Features

A VIEW OF PEOPLE-ORIENTED SYSTEMS

Hugh Smith
Visiting Professor
Annenberg School of Communications
University of Southern California

My object is to discuss what is meant by the notion of a "person-oriented computer system". Typically, we talk about desirable systems being "comfortable" and "responsive" in some way. It seems to me that there are three general conceptual ideas which must underlie responsiveness and comfortableness:

1. In some way the user has an understanding of the system, its purpose, and its behavior.

2. The system has a model of the user that relates to his/her purposes and behavior.

3. That during the process of communication they both modify their behavior towards each other.

This seems to be a reasonable conceptual definition of what are necessary constituents of a person-oriented system. However, in order to understand what system features this definition implies, we have to make a comparison with truly natural communications. To do this, I will describe some of the things which I think people have in their communication patterns. The model I will use is that of having a perfect secretary or a good research assistant. This will help to narrow down some of the attributes which define good communication, attributes which are seldom encapsulated in any of today's computer systems, and which may in fact be fundamentally impossible to encapsulate.

Natural Communication

People, of course, converse in a common language structure that defines their knowledge about objects in the world and also knowledge of human social behavior. In the computer science field attempts to provide "natural" language capability have focused on the former type of knowledge of the world. It has turned out to be very difficult to build such systems, even in very limited situations, because the variety of utterances possible demands a vast knowledge base. That difficulty has become the predominant focus of interest and has tended to obscure some of the equally important more pragmatic social factors that operate in communication. For example, the duties that a personal assistant fulfills in work transaction include:

Predictability - Each party understands the likely goals and workings of the other and their social and cultural background.

Reliability - Social and moral contracts exist in any human interaction for each party to fulfill agreed obligations.

Responsibility - Each party seeks to define the limits of the other's accountability should things go wrong.

Role Status - The above factors are frequently enshrined in an agreed formal workings relationship.

Confidentiality - Each party will try to protect his right to privacy while (hopefully) respecting the other's.

Learning - Both parties continue to learn and adapt to the other's behavior over the course of transactions.

Although these factors would not enter into a typical work dialogue, they carefully serve to define the scope of the communication. This is true for most human communications; however, when one of the parties is a computer system, difficulties arise in "building in" these background pragmatic factors. To put this in perspective, consider the possible dilemma of a doctor who is advised by a computer system to use a drug treatment which he suspects may not be totally beneficial for a particular patient. Should the doctor suspend his judgment because the system is "normally" correct, or follow it? If the doctor was consulting with another colleague, it would at least be possible to discuss

the pragmatic issues and resolve such matters as moral accountability.

In a somewhat similar vein, can a user "trust" a computer system to notice an error, restrict access to unauthorized information, understand intentions, etc.? None of these questions poses problems to computer specialists who understand both the working and the limitations of their system. However, these issues are very real to the end user who does not have such knowledge although he may be highly skilled in other areas.

If we really want "personal" computer systems, then we should be looking for systems which attempt in some way to attack these issues, if we think we can implement them. What I have observed in myself and at conferences like this is the dominance of technological fascination over these issues. For example, we say "We're developing SYSTEM 2 because SYSTEM 1 didn't do everything that we really wanted...we want to include more clever functions..." I don't think this works very well. Think of an axis where the horizontal scale is specificity, i.e., how specific some function is that you require in your computer world, and the vertical axis is smartness or complexity, smartness in the artificial intelligence sense of the word or complexity in the operations research sense of the word. It seems to me that there is a technological push in our society, in our funding programs, and in human nature to attempt to create artifacts that move successively higher up this vertical scale. We have, for example, simple data bases like MEDLINE and ERIC which use minimally complicated conceptual tools. Now, going up in terms of smartness above clever data base systems, we have inference systems which are intended for use in decision making, in computer-aided instruction, and in medical diagnosis. Such systems intend, in some way, to encapsulate the environment and help the user by advising on causes of action. The great problem in computer programming now is how to transfer one domain of discourse into another domain of discourse or one data base field into another data base field. To generalize ability in this area is very, very difficult. Further up the scale are the theoretical question/answering systems, systems which, if you ask them a question, will respond, "An interesting question, but you really want to know something else." These "impertinent" systems not only know about the world, but they know what is of value and utility to the person. I don't know of any impertinent QA systems, nor do I believe that it is fundamentally easy to achieve them. Finally, if you listen to the AI people, you'll find yet another level--the problem-solving domain. In the world of the future they will be completely general and will have complex problem-solving functions.

The point of discussing this schema is to focus attention on where research should go. In terms of cost benefit, it is perhaps time we think about halting the march of computer science up this particular vertical axis. There is very little research, at the moment, into the "place" of people in computer systems. What I would like to see is a specific research funding and a specific awareness of how to build people into systems such that you don't need to build into the systems themselves a representation of the complexity of the world. I think in terms of cost benefit it will be much more appropriate to look at how we build people into systems rather than how we can supplant them.

* * * * * *