Programming Interactive Dialogues

B.R. Gaines and P.V. Facey

Introduction

One of the most attractive features of modern computer technology is the opportunity for end-users to communicate directly with the computer-based system through an interactive dialogue at a display terminal. Many problems of the previous generation of batch-processing systems are immediately overcome in this way. However, new problems are generated because there are no human intermediaries between the computer-naive users and the interactive system. The responsibility for pre-programming the dialogue sequences is an onerous one with few guidelines currently (1). These notes are based on our experience of interactive dialogue programming for a variety of medical and commercial systems (2-4). They present some "rules" for the system designer based on trial-and-error which, however, have obvious bases in common-sense and user psychology.

Who Controls the Interaction?

There is a danger that the computer-naive user will be regarded as generally naive and incapable of controlling the dialogue with the computer. However, most users placed in this situation are so by virtue of their professional knowledge of the subject area. We use direct interaction to replace a chain of purely clerical operations between the end-user and the system. These end-users have detailed knowledge of the subject matter of the dialogue which it is virtually impossible to embody in a computer system currently.

Thus the ultimate control of the interaction must be in the hands of the user. It is helpful to envisage the computer system in the role of a (Dickensian) servant, courteously guiding and helping, quick to answer requests for information, often apparently more alert than the master, and yet ultimately knowing its place.

The User will Model the System

Again there is a danger of regarding the computer-naive user as a simple communication channel that may be identified and manipulated. However, all people rationalise their experience in their own terms and each user will model the computer system according to his experiences of it. This cannot be prevented and should be made as easy as possible. In particular acausal or time-varying factors in the computer system can create havoc with the users model.

Thus the random fluctuations in response time of a time-shared utility, or attempts to modify the form of interaction according to information being gained about the user are both sources of difficulty for the user in forming a model. Even worse are the effects of deliberate attempts to misinform the user - the use of ELIZA-type (5) tricks to make the system appear "intelligent" can lead to the users expecting far more of the system than it is possible for it to give. Effort should be devoted instead to making

the system simple for the user to model accurately and realistically.

Some Guidelines

The following rules have been described elsewhere (4).

Rule One - Introduce through Experience: Interactive systems are meant to be experienced, not talked about. Get prospective users onto a terminal on a related, or model, system before discussing their expected use of their own system.

Rule Two - Immediate feedback: Give the user feedback by making an immediate unambiguous response to each of his inputs. This should be sufficient to identify the type of activity taking place.

Rule Three - Use the User's Model: Use a model of the activity being undertaken which corresponds to that of the user, and program the interactive dialogue as if it were a conversation between two users mutually accepting this model.

Rule Four - Consistency and Uniformity: Ensure that all terminology and operations are consistently applied, and uniformly available, throughout all system activities.

Rule Five - Avoid Acausality: Do not introduce apparently acausal phenomena into the system. Make changes in the system clear consequences of the user's actions.

Rule Six - Query-in-Depth: Distribute information and tutorial material appropriately throughout the system to be accessed by the user through a simple uniform mechanism.

Rule Seven - Sequential-parallel Tradeoff: Allow the user maximum flexibility to make his responses holistically (in parallel) or serially (in sequence) according to his wishes.

Rule Eight - Observability and Controllability: Regard the system as an automaton controlled by the user and make it simple to observe and control. Provide a "reset" command which aborts the current activity cleanly. Print out the current state of a variable when requesting a new value.

Rule Nine - Validation: Validate data on entry by checking syntax and values, but beware of rejecting data, or querying too much as being outside norms. Have the user himself revalidate major updates before the system acts upon them.

Rule Ten - Use Documents: Do not assume that all users need to interact with the system. In particular, use high-quality document production rather than terminal interaction for information retrieval whenever possible.

Rule Eleven - Log Activities: Use the computer to maintain records of the system and user activities to evaluate the behaviour of both systems and users.

References

1. J. Martin, 'The Design of Man-Computer Dialogues', Prentice Hall 1973.
2. T.C.S. Kennedy and P.V. Facey, 'Experience with a minicomputer-based hospital administration system'. Int. J.Man-Machine Studies, Vol.5 April 1973, pp 237-266.
3. B.R. Gaines, P.V.Facey and J. Sams, 'An on-line fixed interest investment analysis and dealing system'. Proc. Eur. Computing Congress (EUROCOMP) 74), Online, London, May 1974, pp 155-168.
4. B.R. Gaines and P.V.Facey, 'Some experience in interactive system development and application', Proc. IEEE, 63, June 1975, pp 894-911.
5. J. Weizenbaum, 'Contextual understanding by computers', Comm. ACM, 10, August 1967, pp 474-480.