WARD SHAW
Executive Director
Colorado Alliance of Research Libraries

Design Principles for Public Access

Basically, the problem of designing an information system for public access is the same as the problem of designing any kind of system, and perhaps can be stated as a question: How do we construct or plan that interaction of hardware, software, people, and data that will be most likely to lead to a predetermined good or goal? Ideally, we should have a fairly good idea of what the goal is, an understanding of the mechanism of the change required to meet the goal, and a comprehension of the characteristics of the interactions of hardware, software, people, and data, so that we may apply those characteristics to the design and control of the events necessary to cause the change desired. Traditionally stated, this means: define the output, define the input, and then invent a process that will transform the input into the output.

The trouble is that when it comes to the design of information systems, and particularly public access information systems, it is extraordinarily difficult to reach clear definitions of *output* or *input*; and, moreover, nearly impossible to define a process that will transform one into the other. That is why it is fundamentally easier to design a system to send a man to the moon by 1970 than to design an information system . You know where the moon is, and can learn a great deal about its characteristics. You know what a man is, and much about what his biology enables him to do; you know when 1970 is; and there exists a corpus of knowledge of mechanics that allows construction of a process that will get the man to the moon—that is, the problem is well defined and a solution is at least comprehensible. What remains is essentially a mechanistic exercise. However, in designing information systems, it is not enough to define input as those keys which a user strikes on a keyboard, and output as those records or

messages which appear on a screen—surely we must pay attention to those things, but they do not treat the whole system. Information, like most "-tion" words, is a process—specifically, that process by which people become informed. Therefore, input becomes "an uninformed person" and output becomes "an informed person." But those are very fuzzy concepts, and that is particularly true of systems for public access. The basic difference between systems for public access and systems for "private" access is that, in the latter case, the designer can assume boundaries and hence some specification of the "uninformedness" of the user. He can also assume a certain degree of understanding by the user of what the system is supposed to do, and through that, can reduce his problem to that of defining a data retrieval or manipulation system. But in the case of systems for public access, such assumptions are perilous at best, because many different users of many different degrees of confusion will attempt to use the system for many different things; success will be measured not by such easily quantifiable things as relevance/recall ratios or numbers of documents delivered, but rather by user satisfaction.

To confuse the issue further, we don't know much about the process of informing, that is, how uninformed people become informed people. We do know that the process is closely related to, or perhaps equivalent to, learning, but we also know that educational theory is notoriously inadequate in describing exactly how to cause learning to occur. To return to something tangible, consider some of the questions that ought to be addressed in designing a computer-based public access information system—for example, a catalog:

1. How fast should system response be? Should the terminal display messages at reading speed so that users can read along as the terminal is responding, or should messages appear all at once for the user to read at pleasure? (If you think this simple characteristic is unimportant, try an interactive system at 300 baud and then try the same system at 9600 baud. You'll likely find that the system "feels" completely different, and that this "feel" affects your perception of it and its utility. But which is better, for what kinds of interactions?)
2. How much data, in what detail, should be presented, and at what stage of the interaction? If you present bibliographic records, how many elements are necessary? When does the format get in the way of useful information?
3. How much should users by involved in or control the interactive process? In a catalog, should the user who enters "Mark Twain" be informed that the correct name is Samuel Clemens and that the system will proceed with Clemens, or should the system substitute Twain for Clemens automatically, or should (for pedagogical purposes) the

system require the user to acknowledge (or even retype) the change to "Clemens"? What is the specific effect on the informing process of each of these options?

4. When should the system quit? That is, at what point can the user be considered to be "informed" and how is this measured? If left to user control, will he choose to quit before he "should" (for example, before the system directs him to periodical resources in addition to monographs)? Or is it better to make the system totally passive in this regard?

Clearly, the problem of identifying principles of system design for public access is a difficult one, and we certainly do not know enough about public access information systems to generate very many "rules" that can be applied a priori to guarantee a "good" system design. How, then, do we proceed?

We really know only three ways to approach design. The most frequently used, and usually the best, is the algorithmic method. This method is characterized by the systematic application of rules or formulas which will lead to the desired results. But, as we have seen, in the case of public access information systems, we don't understand enough about information-seeking behavior and the informing process to use this method successfully. In fact, we will probably generate this knowledge only through analysis of the use of many public access information systems over a considerable time; and, if the analogy to learning holds, we may never understand the process.

The second method is simulation. This method consists of creating a model of whatever it is you want to learn about (in this case, an information system). The model must have two characteristics: (1) it must represent the real world accurately with regard to significant variables, and (2) it must be capable of being efficiently and repeatedly exercised. The process is to run the model many times, each time changing one of the variable values, to learn which changes produce desirable results and which produce undesirable results. The problem is that it is very difficult to identify the relevant variables in information-seeking behavior, and much more difficult to derive accurate and representative relationships among those variables. We simply do not have a very good model available, nor does it appear likely that we will get one soon.

The third approach to design is the heuristic method, and I believe it to be the most promising approach to the design of systems for public access. Essentially, the heuristic methodology is as follows: in a case in which the designer does not understand enough about the mechanisms of a particular situation to invent appropriate algorithms—e.g., the case of public access systems—he generates a number of statements of value, or targets, which he believes will lead to desirable results. These statements of

value are called heuristics, and perhaps the most common examples of systems developed using this approach are many of the chess-playing programs. It is clearly impractical to design algorithmic chess-playing programs. There are too many possible moves for even the fastest computer to evaluate. So, heuristics are developed—for example, "it is desirable to take your opponent's pawns," or "it is desirable to trade pieces of lower mobility for pieces of higher mobility," or "it is desirable to move powerful pieces quickly to the front." These moves will not guarantee that the program will win chess games, but most players believe that they are likely to lead to success. Thus, the designer develops his program to evaluate a few possible moves in any given situation, measuring their value by determining which of the heuristics they might satisfy. Value in the case of chess can, of course, be represented numerically, and the program can "score" possible moves to identify the "best" one.

A key element of the heuristic approach is feedback. Heuristics or value statements are developed and assigned a particular numeric value at the beginning of system use. The system, as it is exercised, monitors its success, and by tracking itself, modifies the numeric values or weights assigned to its various heuristics, seeking more frequent success. In this sense, then, the chess programs are self-modifying and "learn" from experience, becoming more and more likely to win. Of course, how good they get depends in considerable measure on how intelligently the original heuristics were selected, and it is therefore axiomatic that the software be designed for easy modification so that new heuristics can be added and useless ones removed.

Let's look again at the example of designing a public access catalog. We don't know exactly how to produce informed (or satisfied) users; that is to say, we do not understand the mechanisms by which uninformed users are transformed into informed users. Consequently, it is difficult to see how to apply the algorithmic method to the design of public access catalogs. We furthermore do not have a model of the information process that yields much confidence in its "goodness of fit," and so the simulation method does not appear very helpful. But we can identify heuristics—statements of value—about the behavior of an information system which might lead toward informed users, and if that is so, the heuristic method of design is probably the best bet. Remember that we are looking at the information system as a whole—not simply the hardware and software, but also the users and the data.

At the design level, there are many heuristics that might be identified. Following are a few examples, not intended to be exhaustive by any means, but rather illustrative. These statements may or may not be true, but do begin to describe a corpus of system characteristics which will lead directly to an initial system specification and a description of a process for system

evolution or "self-tuning" as described above. First, there are a number of things we might say about the interaction of users with the rest of the system.

1. It is desirable (i.e., will lead to information) that the system allow the user maximum entry possibilities. That means that, for example, it is better to have indexes to ten data elements than to have indexes to five data elements. Thus, if you are deciding whether to provide access to the bibliographic records by ISBN, it is desirable to do so. Note that these kinds of decisions involve trade-offs, and the set of heuristics or value statements provides a framework for formalizing, controlling and eventually enhancing those trade-offs.
2. It is desirable that the user be able to ask the system for help at any point in the interaction, and that the help supplied be relevant. This means that a general-purpose "help" file is inadequate. Explanatory information presented to users should reflect where they are in the interaction, the path they have taken to get there, and the substantial data they are working with. This will probably require some code to generate appropriate responses on a semi-individual basis.
3. It is desirable that the user control the interaction, i.e., the user should feel like he is driving the machine rather than the other way around. The idea is that the machine should be responsive to the user.
4. It is desirable that the machine react to the user at the user's skill level. Novice users may need considerable instruction and step-by-step guidance during the interaction, but skilled users should not be confronted with repetition of instructions they already know.
5. It is desirable that the machine respond with data that appropriately answer a user's need. Messages should neither overwhelm nor "underwhelm" users. For example, if a freshman asks for a book on American education, it is not helpful for the machine to respond with bibliographic listings of all 600 of them. On the other hand, if someone wants a technical discussion of Rommel's Africa campaign, a response directing them to a general history of war is likely to be less than informative.

These are examples of statements that will allow basic design decisions to be made from the point of view of the users of the system. All involve trade-offs, all may or may not be true, and all may prove to be impossible, but they at least provide a framework for design decisions.

Similarly, there are any number of heuristics relating to the hardware/software area, and examples of these are presented below. Because one of the important considerations in heuristic design is the ability of the system to change, the first four examples relate to flexibility.

1. It is desirable to structure hardware and software in a modular fashion. The idea is that pieces of the system can be easily modified without dire

consequences to the rest of the system.

2. It is desirable that constant values be passed to software as variable values. A specific case of this is table-driven terminal control, and the benefit is that the constant values can be easily changed to meet different circumstances.
3. It is desirable that message content and message form be separated. For example, in designing screens it is far easier to deal with one software unit which formats information and another which collects the variable data to be presented than to deal with both at once.
4. It is desirable that data structures be as flexible as possible. As soon as you have designed an on-line catalog, someone will come along and insist that you turn it into a circulation system as well. If, for example, you are restricted by design to fixed-length records of fixed-length fields, this becomes difficult to accomplish.
5. It is desirable that terminals have considerable graphics capability. This will lead to increased capabilities. Some kinds of data are best presented in text or list form, but consider the difficulty of describing a map over the telephone. In that case, a picture does the job far better. This is probably also the case with exposing the syndetic structure of a catalog to its users.
6. It is desirable to minimize the time the machine spends accessing Disk files. The most common critical, limiting bottleneck in on-line bibliographic systems is the time the machine spends accessing index records and bibliographic records from disks, and the designer will do wonders for response time by attending to this at the design stage.

These, then, are examples of heuristics that, taken together and expanded, will probably lead to successful designs. None are true a priori, but we believe that they may lead to successful design.

The process, therefore, is to select a set of heuristics, and carefully state them. These will describe characteristics of the hardware, software, data, and interaction which the designer believes will lead to a successful system. The next step is to assign some relative value to each statement. The result will be a mechanism for making design decisions and trade-offs. Then, using these valued heuristics, the system can be designed. It can then be tested and, based on the results, the heuristics and the values assigned to them can be modified. In a continuous process, the system implementation will be changed to reflect these changed values. This recursive process, it is hoped, will at some point stabilize. At that point, the heuristics and associated values will become potential principles of design for public access. The heuristic method, then, is a codification of and control mechanism for the rough-and-ready approach that says "put something up and then tinker with it"—and I believe it offers the best hope of developing and learning about successful public access systems.