Issues and Opinions

User Friendliness

Friendliness is the level of tolerance built into a system that enables the user to cope with complexity and permits minor error and variable performance on the part of the user. In *Datamation*, D. Verne Morland records that the term "user friendly" is being widely applied to systems. "The suggestion that systems can actually befriend their users is attractive," says Morland, as "many users have come to believe that data processing should be more than a passive tool for the solution of predefined problems" [1]. Morland defines user friendliness in terms of eight characteristics which we have expanded, based on our experience, to provide guidance for systems builders.

Systems Design

Clarity of purpose. Each system should be designed with specific objectives in mind from its inception. Each part of the system should be clear and unambiguous in its function. The complexity of the system should always be less (for the user) than the complexity of the task.

Functional flexibility. The system should be able to support the user in handling most problems in well defined classes.

Unity of systems design. All systems components should be based on the same design principles and implemented as an integral unit.

Completeness. The system should be subdivided into reasonable tasks defined by the end user. The system should be able to handle any task, basic or advanced, or any combination of tasks which have been defined.

Error acceptability. The system should be designed and constructed with the bias that the user is correct; all user inputs, requests and actions should be interpreted and executed in this context [2, 3].

Training and Documentation

User manual. Clear user manuals, preferably prepared by the user and vetted by an analyst, should show the means for correct operation and a complete set of diagnostics and messages.

Training. The user should receive training in the use of the system.

User-skill. The system should allow users at every level of skill and experience to obtain useful results. The system must allow for relapses and ill-health of the user. Each system should have a minimum expectancy and graded steps to more difficult (or complex) accomplishments.

Test data. A system should have a sample of data with which to test the system, stage a demonstration, and train the user.

Self-teaching. As an extension of a demonstration facility, self-teaching facilities to guide the user through main facilities into more sophisticated functions should be available [4].

Aid in process. The system should show the user how to enquire about the logical sequence of operations and alternative actions that can be taken.

Questioning. The user should be able to question the meaning of any function, command, data, or operation in the system.

Welcoming

Easy access. The system should encourage use in physical and logical terms. It should not ignore a command. If it is busy, it should say so in apologetic terms.

Passwords. Privacy and password provisions should be explained before the user is permitted access to the system.

Importance. The user should be made to feel that his job is the most important task in the system.

User Working Dialogue

Conversation. The system should behave like a machine, not "chatty," overly friendly, or condescending. It should be possible for expert users to skip parts of the introductory level, primarily included for beginners and intermediate users.

Useful abbreviation. When options exist and the context is clear, the system should permit abbreviation of mandatory and optional keywords.

When entries are repetitive, commands may be omitted.

Diverse data. When ambiguity exists in referencing data, alternative spellings and synonyms should be presented as options. When hierarchical libraries exist, the hierarchy should be annotated with the option. When homonyms exist, the definition should be shown if the context is not clear.

User Errors

Error notification. The system should be sympathetic and helpful when it ascertains a user error. It should explain in clear terms why something is not possible.

Misspelling. The system should make every effort to match the user's intention. It should allow common transpositions and misspelled words. When ambiguity exists, possible choices should be presented to the user for selection of the correct one.

Error minimization. The system should not allow damage to be caused by a user error. The system should require only that erroneous data or commands be re-entered correctly. Potentially disastrous errors like deletion of a file should be reversible.

Dependability. The system should rarely break down or throw confusing surprises at the user. Error outputs for specialists should be clearly marked as outside the user's knowledge and addressed to the systems software team. The system should be durable, flexible, and seek alternatives — even to reconfiguration. It should flash messages of overload, embargoes, or degradation.

No scrutiny. The system should not tabulate the number of errors without permission of the user. No reports should be prepared on the user's competence.

System Response

Predictability. The system should estimate the time required for any job and should apply standard installation costs to give the user an idea of the resources to be used [4].

Feedback. The system should let the user know what it is doing, when it is experiencing difficulty, or doing a long task.

Security. The system should not lose work that has been successfully completed, even though some of the (preceding or subsequent) tasks fail.

Incompleteness. The system should not create unfinished business for the user. It should acknowledge completeness.

Isolation. The system should not make the user feel isolated, as though contact with the terminal is being substituted for personal contact and normal social interchange.

Maintainability. System enhancement should be made based on a periodic assessment of user comments. Any changes should be incremental, not revolutionary. Syntax and semantics should be carried forward between releases.

In sum, user friendly systems must be efficient, give good performance, be available when needed, and be reliable. It must be superbly documented. The system must be free of idiosyncracy. Maintainability is essential and the system must be amenable to change.

These user friendly characteristics represent objectives that should be clearly in mind before designing the system. We believe that the judging and design of all information systems, large and small, should be subject to a weighted evaluation of these criteria.

Ken Meyer and
Mike Harper
British Gas

Acknowledgements

The authors are obliged to Almos Kovacs and C.C. Change [4] for suggesting better structuring of the points.

References

[1] Morland, D. V. "Friendliness," *Datamation*, Volume 28, Number 2, February 1982, p. 224.

[2] Jackson, M.A., "Data Structure as a Basis For Program Design," in *The Michael Jackson Structured Program Design Technique*," Infotech International, Maidenhead, Berkshire, England, Booklet G-138, Issue 2, 1976.

[3] Jackson, M.A. *Principles of Program Design*, Academic Press, New York, New York, 1975, pp. 95-109.

[4] Chang, C.C., Butler Cox and Partner Limited, correspondence to K. Meyer, July 15, 1982.

Issues and Opinions Author Guidelines

The "Issues and Opinions" feature of the *MIS Quarterly* has the objective of providing a forum for communicating opinions concerning important issues in MIS. The feature will appear from time to time in the *Quarterly* in the position usually occupied by the "Editor's Comment."

I & O submissions should be no more than 1250 words (or, in rare cases, 2500 words). They should be written in a style which facilitates effective communication; academic and technical jargon is to be deemphasized and there is no requirement for references (unless they are necessary to accomplishing the objective).

Submissions should be structured so as to:

a) identify the "issue" in terms that are easy to understand,

b) offer an opinion and any evidence to support it,

c) precisely identifying the constituent elements and their relationships (the key objective of providing such a structure is to allow others to offer contrary opinions in an organized and precise way — i.e., in terms of the structure that you have presented).