Concepts that determine the user interface[1]

Dr. Andrew U. Frank
Department of Surveying Engineering
National Center for Geographic Information and Analysis
University of Maine
Orono ME 04469
(207) 581-2174 (Fax: (207) 581-2206)
FRANK@MECAN1.bitnet

1. What is the user interface?

User interface is apparently a technical term describing the interface between the system and the user, but clearly seen from the systems perspective. In a recent paper, Grudin argued convincingly that this is a misnomer, in the same class as 'casual user' (typically an expert at a given task, for example a lawyer, but one that is not interested in computers and how they work) and other terms used in this context [Grudin 1990]. Grudin indicates that the user interface should be considered broadly as all aspects of how the user (i.e. the end user in computer jargon) interact with the computer system. This does include considerably more than the command names or the color of the menus. It includes the user manual and all other literature the user sees, the training of the vendor, and the form of the output etc.

2. Why does the user interface matter?

In a first phase the major concern for GIS software designers was to find ways to construct programs that did (more or less) what people wanted on the hardware available to them. The user –planners, surveyors and the general public – stood in awe when a computer drew a map, often without sufficiently critically questioning whether the results were useful. This time has fortunately passed, as can be observed during the large exhibitions of GIS equipment, when every vendor shows very similar 'glitzy' screens and the public passes by, taking this level of achievement for granted.

Parallel to this, one might also observe a change in the attitude of reports about GIS installations [Croswell 1991]. In the early days, the 'champions' reported about the magnificent plans they had and how GIS would solve important problems and would contribute to overall improvement of their agencies' performance. They listed equipment bought etc.
The current focus has moved away from hardware speed (where some of the vendors would prefer to have the debate), away from functionality expressed as long lists of commands and towards solutions. Buyers at exhibits are interested in complete applications that demonstrate answers to their needs and reports are welcome if they show the real contribution to the organization. A recent research initiative of the NCGIA was centered around the concepts of 'use and value' of geographic information (not GIS). Preliminary results indicate that not all the factors that lead to adoption of GIS technology in an organization - which are the ones often discussed and studied - also lead to its contribution to the organization's goals - the value of GIS.

[1] Funding from NSF for the NCGIA under grant SES 88-10917 and from Intergraph Corporation is gratefully acknowledged.

The difference between buying a GIS, adopting GIS in an organization and the GIS contributing to the organization's goals, is not in the hardware speed nor in the lack of basic functionality in a GIS (most GIS offer by now the most fundamental operations with sufficient performance [Denmark 1991]), but rather it is in the effective use of these tools to solve the organization's problems. The user interface is one of the major contributing factors - depending how one defines 'user interface' it may be *the* factor. It is well known, that cost for training of personnel is often higher than the cost for the system (hard- and software). Training of a few people for a few month quickly adds up to a man-year, with a cost of $100,000 or more. The interest in user interfaces is therefore economically warranted.

Assuming that a GIS provides the functionality an organization needs (and this is most likely the case today), we posit the hypothesis that

the user interface is the most important factor contributing to the economic success of a system or its failure.

This is certainly justified by the observation that systems that are adopted and installed but not used are obviously economic failures because they cannot produce any benefits. Casual observations of word processing users - users that spend 50% of their working time with a given program - reveals always, that users are quickly satisfied with the few basic operations and (probably in their great majority) never really learn most of the commands. For the success of a program it is not relevant if a command is available but it is relevant if the user does know about it - it cannot produce a benefit otherwise. Oftentimes, the remedy is to provide more training or point to the manual that the user should read, but this first costs money (employe time) and is strongly resisted. Thus it does not address the issue.

The most important cost factor in using a system is - the time of the employees using the system. For example, the cost (not the salary) of an employee with an advanced degree is above $80,000 per year - a single GIS workstation, software included, is perhaps $20,000 per year (and is often shared by more than one employee). Again, productivity of personnel is crucial and the user interface certainly influences this. There was a study made by a renowned management consulting firm which indicated that productivity in comparable situations differed between an Apple Macintosh and an IBM PC considerably - the difference clearly attributable to the difference in the user interface (Command line interface vs. WIMP).

For tasks that are performed very often during a working day - repetitive operations that some workers perform hundreds of times - it is worthwhile to carefully optimize performance. There are methods known - key stroke model and later refinements [Card, et al. 1980, Card, et al. 1983]- that allow the prediction of the actual performance of real people, once they have mastered the system so they perform without error. This might be of importance for some parts of a GIS interface - digitizing tasks for example - but for most of the uses of GIS the problem is not at the level of automatic, repetitive steps but the solution of complex application problems.

3. Interface as communication

The individual using the computer program and the computer program must communicate in some form, exchanging information about the tasks and their results. This exchange is not a communication between equal partners - at least not with the current state of the technology. The behavior of the computer is fully determined by the program. The burden to adapt is on the human user. It is expected that they learn the language the computer uses (like an inflexible administrative agency - for example the IRS - where the clients are supposed to learn its jargon etc.).
Much attention has been devoted lately to details of the user interface. One of the most important achievements is the attention to consistency in the interface - mostly recommending that

similar actions or similar symbols have similar meaning. Guidelines have been prepared for different interface styles which indicate how buttons, menus and icons should be used and what symbols should be used consistently. It is clear - for example from the success of the Apple Macintosh - that this is an important effort with an enormous payback (it is rather surprising to see, that this is difficult to achieve and more often than not violated in important aspects - e.g. in MS Windows 3). Consistency simplifies the user's learning as he/she has to learn only one basic vocabulary for all programs, not a new one for each one. This could perhaps be likened to the grammar and the closed class vocabulary in a language: it provides the basic structure to express complex ideas with specialized vocabulary.

4. Communication relies on context

To understand (i.e. to interpret) the symbols exchanged in a communication requires a context established. In human communication such contexts are established by the circumstances of an exchange and the commonsense knowledge shared in a culture. It was estimated that to understand 1 fact in a dialog, about 7 additional commonsense facts are used (traupel, video), but human dialog patterns contain numerous methods to establish common context and feedback mechanism to assure the commonality of the context. This is not the case for user interfaces - the computer program does not contain provisions for dealing with situations where the context of the human user differs from the one the program assumes (some context sensitive help facilities provide reminders for the user of the definitions of terms used, thus asserting the communicative context). It is the user that has to learn the context of the program - i.e. the conceptual context embodied in the program by the programmer. This leads to a second hypothesis:

Programs embody a complex conceptual context that the user must learn in order to understand the program and the required input, the output and the documentation.

This is not so obvious for the standard 'office application' programs like word processing, where the program can rely on the common sense understanding of letters, paragraphs etc. (but observe the difficulties that result from discrepancies between the commonsense definitions and the technical definitions of a paragraph in a program like Word), but is required for complex applications like accounting - where one has to learn the accounting practice of the company - or GIS. Some vendors provide a system with an "Introduction to GIS" text, which explains to the new user not only what a GIS is but also the conceptual framework for GIS this vendor uses, and last but not least, the terminology to be used. This is an effective means to address this problem for novice users, who must learn all about GIS. It is not practical for people with a background in GIS (for example, from using another GIS) who know most of these basic principles, but expressed in a different conceptual framework.

5. Design conceptual framework should be a rational process

The conceptual framework of current GIS are the results of an evolution, not a deliberate design. They are more influenced by the demands of an implementation, the background of the software designers and programmers working on them etc. than the result of careful rational decision-making. As a consequence, they contain contradictions. The same term may mean different things under different circumstances and two different terms may describe the same idea. Most of all, they most likely contain concepts that are not really necessary to understand or perform the task from a user's perspective, but were included as they explain the internal workings of the program. It is important to note that the collection of concepts is not limited to the ones found in the 'user interface' of the program in its narrow sense, but includes all concepts that appear in the literature, user manual, training etc.

As the user has to learn all these concepts, the hypothesis that *the number of concepts in a system is related to the effort to learn the system*

is justified. It would therefore be worthwhile to analyse the conceptual structure of a program and redesign it carefully to reduce the total number of concepts. If a set of concepts is carefully designed, it should be possible to explain them in a relatively short document.
Programs are often constructed from the 'code' outwards - the user interface and the manual are the last effort - explaining to the user what the program does. This is of little interest to the user - he does not want to learn what the program does, but wants to learn how the program helps him doing his work.

Not only should the concepts be explained in terms of the user's task, but the whole program should be designed from this perspective - first the conceptual level design, then the user manual and the interface and then - last - the implementation. This is often proposed but very seldom done.

The 'casual' user that has to solve a complex problem using a program wants to concentrate on his problem. He is forced to translate the task from his application language and concepts to the command language and the concepts of the program. The closer the two, the easier the translation task (optimally there would not be a translation and the task is understood by the user in the conceptual framework the program uses). One could formulate a hypothesis that

users are more effective using a GIS if the task related and the GIS enforced concepts are similar.

6. Why is that of particular importance for a GIS

The discussion so far is generic for most information systems and not particular for GIS. A major problem of a GIS is the modeling of space and spatial objects. Every information system uses a data model and specifically, every GIS uses a geometric data model. The facts about the world and the tasks must be translated into this model, which is not always possible. The geometric models of the GIS and the spatial concepts of the users are varied and translation between them is very limited. This is different from, say commercial operations, where the conceptualization of an accounting system is quite limited and implementation comparable.
GIS differ most substantially in the geometric model and therefore in the spatial concepts that they can be used to represent. With the geometric model comes the appropriate operations (and other operations that are not compatible with the geometric model cannot be provided).

Today's GIS evolved from an ancestry of map maintenance systems, computerized systems to maintain a spatial data collection in form of graphical maps stored in computers. This is quite natural, as maps have served for a very long time not only as a means of communication of spatial phenomena, but as the prime method of storage of spatial data. Only with the advent of computerized spatial databases and flexible (cartographic) output programs has it become possible to sewer this linkage - the cartographic representation and the internal representation for storage and manipulation (including spatial analysis) are not necessarily the same or strongly related. This has consequences for the user interface, where I see still enormous reliance on 'map' concepts and a mixing of analytical operations and their cartographic implementation. It is probably not often that the goal of using a GIS is the preparation of a map, but more often some space-related decision that must be taken. We too often assume that a map is the appropriate information product that the GIS should produce - despite the fact that we know that a large percentage of the population cannot interpret maps. Other means of communicating spatial information from the GIS to the user should be found and the GIS interface should not rely exclusively on a map metaphor.

The geometric models of different GIS vary in small details and these detail are shown at the user interface. This contrasts with other areas of data management, where the interface is based on a 'reference model' of the data model - usually the relational data model - and the semantics of the operations are explained in these terms. The actual implementation may vary and may offer some additional features, but a user who has learned the reference model is most likely able to use the system effectively (SQL standard). It is important to recall that a standardization of the SQL query language is only possible, because it is defined in terms of the reference model. A similar effort is currently not possible for a spatial query language, because there is not yet agreement on a spatial data model (or a few spatial data models) as reference models. This is probably a very important research problem, which would have immediate benefits.

Goal:
Define a small set of generic spatial data models for reference purposes and define user interfaces in these terms (not in terms of the actual implementation).
Benefits:
- standardization of user interfaces
- reference models must necessary be simpler than the current (idiosyncratic) models

7. GIS and their interfaces are classed by spatial concept

In the absence of an encompassing spatial concept and recognizing the probable existence and concurrent use of multiple spatial concepts for different tasks, GIS must be built to deal with a specific spatial concept - the raster model (or Sarah Douglas' matrix, channel, regions model of space), the coverage model etc. Each of these models relies on an implied metaphor of varying complexity and for each a geometric reference model can be established. Interfaces for these reference models can be created easily and - I assume - will be straight forward. Conceptual interfaces for any of the 'pure' spatial concepts are easy to understand - it does not take more than a few minutes to understand the concept of Tomlin's map algebra and a number of very convincing visual interfaces have been presented [Kirby and Pazner 1990].
Then - where is the problem? The problem is in the attempts to stretch these geometric data models to make them include more than one spatial concept - arguably necessary to make them practically useful, as most realistic tasks use more than one spatial concept (typically planning requires an area concept (categorical coverage) and a networked concept for e.g.. traffic). I conclude that the most pressing research problem in user interfaces for GIS is to study
- how multiple spatial concepts are blended in actual use;
- how combinations of geometric models can be formalized; and
- how the interfaces for the single geometric models can be merged.
Realistic GIS are complex systems (and most current GIS prove that with their training requirements) - but they can be constructed from simpler parts. Mastering the complexity is the word!

REFERENCES

S. K. Card, T. P. Moran and A. Newell. 1980. *The Keystroke-Level Model for User Performance Time with Interactive Systems.* ACM Communications//vol. 23, No. 7, July 1980//pp. 396 - 410: copy.

S. K. Card, T. P. Moran and A. Newell. 1983. *The Psychology of Human-Computer Interaction.* Lawrence Erlbaum Associates//Hillsdale, N.J. :

P. L. Croswell. 1991. Obstacles to GIS Implementation and Guidelines to Increase the Opportunities for Success. *URISA Journal* 3 (1) : 43 - 56.

P. H. V. Denmark. 1991. Head-to-Head: A comparison of four MS-DOS desktop Geographic Information Systems: ATLAS*GIS, Land Track, MapInfor for DOS, and PC ARC/INFO. *URISA Journal* 3 (1) : 102 - 124.

J. Grudin. 1990. "Interface". In *CSCW 90*. 269 - 278. : ACM.

K. C. Kirby and M. Pazner. 1990. "Graphic Map Algebra". In *4th International Symposium on Spatial Data Handling*. Edited by K. Brassel. 413-422. Zurich, Switzerland: International Geographical Union IGU, Commission on Geographic Information Systems.