this program, and his wish to solve it, comes about precisely because present software and present computer environments all lack the qualities he is hoping for. They all lack these qualities because the present culture of computer science has, within it, fundamental assumptions which cause the present over-technical nature of available software. One of these hidden assumptions is precisely that which is defined by the character, psychology, nature, of the computer environment, as it touches human beings.

To make the deep changes which Bill Joy is hoping for, require substantial changes in software and the way it feels, which will "hurt" intellectually. That is inevitable. The fact that Gatemaker hurts, in this sense (by seeming inept), and was viewed disparagingly by the Aspen group when they first examined it is – I believe – an indirect mark of its success.

- **CHARACTER OF COMPUTER SCIENCE**

I believe that the changes visible in Gatemaker go to the core of present-day computer science issues, and do not merely reflect questions about the screen or the "glass."

In discussions with both Mike Clary and Dick Gabriel about the April 8 memorandum, they expressed a realization that they had not at first grasped the experimental depth of the psychological program being followed by our work on Gatemaker, and then expressed a ready willingness to see that the user interface -- the surface of the screen -- might require big changes to make a proper user interface, and that this work might, then, have implications for all software.

However (and here I am guessing only) that their enthusiastic acceptance of the need for screen and interface changes (based on the types of argument presented in the April 8 memo), masked a hidden conviction that the computer science in the background, and the coding issues, have remained unaffected by the arguments of the April 8 paper. In effect, they were saying, to themselves, and perhaps to me, that the deeper aspects of computer science have remained as they are, even though it was becoming evident that one would have to tweak the screen and the user interface to arrive at a more humane computing environment.

I believe that the changes suggested and hinted at in the Gatemaker pilot, come from sources which are much deeper than they seem, and quite deep enough ultimately to change the field of computer science at its roots; not merely in the trivial (but important) detail of the human user interface.

My argument is fairly straightforward.

The issues which have driven the Gatemaker pilot, are matters of feeling and human spirit. The four volumes of The Nature of Order, establish, at several different levels, the fact that issues of feeling and human spirit, are connected to profound and definable structural aspects of the world. This touches the sphere of buildings, of course. But it goes very much deeper than that.

There is, implicit in the Nature of Order, a revised world view which is based on a new marriage of human feeling with precise thought about structure in the world (physics, biology, etc.). Within this marriage, questions of feeling and spirit arise from structural considerations; and these structural considerations arise from the inner mathematical structure of wholeness and its consequences.

The childish spirit, visible on the screen of Gatemaker, is an outward expression of this inner mathematical structure, and so reflects a very much deeper issue than it seems, which touches questions of structure at the core of representations of reality.

These structural issues (concerning centers, structure-preserving transformations, and so forth) will inevitably affect deep structure of programs, too, because they affect the deep structure of everything. Once a decision is made, to find methods in software and in computer science, which go toward solving Bill Joy's program, it is certain that these methods will bring with them, changes in computer science itself, in the structure of programs: and will ultimately cause a revolution in structure of programs which might be

compared, for example, to the jump in structure that occurred when McCarthy went from SAP, other machine languages, and FORTRAN-like languages, to LISP. LISP was based on a radically different and more profound conception of structure. These next jumps and changes will be bigger. Very deep changes in structure can be expected, when one begins to write programs that have to do with centers, with wholeness, and with structure preserving transformations, and these changes will not be restricted to the computer screen.

I do not think that such developments are necessary to our immediate success. What is quite certain, in my view, is that implementation of Bill Joy's program, and solution of his requirements, cannot be confined to the user-interface, and will require changes in the deep structure of programs, too. The childish and more intuitive forms of expression and use, which come closer to experienced human reality, and which Bill Joy is looking for, will – ultimately -- require comparable changes in the structure of the programs that are used to write them. Early versions of the new wholeness seeking software which we aim for, can be written in contemporary code. However, I think it likely, and perhaps inevitable, that a new generation of languages and development principles based on centers as key structural concept will make C++, in retrospect, look as crude as FORTRAN looks to us today.

Thus the apparent childishness of Gatemaker is very far from what it seems. Its character, its apparently informal and naïve nature, express deep changes of attitude which one day revise computer applications from top to bottom, including appearance, interface, software architecture, and key coding concepts.

- **CHANGE IN NATURE AND STRUCTURE OF SOFTWARE, NOT MERELY "ON THE GLASS."**

I come back, now, to the childish character of Gatemaker, and to the meaning of this childish character.

In a recent discussion with Dick Gabriel, I told him that this childish quality of the program, and what he had perceived as its ineptness, was done by us deliberately. We intentionally sought this childish character, because it is that seemingly childish character, ultimately, which distinguishes human beings, which provides the origin of value in human society and human feeling.

After hearing this, and especially after hearing that I had intentionally sought to create this childish feeling in Gatemaker, and that achieving it was one of the reasons for its functional success, Dick said to me: I hope you realize that is enormously important information. That changes the situation profoundly.

There is no doubt, that present day software is heavily technical in nature.

The superficial technicality of early programs, has now given way to a postmodern kind of technical nature, which appears more oriented to fun, to dazzle, to excitement. The graphics of MS-NBC are a good example. Here is the top sophistication of present-day software developers. It hinges on eye candy, motion, excitement, dazzle. It is certainly compelling, and may be judged a commercial success. People like watching it, and it draws crowds.

That is no doubt because Bill Gates' instinct for the commercial, has succeeded, and his nose for commercial success, coupled with postmodern notions of excitement, lead naturally to the type of environment which MS-NBC provides.

Although computer scientists might laugh at it, and try to shrug it off, I believe that the canon of sophistication in present day computer science, is guided by just these kinds of commercial considerations. So, when Mike Clary tells me that he (and Sun) are used to a higher level of presentation from their developers, and repeats to me, that he believes something like Gatemaker could be written in two or three days by capable developers, and would be much better anyway if done by top developers, he is worshipping, indirectly, the rather tarnished gods of Microsoft, and is too deeply influenced, I think, not by science, not by computer science, but by images of sophistication.

The issue is, ultimately, what is true sophistication? Is true sophistication to be understood as the razzle dazzle of MS-NBC, and is one's sense of computer science to be influenced by this rather shallow form of sophistication? Or is one to understand true sophistication as a form of code, and a form of user interface which deeply reflects issues of human spirit and human feeling, where a new generation of software – instead of pandering to MS-sophistication -- makes a commitment to support and nourish the real nature of human life and human experience.

That, if done, would change the world of software completely.

Since Microsoft has chosen the MS-NBC route, it seems possible that Sun might decide, quite deliberately, to develop, at least in parallel with its other developments, a new form of software which tries to connect, genuinely, with human feelings as they really are, thus going far beyond what Microsoft is presently able to contemplate.

This goal, once attained, would sweep the board. It would lead to a form of program which, in the long run, would place Sun far ahead of its competitors. And that would be for the simple reason, that this would then truly help people to become better people, and to live more meaningful lives.

It is highly significant – though it was not much discussed in Aspen – that people feel genuinely different when using Gatemaker, from the way they feel in other applications. They feel that a part of themselves has been extended, enlarged. And they feel comfortable.

Isn't it just this, which is the essence of Bill Joy's program? He is looking for a form of software, which connects more simply, and more naturally, to peoples aspirations. That cannot be attained by MS-NBC. Indeed, the apparent sophistication of MS-NBC is only a surrogate for the real thing, and as soon as the real thing becomes available, when packaged and distributed in an appropriate form, people will feel genuinely helped, more genuinely comfortable.

It seems to me quite possible that a new type of software, based on these principles, might give Sun a distinguishing edge. UNIX, up until now, has given Sun its strong differentiating position. In the light of NT and other operating system developments UNIX may be fading in its capacity to give Sun an edge.

I believe the possibility of writing software which is based , genuinely, on human feeling, can give a second generation Sun expansion, another, different kind of edge.

But to do this, it would be absurd to turn away from the core issues, represented (in barely visible form, yet) in Gatemaker. No matter how uncomfortable they seem, these aspects of Gatemaker represent part of what lies in our future.

That is, I am sure, what has brought Bill Joy to articulate the vision he has expressed repeatedly to us. And it is that possibility which now lies in front of us.

- **DEEPER CHARACTER OF DEVELOPMENT PROCESS**

These observations lead me back to the development process, which even now, in the light of the April 8 memo, may not be fully understood.

For some at the Aspen meeting (and we have heard the same view expressed by others in the computer science community also perhaps not familiar with the fundamental experience which was under investigation in our study), there was a tendency to compare our process with the familiar programming methods of "Rapid Prototyping." This comparison of course missed the underlying principles guiding the creation of our tool. It is this principle-based exploration which distinguishes this project from Rapid Prototyping whose weakness is a lack of underlying principles.

In essence, the main principle guiding the development, and guiding the objects which can be designed within the program, too, is that at each stage one tries to make the thing (whether it be a software, building, or gate) have more life. The evolving, and increasing life of the thing under development, is the single key principle which guides. It is the implementation of this principle which gets results.

Empirically, it is highly significant that in the environment of Gatemaker, people did in fact manage to behave in a truly ordinary fashion, and did things which they really liked. The depth of this concept needs further comment. Mention was made, for instance, of Kai Krause's software, and this was held up as a possible model. It is certainly ingenious and fun, and its morphing abilities are very playful. But that does not mean that the program enables the user to make things which are genuinely touching, or genuinely meaningful in any profound sense. They are merely a flashy superficial kind of fun. The user is not able, with that tool, to approach more and more closely, the creation of a living structure. Nor, I am certain, was Krause's software developed under a procedure which embodies this procedure. Yet, to the eyes of 1997, it looks very interesting, and a wave of the future.

I give this example, only to re-iterate the very great magnitude of the transformation which is needed, the character of the transformation which has been accomplished in Gatemaker, contrary to all appearance, and the very small contribution that can be made by futuristic programs which do not involve the fundamental shift in attitude we have been moving towards.

Gatemaker represents a different, and deeper, intent. And its implementation, though crude, must be taken seriously, as a step towards realization of that intent. The form and substance cannot be separated.

My view about this subject has not been changed by conversations with Dick Gabriel, during which he expressed the opinion that a good programmer could have written Gatemaker in about two days, once it had been specified. That procedure would not have generated the results we achieved, nor, in my view, would it have been possible. In a subsequent discussion, Dick has repeated his opinion, and Mike has repeated the same opinion, that the Gatemaker pilot could be reproduced in two days by a capable programmer. Although it is obvious that we should work with the best possible programmers and developers, and that will undoubtedly bring great benefits to what we are doing, at the core of this feeling, there is, mixed in, a factual point on which I believe they are mistaken.

It is worth adding a final note of explanation. The reason is that even the minor changes which would be reintroduced as the system was cleaned up, or re-coded, would introduce minor trace effects, whose effect on the larger whole would ripple outward, very much as weather patterns propagate from small disturbances, and one could not avoid having to address the same fundamental issues which were addressed in the process of developing the Gatemaker pilot, in a second version while the reproduced code was being written. Thus the problem of subtlety and sophistication inherent in Gatemaker, will extend to its careful reproduction, not only to its existence as a pilot. This follows from the theorems of process, partially expounded in The Nature of Order, Book Two.

- **ARTICULATED RESPECT**

There is a need to see Gatemaker, with its warts, as a first, very preliminary statement of a goal, give due respect to it, and to acknowledge its stature as a potential element in Sun's overall future development program.