WIMPS AND NERDS: AN EXTENDED VIEW OF THE USER INTERFACE

MARK H. CHIGNELL
JOHN A. WATERWORTH

1. INTRODUCTION

Since the introduction of the Xerox Star computer and its more popular successor, the Apple Macintosh, the so-called WIMP interface has become predominant in software interfaces. In this paper we discuss the fact that the WIMP interface handles the physical, but not the cognitive interface to software. We propose to supplement the WIMP model of the physical interface with the NERD model of the cognitive interface. Like the WIMP model, the NERD model is designed to enhance interface consistency and to increase the power of the user while reducing cognitive load associated with the task.

After discussing the WIMP interface model, we introduce the components of the NERD model. We emphasize that the NERD model only handles the task-independent features of software applications. We also emphasize the need for further development of idiosyncratic interface components for task dependent features of particular applications. A hierarchical model of the user interface is presented that includes the WIMP interface, the NERD interface, and the TASK interface. This paper introduces the main components of the NERD interface. We do not seek to specify how these components should be implemented and standardized, since this will require programmatic research, just as the details of the WIMP model grew out of a fairly extensive research program carried out at SRI, Xerox PARC, and other locations.

1.1 The WIMP Interface

The WIMP interface specifies a style of interaction with computer software based on four main components:

Windows
Icons
Menus and
Pointers.

The WIMP interface has spread from the Macintosh to other types of microcomputers, and to UNIX workstations, with the development of systems such as X-Windows. The essential feature of the WIMP interface is that entry of commands in some (usually arbitrary) language is replaced by a "point and shoot" style of interaction, where the fundamental operation is a selection rather than a command. Furthermore the visual elements of a WIMP interface tend to be preserved across applications so that one WIMP interface tends to "look and feel" like any other WIMP interface (of course, this consistency of look and feel is not always fully achieved). Hence the saying, "All WIMPs tend to look and feel the same". Psychologically, WIMP interfaces provide the advantage of allowing the user to:

- visualize what is going on
- recognize operations instead of having to recall commands (recall is generally much more difficult than recognition)

- transfer knowledge about the architecture of the physical interface from one application to another

- achieve high compatibility between stimulus and response

High stimulus-response compatibility encourages a direct manipulation style of interaction (e.g., Shneiderman, 1987, Chapter 5). The idea of stimulus-response compatibility arose out of applied psychology (e.g. Fitts and Seeger, 1953) and was the basis of the human factors (knobs and dials) approach to physical interface design (e.g., Sanders and McCormick, 1987). Simple examples of stimulus-response compatibility in the WIMP interface include the mapping between mouse movements and cursor location (e.g, Card and English, 1978) and the use of a "thumb" on a scroll bar to manipulate cursor location in a scrolling field or window.

For more extensive treatments of components of the WIMP interface, see the chapter on menu design by Billingsley (1988), and the chapter on windowing by Paap and Roske-Hofstrand (1988).

1.2 The Physical Interface and the Cognitive Interface

The user interface is actually composed of a physical interface and a conceptual or cognitive interface, although most discussions of user interface design tend to focus on the physical interface. This distinction was implied in the work of Norman (1986) and was explicitly stated by Chignell and Hancock (1986). The WIMP interface is essentially a formula for handling the physical interface.

Our goal in this paper is to present the framework of a NERD model of the cognitive interface that provides the characteristics of consistency, transparency, and manipulability that are often ascribed to the WIMP model of the physical interface.

1.2.1 Elements of the Cognitive Interface

What forms of consistency are to be found in cognitive interfaces, and what basic sets of conceptual elements should be defined? We begin our answer to this question by assuming that there is a basic set of elements in conceptual interfaces, even if they haven't been formulated as yet.

In the following discussion we attempt to delineate the aspects of the conceptual interface which are task independent, where consistency may be sought for and applied, and those aspects that are task dependent and necessarily idiosyncratic. In this fashion we attempt to resolve the obvious need for consistency (Nielsen, 1989) with the need to capture the unique features of different tasks (Grudin, 1989), in identifying the essential components of conceptual/cognitive interfaces.

1.2.2 Relationship between the Physical and the Cognitive Interface

The goal of users is to carry out tasks (or else why wouldn't they be at the beach?). We already take for granted the fact that the physical operations and visual displays required in satisfying the goal are likely to be handled through a WIMP interface. However, the detailed physical actions will be carried out in order to achieve task-oriented conceptual goals. Thus the user is not seeking to select cursor locations with the mouse or open and close windows per se, but through these actions he operates on the system's model of the task in such a way as to achieve his goals.

We may view the human computer interface as a hierarchy of layers, each satisfying a different type (or level) of task-related goal. The physical (WIMP) interface provides a lower level of detailed physical activity and manipulation of objects within the user interface metaphor. At the next level the NERD interface (defined in Section 2, below) handles conceptual subgoals within the task. At a higher level still are the overall task strategies, many of which will be idiosyncratic for the current task. Thus we can imagine a hierarchy of interfaces as shown in Figure 1. The base of the hierarchy is the WIMP interface. The next level of the hierarchy is the NERD interface. At the top level of the interface is the TASK interface consisting of idiosyncratic components that represent the unique features of the current task.

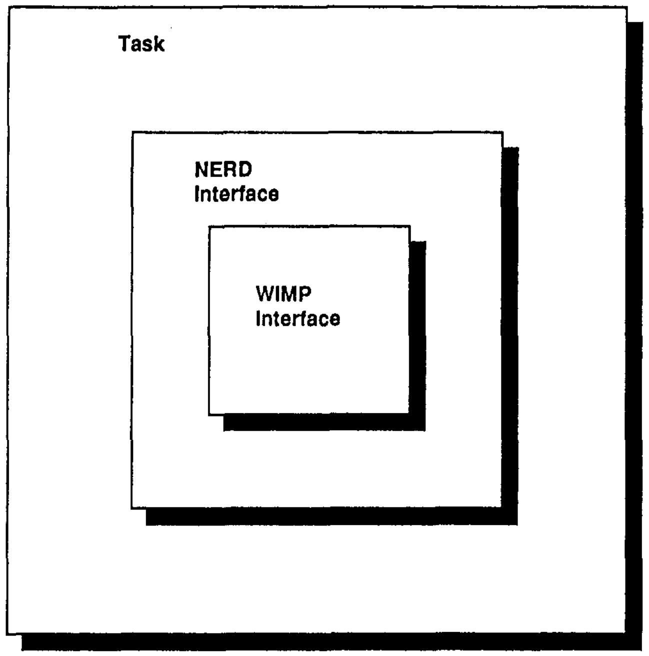

Figure 1 - The Relationship between the WIMP and NERD Interfaces, and the Task.

2. THE NERD INTERFACE

We begin our discussion of the NERD model by noting that it has been strongly motivated by the success of the WIMP model, and that in specifying the NERD approach we seek a similar set of advantages to those provided by the WIMP model. Thus the goals of the NERD approach are to:

- assist in visualization of the task (and the conceptual structure or "feel" of the task model in the software)
- recognize operations instead of having to recall com mands, transferring knowledge about the architecture of the cognitive interface from one application to another
- allow customization of the cognitive interface based on the cognitive model of the user
- produce an apposite acronym that everyone can re member.

The elements making up the NERD interface are:

Navigation
Evaluation
Refinement and
Demonstration.

Navigation refers to the fact that the user has direct access to the structure of the system's view of the task. Evaluation is carried out by the system through explicit information to the user about task performance, and Refinement then operates to correct misperceptions in the user's view of task status. Finally, Demonstration allows the user to modify the cognitive interface so that it matches more closely his own cognitive model. In the following sections we expand on these four elements of the NERD interface.

2.1 Navigation

The first element of the NERD interface is Navigation. Following Norman's (1986) discussion of cognitive compatibility in the user interface, we see a central problem of the user as that of mapping their mental model of the task onto the system's representation. However, the concept of cognitive compatibility is expressed as a measure of interface usability and a design criterion, rather than as a strategy for design per se.

In the NERD model of the cognitive interface, navigation is the mechanism by which cognitive compatibility is achieved. The strategy is to present the structure of the system's model of the task to the user directly, thereby encouraging him to incorporate the system's view of the task into his own model. We admit that our views on the importance of navigation in the conceptual interface are inspired by our experience with hypermedia, but since hypermedia is a microcosm of the universe of HCI issues, we believe that the fundamental issues of navigating a hyperbase are illustrative of the more general problem of human-computer integration.

Navigation is the process by which the user moves through the conceptual structure of the interface (i.e. to navigate is "to work out which direction to go while travelling"; see Waterworth and Chignell, in preparation). This structure may be represented by a set of hierarchically ordered menus, by a network of information, by a set of windows that open and close in different task contexts, and so on. Whatever this structure is, however, the NERD model requires that it be made available to the user, i.e., that navigability is enhanced. Recent work in hypermedia has highlighted the need for navigable interfaces, and we expect that current work on visual metaphors and browse tools for hypermedia will eventually be incorporated into the navigation component of the NERD model.

Figure 2 shows the human-computer interaction processing loop that is assumed by the NERD model. Like most conceptual models in human-computer interaction it might conceivably be mistaken for a map for discovering unmarked burial chambers in King Tut's tomb.

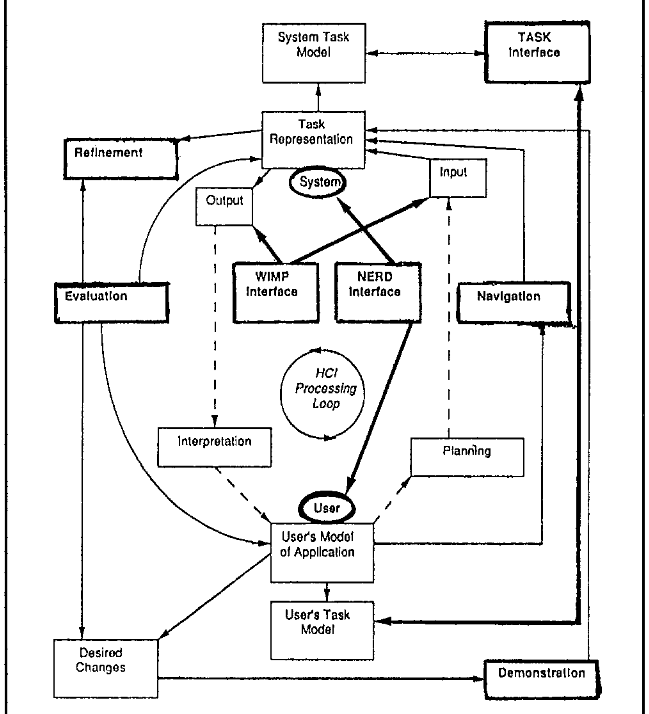

Figure 2 - A Schematic Representation of the WIMP, NERD, and TASK interfaces (in bold), showing the components of the NERD model (bold), in the context of the HCI Processing Loop (dashed lines).

2.2 Evaluation

Navigation involves the representation of the task in the cognitive interface and the user's interaction with that representation. In other words, navigation emphasizes the input from user to system. The second component of the NERD model, Evaluation, deals with the output from the cognitive interface. Broadly speaking, evaluation is concerned with conveying information about task performance to the user (in terms of the model presented in the cognitive interface) and in assessing the user's understanding of updates to the current task status as represented in the interface.

Note here that the evaluative component is the NERD equivalent of an error handling process. While navigation is concerned with allowing the user to operate on a cognitive representation of the task, evaluation is concerned with checking for discrepancies between the system's and the user's understanding of task status. This evaluative component, and the "error signal" that it produces, corresponds somewhat to the notion of semantic distance discussed by Norman (1986).

2.3 Refinement

Continuing with a control-theoretic or cybernetic approach (cf., Powers, 1973), we need a component that reduces the error signal detected through evaluation. We refer to this latter component as Refinement. In refinement, the output of the cognitive interface is modified to correct the user's perception of task status.

We should note here that the notion of refinement may be quite difficult to achieve. It implies an ability on the part of the system to recognize discrepancies in the user's cognitive model. Thus we are defining refinement as a desirable function to be included in the NERD model rather than as a process that can be implemented immediately. Note also that the particular implementation of refinement may depend on the type of task being carried out. This reflects the fact that higher levels of the interface will generally achieve consistency at a more abstract level (e.g., NERD interfaces will include a refinement component, but the precise way in which refinement is carried out will vary from application to application).

2.4 Demonstration

The final component of the NERD model is demonstration. This is the mechanism by which the user is able to customize the cognitive interface that it becomes more consistent with his cognitive model. Like refinement, demonstration is a fairly high level concept that may be achieved in a number of different ways. The basic idea of demonstration is that the user can define an operation from his cognitive model of the task directly onto the cognitive interface. This modification of the cognitive interface through demonstration is somewhat related to the notion of visual programming, of macro definition, and similar capabilities. For instance, some software packages have a type of "watch me" capability where the user can carry out a series of actions and then have them saved as a single operation (e.g., the macro-maker utility that can be used with Microsoft Word on the Macintosh).

Recent work in User Interface Management Systems (e.g Singh et al, 1990) provides demonstrational facilities for the specification of the dialogue aspects of WIMP interfaces. The NERD model also assumes some kind of customization of the cognitive interface through demonstration.

To summarize, the NERD model is concerned with specifying a core set of functions that enhance the compatibility between the cognitive interface of software and the cognitive model of the user. Navigation increases cognitive compatibility by making the structure of the cognitive interface obvious to the user, who may then operate on that structure directly. Demonstration increases cognitive compatibility by allowing the user to customize the behavior of the cognitive interface so that it conforms more closely to his cognitive model. Refinement increases cognitive compatibility through modification of the cognitive interface by the system. Thus demonstration and refinement both attempt to improve cognitive compatibility through modification of the cognitive interface, but demonstration does this on the initiative of the user, while refinement does this on the initiative of the system. Finally evaluation is the process by which the error signal (i.e., lack of cognitive compatibility) is assessed. Evaluation forms the basis for refinement, and may also be used for requesting the user to customize by demonstration, in cases of low cognitive compatibility, where refinement is either unavailable or insufficient.

3. WIMPS, NERDS AND USERS' TASKS

In this section we attempt to consolidate our approach into a more general view of human-computer interaction.

3.1 Are NERDs Metaphorical ?

We have alluded only briefly to the role of metaphor in interface design. It is important to bring out the correspondences and contrasts between the increasingly prevalent tendency to stress an analogy-based view of interface design and our distinction between the WIMP, NERD and TASK perspectives. In our view, a metaphor, or set of metaphors, can serve as the overarching framework bridging the WIMP physical layer, the NERD conceptual layer, and the TASK pragmatic layer.

The WIMP interface is intrinsically metaphorical; at its simplest the analogy is that of direct manipulation of real world objects. This is now so pervasive that many users no longer conceive of direct manipulation as a metaphor. Rather, it seems the very essence of the nature of human-computer interaction. At its simplest, the analogy operates at the lexical or semiotic level. An action, say clicking on an icon, comprises an item in a universe of possible physical events. Similarly, an icon stands as a symbol for one of a limited set of system entities. As the metaphor expands to encompass sequences or combinations of items and events, operations move to a syntactic or gestural level, and inevitably approximate more closely to the pragmatics of task performance. An example of this level of operation is choosing an icon that stands for topic guide, and clicking on the icon to commence a consultation. At the semantic level the user moves even closer to fulfilling tasks. A complete consultation with our topic guide would be an example, and may be seen as combining lexical level WIMP actions with higher level syntactic and semantic operations as components of overall task completion (see also Hammond and Allinson, 1987).

In terms of the levels of metaphor mapping outlined above, the NERD interface operates at the semantic level. But it goes beyond the semantics of defining possible ways of accomplishing tasks via physical operations. In terms of navigation, the interactive process can clearly be seen as exploratory; that is, the user becomes an active participant in discovering possible ways in which his needs may be satisfied. In a sense, this is the first glimpse into a larger set of interactive possibilities realisable by NERDs. Combined with evaluation and refinement, navigation becomes a part of a creative process of meeting users' needs in a flexible way. The Demonstration component most clearly illustrates the transcendental ability of NERDs, as compared to WIMPs, to go beyond the limits of the metaphorical approach to human-computer interaction. This is achieved by moving beyond the display level, and beyond presentation management to sequences or structures of interaction that achieve functions in relation to users' tasks.

3.2 Artificial Reality: The Answer to a WIMP's prayer?

Why are we promoting NERDs when we could be promoting the development of artificial reality? In our view, NERD interfaces and artificial realities lie on separate evolutionary paths. An artificial reality may be thought of as a form of expanded metaphor that provides both a convincing visual model and a set of metaphor-relevant actions. Artificial realities appear to be extensions of WIMP interfaces because they handle the visual/physical interface. They do not handle the problem of interface structure, and they do not provide mechanisms for assisting the user in understanding the structure except through the mechanism of the metaphor. Most importantly, they do not provide mechanisms for evaluating or improving cognitive compatibility.

Artificial realities are an attempt to answer questions such as:

How do we make the feelings of direct engagement and direct manipulation more compelling?

How do we make the conceptual structure of the task transparent to the user by mapping it into a compelling metaphor?

Thus the essential difference between NERD interfaces and artificial realities is that the former seeks to make the conceptual structure of the interface explicit and provide tools for modifying it, while artificial realities seek to make the conceptual structure transparent within the context of a comprehensive metaphor. In our view, NERD interfaces generally offer a better prospect because they are adaptive and do not require the unlikely situation (in many cases) of finding a compelling metaphor which can be incorporated as an artificial reality model without conflicting at any point with the conceptual structure of the task. Rather, the user becomes, through interactions via the NERD interface, the designer of his own interface.

3.3 Non-NERD Aspects of the Cognitive Interface:The TASK

In this section we consider the idiosyncratic features of the task. Most important among these are the task specific metaphors that are used. We should make a clear distinction between these task specific metaphors and a more general metaphor that might eventually be included within the NERD model (e.g., a general spatial model that we are currently considering at the Institute of Systems Science).

No one would deny that the focus of systems design work should be on helping people fulfill their purposes, that is, complete their tasks, in ways that are attractive and efficient and, ideally, that lead to progressive improvements in performance and satisfaction.

It would help of course, if we knew what a task was, or how to specify its structure and components. The problem is that it is easier to point at someone performing a task than it is to say exactly what the task is that is being performed. We are reminded of the story of someone who is walking around their garden digging holes and then filling them in again. To an observer, this hardly seems like task-oriented behaviour until he is told that the task is to look for the location of an underground pipe.

In spite of, or perhaps because of, the difficulty of task analysis, there are quite a number of task analytic methods that have been suggested. Fleishman and Quaintance (1984) provide a broad overview of task analysis methods, while Wilson et al (1988) have reviewed eleven different techniques that have been developed or applied more specifically with HCI in mind.

A relatively common approach to task analysis is to view tasks as consisting of series of goal-directed transactions controlled by programs or operations that proceed through some state or event space. This approach is exemplified in the GPS approach to problem solving (Newell and Simon, 1972) which seems to be consistent with a lot of the more recent work on task analysis.

An alternative to the state-space representation is the grammar approach where generic actions and objects are combined into some knowledge representation grammar. Of course, saying that one will use a grammar is not very helpful in itself, one still has to develop a set of grammatical constructs that will function as a set of task analysis tools. A common trick at this point is to divide the prospective grammar into components and levels, and to name these components and levels in meaningful and task-oriented ways.

One can also attempt to combine the properties of state-space and grammar representations. For instance we might propose an analytic system called GROAN (Goals, Rules, Operators, ANalytic methods). In this hypothetical system one might analyze task related mental processes and activities in terms of task-relevant goals, operators used to achieve goals, analytic methods which are sequences (plans) of operators, and rules for selecting methods (plans). One could then GROAN all the way from the top level goal of the task down to the fundamental units of human activity (hopefully, though not necessarily, getting off the wagon before reaching the psychophysical level).

Thus one could have a hierarchical decomposition of nested GROANs. One could then link nested GROANs into procedural sequences using "production rules". One could even develop various theories about the complexity or usability of different GROAN sequences. We could develop this further, but the question remains, what is an adequate model of the task for the conceptual interface? We expect that this will be goal-oriented and will involve a state-space representation of operator sequences. It could even be implemented in production rules.

4. CONCLUSIONS: WHITHER WIMPS AND NERDS ?

Despite the recent explosion of interest in WIMP interfaces, they have not yet been approached from a viewpoint that allows them to be fully integrated with users' tasks and purposes. The interactions are, literally, and in more ways than one, 2-dimensional. User and system view each other through a window against which the user's nose gets flattened. Only a very few attempts have been made to enter the user's world through multi-dimensional interactions (which would take account of work aims and structures, sequences of actions, meaning and effects in context). The NERD interface bridges these two worlds. NERD interfaces should supplement WIMP interfaces rather than supplanting them, and they should be closely integrated with the task level of the interface.

In conclusion, we claim that NERDSs may need WIMPs, but TASKs need NERDs. NERD interfaces provide the essential bridge between the physical interface and the user's conceptual task. At the TASK level, the conceptual dimension of NERD-based interaction is realised as meaningful output, the accomplishment of users' goals. In the future, we hope to carry these notions to their logical conclusion, through the development of integrated systems combining the 3-levels of performance within an interface that takes account of physical interaction (WIMP), dynamic conceptual operation (NERD), and more specific goal-related elements (TASK). The WIMP is merely the tool by which the NERD can achieve the TASK.

ACKNOWLEDGEMENTS

This paper was written while Mark Chignell was a Visiting Scientist at the Institute of Systems Science, National University of Singapore, in 1990. Thanks to Gurminder Singh for at least one helpful comment, and to the Editor of the *Bulletin* for constructive suggestions. It is dedicated to the WIMP pioneers, and to the developers of GOMS, TAGS, PUMS, GUEPS and the many other approaches to modelling HCI, both acronymal and apocryphal.

REFERENCES

Billingsley, P. (1988). Design of Menus. In M. Helander (Ed.), *Handbook of Human-Computer Interaction.* N.Y.: North-Holland.

Card, S.K., English, W.K., and Burr, B.J. (1978). Evaluation of mouse, rate-controlled isometric joystick, step keys, and task keys for text selection on a CRT. *Ergonomics*, 21, 601-613.

Chignell, M.H. and Hancock, P.A. (1986). Integration of the cognitive and physical aspects of the human-machine interface. *Proceedings of the Human Factors Society*, 1986, 1007-1011, Dayton, Ohio.

Fitts, P.M. and Seeger, C.M (1953). S-R compatibility: spatial characteristics of stimulus and response codes. *Journal of Experimental Psychology*, 46, 199-210.

Fleishman, E.A. and Quaintance, M.K. (1984). *Taxonomies of Human Performance.* San Diego: Academic Press.

Grudin, J. (1989). *The Case Against User Interface Consistency.* MCC Technical Report Number ACA-HI-002-89.

Hammond, N. and Allinson, L. (1987). The Travel Metaphor as Design Principle and Training Aid for Navigating around Complex Systems. In D. Diaper and R. Winder, *People and Computers III.* Cambridge: Cambridge University Press.

Newell, A. and Simon, H.A. (1972). *Human Problem Solving.* Englewood Cliffs, N.J.: Prentice-Hall.

Nielsen, J. (Ed.) (1989). *Coordinating User Interfaces for Consistency.* N.Y.: Academic Press.

Norman, D.A. (1986). Cognitive Engineering. In D.A. Norman and S.W. Draper (Eds.), *User-Centered System Design.* Hillsdale, N.J.: Erlbaum.

Paap, K. and Roske-Hofstrand, J. (1988). Taking panes with windows. In M. Helander (Ed.), *Handbook of Human-Computer Interaction.* N.Y.: North-Holland.

Powers, W.T. (1973). *Behaviour: The Control of Perception.* London: Wildwood House.

Sanders, M.S. and McCormick, E.J. (1987). *Human Factors in Engineering Design, 6th Edition.* N.Y.: McGraw-Hill.

Shneiderman, B. (1987). *Designing the User Interface: Strategies for Effective Human-Computer Interaction.* Reading, Mass.: Addison-Wesley.

Singh, G., Kok, C.H. and Ngan, T.Y. (1990). The DRUID User Interface Management System. *ACM SIGGRAPH Symposium on User Interface Software and Technology, 1990.*

Waterworth, J.A. and Chignell, M.H.. A Model of Information Exploration. In preparation.

Wilson, M.D., Barnard, P.J., Green, T.R.G., and MacLean, A. (1988). Knowledge-based task analysis for human-computer systems. In G.C. Van Der Veer, T.R.G. Green, J-M. Hoc, and D.M. Murray (Eds.), *Working with Computers: Theory versus Outcome.* London: Academic Press.