Fighting for the User

How to test and evaluate human performance with information systems

By Ben Shneiderman

Frustration and anxiety are a part of daily life for many users of computerized information systems. They struggle to learn command languages or menu selection systems which are supposed to help them do their jobs. Some people encounter such serious cases of computer shock, terminal terror, or network neurosis that they avoid using computerized systems. These electronic age maladies are growing more common, but help is on the way.

Researchers have shown that redesign of the human-computer interface can make a substantial difference in training time, performance speed, error rates, and user satisfaction. Information and computer scientists have been testing design alternatives for their impact on these human performance measures. Commercial designers recognize that systems which are easier to use will have a competitive edge in information retrieval, office automation, and personal computing. Human engineering, which was seen as the paint put on at the end of a project, is becoming the steel frame on which the structure is built.

But, an awareness of the problems and a desire to do well are not sufficient. Designers, managers, and programmers must be willing to step forward and fight for the user. The enemies include inconsistent command languages, confusing operation sequences, chaotic display formats, inconsistent terminology, incomplete instructions, complex error recovery procedures, and misleading or threatening error messages.

The battle will not be won by angry argumentation over the "user friendliness" of competing systems or by biased claims that "my design is more natural than your design." I believe that victory will come to those who take a disciplined and empirical approach to the study of human performance in the

Shneiderman is an associate professor of computer science at the University of Maryland, College Park. He has produced six books and more than 80 technical articles on human-computer interaction, database systems, and programming. He is editor for the Ablex Publishers series on Human-Computer Interaction. The first volume in the series, Directions in Human-Computer Interaction (A. Badre and B. Shneiderman, editors), has just recently been published.
use of interactive systems. More and more, system developers, maintainers, and managers are collecting performance data from users, distributing subjective satisfaction surveys, conducting informal field trials for novel proposals, and using field study data to support organizational decision making.

**Controlled psychologically-oriented experimentation**

Academic and industrial researchers are rediscovering the power of the traditional scientific method. They make sure that they

- begin with a lucid and testable hypothesis,
- explicitly state the independent variables that are to be altered,
- carefully choose the dependent variables that they will measure,
- judiciously select and assign subjects to groups,
- control for biasing factors,
- apply statistical methods to data analysis.

The classic experimental methods of psychology are being refined to deal with the complex cognitive tasks of human performance with information and computer systems. The transformation from Aristotelian introspection to Galilean experimentation which took two millenia in physics is being accomplished in two decades in the study of human-computer interaction.

---

**The classic experimental methods of psychology are being refined to deal with the complex cognitive tasks of human performance...**

---

The reductionist approach required for controlled experimentation yields small but reliable results. Through multiple replications with similar tasks, subjects, and experimental conditions generality and validity can be enhanced. Each small experimental result acts as a tile in the mosaic of human performance with computerized information systems.

Basic research in industrial and academic centers is beginning to yield guidelines for interactive systems designers. Industrial and governmental system developers employ empirical techniques by conducting informal evaluations of early prototypes, more careful studies of system components, rigorous acceptance tests, and continuous performance evaluation during the system's active use. If you are not measuring, you are not doing human factors!

**Evaluations during system development**

During the early design stages data about current performance should be collected as a baseline. Information about similar systems can be gathered, and interviews can be conducted with interested parties such as users and managers. Several design teams might make independent proposals from which the final design will emerge. Standards should be proposed for items such as

- menu selection formats,
- wording of prompts and feedback messages,
- keyboard, display, and cursor control devices,
- screen layout and use of multiple windows,
- response times and display rates,
- use of color, highlighting, blinking, inverse video, etc.
- data entry and display formats,
- command syntax, semantics, and sequences,
- error messages and recovery procedures,
- online assistance and tutorials,
- and training and reference materials.

Controversial standards can be reviewed by colleagues or empirically tested. Procedures must be established to distribute the standards, permit amendments, allow exceptions, and ensure enforcement.

Early pilot studies can be conducted using typewritten versions of screen displays. Instructional materials and command language designs can be evaluated with paper and pencil tests on typical users. As prototype versions become available, testing can be more elaborate. These preliminary tests help build confidence that the stringent acceptance test can be satisfied when the implementation is complete.

An acceptance test might specify that

30 typical users will be trained in using the system for 45 minutes. These users will be given 15 minutes to carry out the enclosed benchmark set of tasks. The average completion rate must be above 80 percent and the average number of errors must be below 3.

In a modest sized system there may be eight or ten such tests to carry out on different components of the system and different user classes. Other issues such as subjective satisfaction or retention of commands after a week may also be tested.

The goal of early testing is to force as much of the evolutionary development into the pre-release phase, when change is relatively easy and inexpensive.
Evaluation during active use

Gradual system dissemination is useful so that problems can be repaired with minimal impact. As more and more people use the system further changes should be limited to an annual or semi-annual system revision which is adequately announced. If system users can anticipate the change, then resistance will be reduced, especially if they have positive expectations of improvement.

Human engineering, which was seen as the paint put on at the end of a project, is becoming the steel frame on which the structure is built.

There are many productive avenues for assessing user performance and attitudes. Written user surveys are an inexpensive and generally acceptable approach with both management and users. A survey form should be prepared, reviewed among colleagues, and tested with a small sample of users, before a large scale survey is conducted.

Online surveys avoid the cost of printing and the extra effort in distribution and collection. Many people prefer answering a brief survey displayed on a screen, instead of filling in and returning a printed form. To keep costs low, the survey might be administered to only a fraction of the user community.

Interviews with individual users can be productive because questioning can focus on specific issues of concern. After a series of individual discussions, group discussions are valuable to ascertain the universality of comments. Interviewing can be costly and time consuming, so only small fractions of the user community are involved. On the other hand, the direct contact with users often leads to very specific constructive suggestions.

Online or telephone consultants are an extremely effective idea for providing assistance to users who are experiencing difficulties. Many users feel reassured if they know that there is a human being to whom they can turn when problems arise. The consultants are an excellent source of information about problems that users are having, suggestions for improvement, and potential extensions.

Electronic mail can be employed to allow users to send messages to the maintainers or designers. Such an online suggestion box encourages some users to make productive comments, since writing a letter may be seen as requiring too much effort.

By soliciting user feedback in any of these ways, the system managers can gauge user attitudes and elicit useful suggestions. Furthermore, users may have a more positive attitude toward the system if they see that the system managers genuinely desire comments and suggestions.

The software architecture should make it easy for system managers to collect data about the patterns of system usage, speed of user performance, rate of errors, or frequency of request for online assistance. Specific data provides guidance in the acquisition of new hardware, changes in operating procedures, improvements to training, plans for system expansion, etc.

For example, if the frequency of each error message is recorded then the highest frequency error is a candidate for attention. The message might be rewritten, training materials could be revised, the software could be changed to provide more specific information, or the command syntax might be simplified. Without specific data, the system maintenance staff has no way of knowing which of the many hundreds of error message situations is the biggest problem for users.

Rally round the flag

There is great excitement in the computing community as information systems are being applied in ever more numerous ways. The challenge of building appealing and effective user interfaces will be with us for many decades.

It’s easy to build just an ordinary system, but to build a really good system requires much more effort during the design phase.

It’s easy to build just an ordinary system, but to build a really good system requires much more effort during the design phase. The investment in time and money during design can dramatically reduce the development time and cost. Well-designed systems have lower lifetime costs, enable rapid task performance, substantially reduce error rates, shorten learning times, and bring satisfaction to the user community. Users who experience the competence of mastery, the confidence to explore novel features, the satisfaction of being able to perform their work, and the joyous sense of accomplishment will celebrate the role of the system designers, maintainers and managers. And that’s worth fighting for.