Human-Computer Interaction – INTERACT '84 / B. Shackel (ed.)
Elsevier Science Publishers B.V. (North-Holland)
© IFIP, 1985

DIALOG SHELL DESIGN

Brian R. Gaines & Mildred L. G. Shaw

Department of Industrial Engineering
University of Toronto, Ontario, Canada M5S 1A4
&
Department of Computer Science, York University,
4700 Keele Street, Downsview, Ontario, Canada M3J 1P3

Many rules have been proposed for dialog engineering effective human-computer interfaces. The underlying technology has been changing rapidly with the introduction of windows, icons, and natural language. It is not clear how coherent, complete and consistent are the various systems of rules, and how applicable they are to the new technologies. A systematic model of human protocols is needed where the principles and technology dependencies are clearly expressed. There is a need also for such protocols to be made available as application-independent processes, dialog shells, that implement effective human protocols. This paper gives a systematic exposition of the principles of dialog engineering, shows how these structure effective human protocols in different dialog technologies, and how this leads to the design of dialog shells.

1. INTRODUCTION

Since the early days of interactive computer systems the need to design effective human-computer dialog has been recognized. In 1967 White [33] discussed the problems of on-line software, the use of different error messages for novice and expert users, and the advantage of HELP facilities. As experience grew various authors presented guidelines for the design of effective human-computer dialog: Hansen, 1971 [16]; Wasserman, 1973 [32]; Martin, 1973 [19]; Kennedy, 1974 [17]; Gaines and Facey, 1975 [11]; Pew and Rollins, 1975 [23]; Cheriton, 1976 [4]; Gilb and Weinberg, 1977 [15]; Gebhardt and Stellmacher, 1978 [14]; Turoff, Whitescarver and Hiltz, 1978 [31]; Kidd, 1982 [18]. Shneiderman [26] has collected many of these rules together, and Maguire [20] has analysed some of them systematically, noting contradictions and proposing their resolution.

The number of guidelines for dialog engineering has grown over the years. Ours [12] started as a set of 11 rules for **programming interactive dialog** in 1975 [11], growing to 15 in 1978 [8], 17 in 1981 [9], 22 as a **human protocol** in 1983 [25], and 30 **proverbs** in our 1984 book on the **Art of Computer Conversation** [13]. They were first formulated for interaction through low-speed, upper-case teleprinters, but generalized well to visual displays and dialog through menus and forms. They appear to generalize adequately to dialog through windows, icons and mice, and through restricted natural language. However, the proliferation of rules, the development of new technologies for HCI, the growing maturity of studies of human cognition, human-computer interaction and computer communication protocols, the need for standard dialog engineering tools, and the emergence of commercial products to satisfy that need, suggests that time is ripe for a new approach to dialog engineering.

What we might reasonably ask for at this stage in the development of human-computer interaction (HCI) is a set of **principles** that systematically **generates** rules for dialog engineering. The principles should be **grounded** in **system theory, computer science** and **cognitive psychology.** The principles should be **applicable** to the entire range of possible dialog **styles and technologies,** now and in the future. The principles should be **operational** so that they can be embedded in standard procedures, **dialog shells** widely available for all interactive systems.

There have been a number of developments that satisfy some of these objectives. Thimbleby [30] has explored a number of generative principles for dialog design such as **dialog determination.** Nelson [21] has proposed **virtuality** as an analytic principle, noting that we project to the user a virtual world from within the computer. In his development of Pygmalion, Smith [28] propounds the creation of visual worlds of virtual objects and this became the design principle of the Xerox Star [29]. Shneiderman [27] has subsumed a range of such recent developments in display-orientated systems as examples of **direct manipulation.** Each of these concepts gives a logic for HCI design that is supported psychologically and operationally, and hence satisfies some of the objectives expressed above.

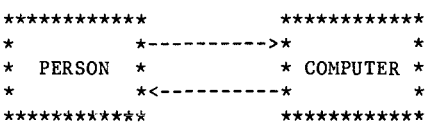
This paper presents a first attempt to satisfy all the objectives and provide an overall framework for dialog design that contains such developments as particular instances of the application of general principles.

2. SOURCES OF INFORMATION ON HCI DESIGN

The design principles that we apply to human-computer interaction are part of a larger context of studies of applied psychology, human and computer communication, and general systemic principles. Although they may require fresh validation in the context circumstances of HCI, they have richer origins and support than studies of HCI alone. This section explores the wider context systematically.

2.1 Person-Computer Interaction

The basic situation of human-computer interaction may be shown diagramatically as:

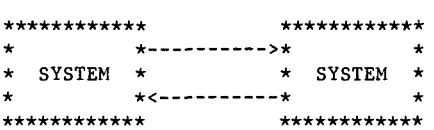


I. PERSON-COMPUTER INTERACTION

The primary analysis has to consider: the nature of a person; the nature of a computer; the interface from a person to a computer; the interface from a computer to a person; and the interaction between person and computer through these interfaces.

2.2 System-System Interaction

However, there is also the possibility of analysis through generalization, noting that people and computers are both examples of **systems** in general:



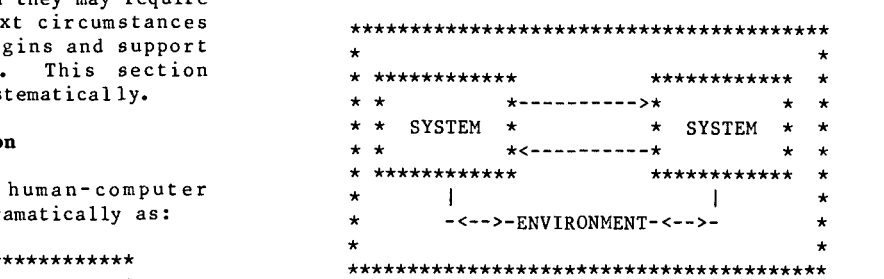
II. SYSTEM-SYSTEM INTERACTION
(Conversation Theory)

At an abstract level one notes that systems can interact in many ways and that there are constraints on the type of interaction that we consider in HCI. In particular, we would expect that at least one of the systems will be **goal-seeking** and that the satisfaction of its goals, and the cost in doing so, will give us a basis for **evaluating** the interaction.

The goal of one system may be to transfer information from the other through **communication**, to predict the behavior of the other through **modeling**, or to change the state of the other through **control**.

Both systems may be goal-seeking in which case considerations of **co-operation** and **competition** arise. The goal of one system may be to aid the other in, or to prevent it from, communication, modeling or control.

It is also possible to consider goals relating to the coupled systems that arise outside them and may not be manifest to them. In this context, we may wish to consider possible interaction of one or both systems with an environment:



IIa. SYSTEM-SYSTEM & ENVIRONMENT INTERACTION

There are many system-theoretic principles that have been developed for II and IIa which are instantiated in the situation where one system is a person and the other a computer. Wiener [34] emphasized this is his development of **cybernetics** as the study of "communication and control in men and machines", and many principles of communication theory and control theory apply directly to HCI. For example, Rule 9 (Appendix) that **the user should dominate the computer** derives from a stability-theoretic result in control theory, that two coupled systems with similar time constants may oscillate unstably around their intended equilibrium state: the person modeling a computer and adapting to it while the computer is modeling the person and adapting to him is a potential source of mutual instability.

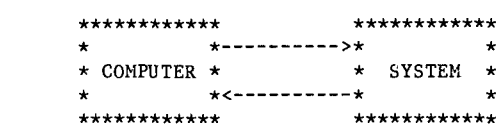
The relationship between dialog rules and their system-theoretic foundations is one of dialectical tension. Concrete rules are needed that can be applied within a range of contexts without excessive development and inference. However, when the context varies the rule may become invalid, indeed the negation of a rule may become valid instead. The system-theoretic foundation for the rule is then essential in order to enable the variant appropriate to the new context to be derived. There are situations where it is appropriate for the computer to dominate the interaction, notably ones where the person is not able to adapt, but where the computer is able to do so to improve the interaction. An example might be the indexing of very large databases that a given person accesses infrequently and does not learn to navigate. The computer might then adapt its portrayal of the indexing material to the nature of the enquiry without ill-effects [35].

The theory of system-system interaction most directly relevant to HCI is Pask's **conversation theory** [22] which considers the problem of systems validating mutual communication and of observers comprehending its occurrence. Pask gives systemic criteria for communication to

occur and has used these as a basis for designing person-computer dialogs in education through computer-based learning [22]. Conversation theory has been applied by Coombs and Alty [5] to the design of expert systems, by Shaw [24] to knowledge engineering, and by Begg [1] to the analysis of HCI in CAD systems. It provides basic systemic principles that may be used in the context of specific situations to generate rules of dialog design for effective person-computer communication.

2.3 Computer-System Interaction

If one of the systems in II is taken to be a computer then HCI can be seen to be analogous to interfacing a computer to another system such as a piece of equipment:

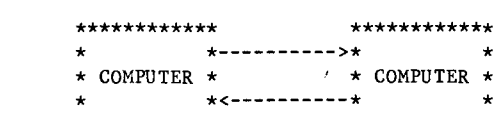


III. COMPUTER-SYSTEM INTERACTION
(Interfacing)

The design principles applicable to computer-equipment interfaces are well known and carry over to person-computer dialog. Problems arise because the system to which the computer is to be interfaced already exists and is not another programmable computer. We may to take it as it is and design an interface that copes with its peculiarities. Rule 2 to use **the user's model** derives from this, that the dialog engineer should identify the existing interface and attempt to emulate it rather than change it. Problems also commonly arise through noise at the interface and the designer attempts both to provide a low-noise channel and to provide error-detection and correction for unavoidable noise. In HCI such noise may arise through lack of clarity in information presentation giving rise to perceptual errors in one direction, mis-keying giving rise to errors in the other direction, and so on. Rule 20 to **validate data on entry** is a principle of communication over a noisy channel.

2.4 Computer-Computer Interaction

When the systems on both sides are specifically computers it is possible to define **protocols** that it is reasonable to expect any programmable digital system to be able to implement, and dialog engineering rules may be seen as defining a **human protocol** [25].



IV. COMPUTER-COMPUTER INTERACTION
(Networking)

The **Open System Interconnection** (OSI) ISO standard [6] is particularly interesting because it hierarchically structures computer-computer protocols for networks in a way that may have relevance for person-computer protocols. The concept of an **open system** is itself relevant because it expresses objectives for computer networks that are equally applicable to people using those networks. The aim is to allow integrated systems to be formed from multiple components not all from one vendor and not all installed at the same time [10]. The OSI concept is that the network is open to all systems that conform in their communications with certain well-defined protocols. In human terms the protocols may be seen as social norms for the behavior of members of a club; anyone may join provide they agree to conform to these norms.

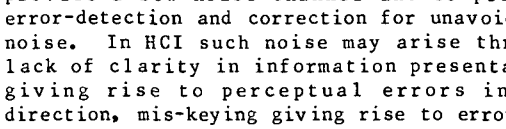
The OSI standard is hierarchical defining a number of layers, each with its own standard:

APPLICATION LAYER
PRESENTATION LAYER
SESSION LAYER
TRANSPORT LAYER
NETWORK LAYER
DATA-LINK LAYER
PHYSICAL LAYER

Each of these levels has an analogy in human information processing and communication, e.g. the physical layer corresponds to the audio-visual perceptual processes, and there is much to be gained in applying the OSI concepts to the human protocol.

2.5 Person-System Interaction

When one of the systems is specialized to be a person we have the classic case of man-machine interaction:



V. PERSON-SYSTEM INTERACTION
(Ergonomics)

Considerations of people interacting with equipment has been treated as a branch of applied psychology termed **ergonomics** that arose, under the same pressures as computer technology, out of World War II studies of pilots, gunners and so on. There is a wealth of results on general problems of human skills, training, its transfer between different learning situations, the effects of fatigue, and so on, that is immediately applicable to HCI. While interactive computers have been used primarily as programming and data entry systems these effects have not been major considerations. However, as computer-based interfaces become increasingly the norm for a wide variety of human activities the classic results of applied psychology and ergonomics

are becoming increasingly important. The novelty of the computer should not blind us to commonality with much earlier equipment. We do not have the time and effort to waste on rediscovering what is already known.

2.6 Person-Person Interaction

When both systems are people we have normal linguistic interaction from which the terms man-computer "conversation" and "dialog" have been generalized:

```
************           ************
*          *---------->*          *
*  PERSON  *           *  PERSON  *
*          *<----------*          *
************           ************
```

VI. PERSON-PERSON INTERACTION
(Linguistics)

Modern linguistic theory [2] has become increasingly concerned with the interaction between participants in a dialog, rather than a view of linguistic output as a predefined stream to be decoded. This provides a rich source of models for person-computer interaction, particularly as AI techniques take us closer to emulating people and their language behavior. There are also useful analogies of casual users in transactional analysis of the behavior of strangers meeting and **What do you say after you say hello?** [3].

Each of the five situations discussed provides a range of principles for interaction which are either directly applicable to HCI or, through analogies, have indirect application. One reason we have been able to develop effective interactive systems is that we can draw upon experience of these related situations. We can also use them more systematically to give foundations for dialog engineering.

3. DIALOG SHELLS

The considerations of Section 2 suggest that we consider a hierarchy of levels of dialog rule generation (Fig. 1): systemic principles; particular features of people and computers: consideration of their interaction: styles or technologies of interaction which lead to dialog shells; and particular applications.

The interaction level is interesting because it is where systemic, person and computer principles come together. For example, Rule 6 to **avoid acausality** has a system-theoretic foundation in that causal modeling systems generate meaningless models of systems with even slight acausalities. However, to apply this principle we have to know that people are causal modelers [7], and we would not regard the rule as significant unless we knew that time-sharing system generate apparently random delay distributions.

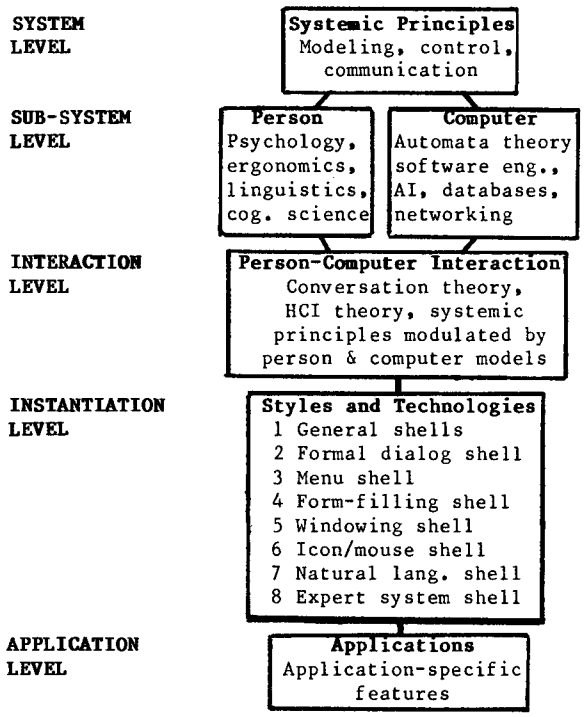


Fig. 1 Levels of dialog rule generation

At the instantiation level the rules are applied to instances of styles and technologies of interaction. To make this application-independent there has been a move to incorporate the rules in dialog shells that interface between the user and the computer system. The first general shells, such as IBM EXEC, originated from job control languages. We have given a shell [13] which incorporates the original formal dialog rules, and shells for form-filling transaction processing and menu-based systems are in widespread use. Several commercial companies have introduced window shells for the IBM PC and Apple Macintosh has a general icon/mouse shell. Intellect is a natural language shell for mainframes and Ask is one for micros. Expert system shells such as EMYCIN and AL/X are recent developments that incorporate significant new dialog techniques such as the uniform availability of **why** and **how** queries.

4. CONCLUSIONS

We are moving into a new era of HCI technology where well-designed dialog shells will provide uniform and consistent interfaces to a wide variety of applications. The systems of dialog engineering rules currently available can be generated systematically from basic principles that form a natural hierarchy. This paper has presented an initial framework for doing this which provides solid foundations for HCI engineering.

5. REFERENCES

[1] Begg, V., **Making computer aided design tools more useable: a study of a complex task shared by people and machines** (Kogan Page, London, 1984).

[2] Bennett, J., **Linguistic Behaviour** (Cambridge University Press, Cambridge, UK, 1976).

[3] Berne, E., **What Do You Do After You Say Hello?** (Andre Deutsch, London, 1974).

[4] Cheriton, D.R., Man-machine interface design for time-sharing systems, **Proceedings of the ACM National Conference** (1976) 362-380.

[5] Coombs, M. and Alty, J., Expert systems: an alternative paradigm, **International Journal of Man-Machine Studies, 20**(1) (1984).

[6] Day, J.D. and Zimmerman, H., The OSI reference model, **Proceedings IEEE, 71**(12) (1983) 1334-1340.

[7] Gaines, B.R., On the complexity of causal models, **IEEE Transactions on Systems, Man & Cybernetics, SMC-6**(1) (1976a) 56-59.

[8] Gaines, B.R., Programming interactive dialogue, **Pragmatic Programming and Sensible Software** (Online Conferences, Uxbridge, Middlesex, UK, 1978) 305-320.

[9] Gaines, B.R., The technology of interaction - dialogue programming rules, **International Journal of Man-Machine Studies, 14**(1) (1981) 133-150.

[10] Gaines, B.R., From word processing to image processing in office systems, **Proceedings of International Electrical, Electronics Conference and Exposition, IEEE 83CH1955-4** (1983) 622-625.

[11] Gaines, B.R. and Facey, P.V., Some experience in interactive system development and application, **Proceedings IEEE, 63** (1975) 155-169.

[12] Gaines, B.R. and Shaw, M.L.G., Dialog engineering, in Sime, M. and Coombs, M.J. (eds.), **Designing for Human-Computer Interaction** (Academic Press, London, 1983) 23-53.

[13] Gaines, B.R. and Shaw, M.L.G., **The Art of Computer Conversation: A New Medium for Communication** (Prentice Hall, New Jersey, 1984).

[14] Gebhardt, F. and Stellmacher, I., Design criteria for documentation retrieval languages, **Journal of the American Society for Information Science, 29**(4) (1978) 191-199.

[15] Gilb, T. and Weinberg, G.M., **Humanized Input** (Winthrop Publishers, Cambridge, Mass., USA, 1977).

[16] Hansen, W.J., User engineering principles for interactive systems, **Proceedings of the Fall Joint Computer Conference, 39** (AFIPS Press, New Jersey, 1971) 523-532.

[17] Kennedy, T.C.S., The design of interactive procedures for man-machine communication, **International Journal of Man-Machine Studies, 6** (1974) 309-334.

[18] Kidd, A., **Man-Machine Dialogue Design** (Martlesham Consultancy Services, BTRL, Ipswich, UK, 1982).

[19] Martin, J., **Design of Man-Computer Dialogues** (Prentice-Hall, New Jersey, 1973).

[20] Maguire, M., An evaluation of published recommendations on the design of man-computer dialogues, **International Journal of Man-Machine Studies, 16**(3) (1982) 237-261.

[21] Nelson, T., Interactive systems and the design of virtuality, **Creative Computing, 6**(11) (1980) 56-62.

[22] Pask, G., **Conversation, Cognition and Learning** (Elsevier, Amsterdam, 1975).

[23] Pew, R.W. and Rollins, A.M., Dialog specification procedure, **Report No.3129** (Bolt, Beranek & Newman, Cambridge, Massachusetts, 1975).

[24] Shaw, M.L.G., Knowledge engineering for expert systems, **Proceedings of INTERACT'84** (North-Holland, Amsterdam, 1984).

[25] Shaw, M.L.G. and Gaines, B.R., Does the human component in a network have a protocol?, **Proceedings of International Electrical, Electronics Conference and Exposition, IEEE 83CH1955-4** (1983) 546-549.

[26] Shneiderman, B., **Softare Psychology** (Winthrop, Cambridge, Massachusetts, 1980).

[27] Shneiderman, B., Direct manipulation: a step beyond programming languages, **Computer, 16**(8) (1983) 57-69.

[28] Smith, D.C., **Pygmalion** (Birkhauser, Basel, 1977).

[29] Smith, D.C., Irby, C., Kimball, R., Verplank, B. and Harslem, E., Designing the Star user interface, in Degano, P. and Sandewall, E. (eds.), **Integrated Interactive Computing Systems** (North-Holland, Amsterdam, 1983) 297-313.

[30] Thimbleby, H., Dialogue determination, **International Journal of Man-Machine Studies, 13**(3) (1980) 295-304.

[31] Turoff, M., Whitescarver, J. and Hiltz, S.R., The human machine interface in a computerized conferencing environment, **Proceedings of the IEE Conference on Interactive Systems, Man and Cybernetics** (1978) 145-157.

[32] Wasserman, T., The design of idiot-proof interactive systems, **Proceedings of the National Computer Conference, 42** (AFIPS Press, New Jersey, 1973) M34-M38.

[33] White, R.R., On-line software - the problems, in Gruenberger, F. (ed.), **The Transition to On-Line Computing** (Thompson, Washington, 1967) 15-26.

[34] Wiener, N., **Cybernetics** (MIT Press, Cambridge,Massachusetts, 1948).

[35] Witten, I.H., Greenberg, S. and Cleary, J., Personalizable directories: a case study in automatic user modelling, **Proceedings of Graphics Interface '83** (National Research Council of Canada, Ottawa, 1983) 183-189.

6. APPENDIX: DIALOG ENGINEERING RULES

6.1 General Principles

Rule 1 Introduce through experience: interactive systems need to be experienced before their operation can be discussed meaningfully. Get prospective users onto a terminal on a related, or model, system before discussing their preferred protocol for interaction with their own system.

Rule 2 Use the users' models: base the protocol on the users' models of their activities so that the interactive dialog is similar to a conversation between two users mutually accepting this model.

6.2 Continuing Design

Rule 3 Design never ceases: use the flexibility of the system through programming so as to close the adaptive loop between system and user through yourself as designer.

Rule 4 Log activities: use the network to maintain selective records of system and user activities and provide programs to analyse these in terms of, for example, errors, broken down by user and by dialog sequence.

6.3 Ease of Understanding

Rule 5 The User will Model the System: do not assume that the user is a passive static system to be controlled, modeled and directed by the computer. Evaluate the protocol in terms of its effect on an actively changing user who is attempting to comprehend the system.

Rule 6 Avoid acausality: make the activity of the system a clear consequence of the user's actions.

Rule 7 Consistency: ensure that all terminology and operational procedures are consistently applied throughout the protocol.

Rule 8 Uniformity: ensure that all terminology and operational procedures are uniformly available throughout the protocol.

Rule 9 User should dominate computer: either the computer or the user should dominate the interaction or there will be instability. If the computer is to dominate it must be programmed to, and have sufficient information to, model the user. If the user is to dominate then the computer system must be simple to understand. At the present state of the art the user should dominate the system.

6.4 Ease of Use

Rule 10 Parallel-sequential tradeoff: allow the user flexibility to make his responses holistically (in parallel) or serially (in sequence) according to his wishes.

Rule 11 Make the state of the dialog observable: give the user feedback as to the state of the dialog by making an immediate unambiguous response to any of his inputs which may cause the dialog to branch - the response should be sufficient to identify the type of activity taking place.

Rule 12 Provide a reset command: that cleanly aborts the current activity back to a convenient checkpoint. The user should be able at any stage in a transaction to abort it cleanly with a system command that takes him back to a well-defined checkpoint as if the transaction had never been initiated.

Rule 13 Provide a backtrack facility: that allows a user to return through the dialog sequence in reverse.

Rule 14 Provide default values: that are accepted by the user by , for example, returning a null message.

Rule 15 Make corrections through re-entry: use the entry dialog with default field printouts from a record as a means of correcting the record.

6.5 Help Facilities

Rule 16 Query in context: distribute information and tutorial material appropriately throughout the protocol to be accessed by the user through a simple uniform mechanism, e.g. responding with a "?".

Rule 17 Query in depth: organize the response to queries so that the user accesses brief memory aids first at any point but has further access to more detailed explanations.

6.6 Training

Rule 18 User manuals should be based on actual user dialog: illustrate the use of the protocol by showing actual dialog sequences that achieve specific objectives. Illustrate the structure of the system through dialog sequences where queries are used to elicit information about the protocol.

Rule 19 Train through experience: get users interacting with an actual system as their initial training and introduce them to the system facilities such as query-in-depth.

6.7 Error Detection and Recovery

Rule 20 Validate data on entry: check syntax, and values against norms.

Rule 21 Be flexible in validation: beware of rejecting data or querying too much as being outside norms. Allow the user to over-ride the error checking where appropriate.

Rule 22 Have the user himself revalidate major updates: show the consequences of major actions clearly before acting upon them and ask for user confirmation.