We can design better user interfaces: A review of human-computer interaction styles

BEN SHNEIDERMAN

Human-Computer Interaction Laboratory, Department of Computer Science, and the Institute for Advanced Computer Studies, University of Maryland, College Park MD 20742, USA

Keywords: User interface; Human-computer interaction; Menu selection; Form fill-in; Command Language; Natural language; Direct manipulation.

The widespread use of computers has opened a new dimension of application for the ergonomic sciences. This review recommends three pillars to support the user interface design process: guidelines documents, User Interface Management Systems, and usability labs for iterative testing. Then it presents five primary interaction styles: menu selection, form fill-in, command language, natural language and direct manipulation. The author encourages greater attention to direct manipulation in which the objects and actions are visible, the actions are invoked by selection or pointing, and the impact is immediately visible and reversible.

1. Introduction

The dramatically increased applications of computers has placed a heavy demand on ergonomics specialists. Physical ergonomists are busy designing keyboards, screens, cabinets, seats, lighting, and various exotic pointing devices (for example, the data glove or foot mouse). Cognitive ergonomists are heavily involved in screen layouts, data or knowledge organization, colour choices, wording of messages, selection of response times, and graphic/iconic design. While it may be too bold to declare a 'golden age of ergonomics' it seems clear that human factors issues and the politics of 'user-friendly' are fashionable topics in professional journals and popular magazines. Advertisements herald the newly improved ease of use of products while cartoonists poke fun at failed attempts.

While it is too early to say that foolproof methods exist for excellent designs, we have come a long way in the past five years in our ability to create and evaluate designs. The designer who merely imitated a familiar pattern and followed his/her intuition can now consult useful guidelines documents and textbooks, be guided by emerging theories, speed development through use of effective software tools, and apply reasonable procedures to usability testing.

In fact, it seems appropriate to identify the 'three pillars' of user interface development:

(1) guidelines documents: each project should develop its own guidelines document to record the rules for designing the user interfaces. In some projects the guidelines document may be a ten page note written by two people in a week, while in other projects it may be a 300 page document written by a dozen
people over two years. While the physical document is a vital record of design decisions, the social process of its creation may be equally valuable. As designers, implementers, trainers, users, managers and others debate the guidelines they become a more effective social community and they foster a communal education that has enduring importance. Successful guidelines documents have multiple levels of strictness: rigid standards that cannot be violated, strict guidelines that require management approval for an exemption, or softer practices that can be violated if the designer finds good justification. Of course, an effective guidelines process must have mechanisms for enforcement, granting of exemptions, and modification. Fortunately, there are increasingly excellent examples of guidelines documents that can provide a sound basis for development in many application areas (Brown 1988, Smith and Mosier 1986).

(2) **User Interface Management Systems:** In the past, user interface developers had to be programmers who would work for months to implement the design. Now there are an increasing number of software aids such as prototyping tools, subroutine packages, and full-scale User Interface Management Systems (UIMS) (Harston 1988). Use of such tools can result in order of magnitude productivity improvements. Many products assist in the design of menus, form fill-in, or command languages but novel graphics oriented or direct manipulation designs must often still be done by the time-consuming programming process.

(3) **Usability laboratories and iterative testing:** hundreds of organizations have developed a usability laboratory to aid developers of user interfaces. These labs are more than a physical place to do testing, they are a source of assistance from the earliest phases of system design. Testing is no longer seen as the last minute validation of a design, but has become the normal process for refining ideas from the beginning. Successful designers recognize the time and resource savings effects of usability testing. In many organizations there is energetic competition for the scarce resources of time in the usability lab and consultation with staff.

Testing can focus on specific users and specific tasks with these possible dependent variables:

1. time for a user to learn to accomplish a task
2. speed of performance on benchmark tasks
3. rate and distribution of errors
4. subjective satisfaction (Norman et al. 1988)
5. user's retention of syntax and semantics over time with intermittent use.

While these three pillars can support the edifice of user interface design, there is still a need for an understanding of foundation principles of the five primary interaction styles.

2. **Taxonomy of interaction styles**

There are many ways of interacting with a computer. The variations result from differences in tasks, computer concepts, and interface devices. The five primary interaction styles described in this paper could each accomplish the full range of interactions, although some would be awkward in certain situations:
### Interaction style comparison

#### Advantages:

**Menu selection**
- shortens training, reduces keystroke structures decision-making,
- permits use of dialog management tools,
- easy to support error handling

**Form fill-in**
- simplifies data entry,
- requires modest training, assistance is convenient, shows context for activity,
- permits use of form management tools

**Command language**
- flexibility, supports user initiative,
- appeals to 'power' users,
- potentially rapid for complex tasks,
- support macro capability

**Natural language**
- relieves burden of learning syntax

**Direct manipulation**
- visually presents task, easy to learn,
- easy to retain, errors can be avoided,
- encourages exploration, high subjective satisfaction

#### Disadvantages:

- danger of many menus,
- may slow frequent users, requires screen space,
- requires rapid display rate

- consumes screen space,
- requires typing skills

- difficult to retain, poor error handling
- requires substantial training and memorization,

- requires clarification dialog,
- may require more keystrokes,
- may not show context, unpredictable

- may require graphics display/pointing devices,
- more programming effort until tools improve,
- may be hard to record history or write macros

The remainder of this section reviews each style and offers guidelines for improving the design. The statements are often extrapolations from experiments that need to be validated and refined.

### 2.1. Menu selection

In menu selection systems the computer displays a list of items from which the user selects. If the items are meaningful to the user then menu selection can be a rapid, accurate, and satisfactory approach. If the items are hard to understand or appear similar to each other, users can become confused and make errors.

Menu selection is advantageous because it decomposes a complex interaction into a series of smaller steps and because it provides structure for decision-making. On the other hand this same decomposition process can be too rigid for some users and it may slow down the frequent knowledgeable user. For many situations menu selection can substantially reduce the number of keystrokes necessary and thereby reduce error rates and performance times. Menu selection mechanisms can be the familiar numbered menus, lettered menus, mnemonic lettered menus, or unlabelled menus. Unlabelled menus operate by movement of a highlight bar over the items, under the control of arrow keys, a mouse, a joystick, a graphics tablet, etc. Smith and Mosier (1986) offer extensive guidelines for menu selection systems and Shneiderman (1987) reviews empirical studies and suggests practical design rules:

**Menu selection guidelines**

- Use task semantics to organize menus
  
  (single, linear sequence, tree structure, acyclic and cyclic networks)
— Prefer broad/shallow to narrow/deep tree
— Show position by graphics, numbers, or titles
— Items become titles for trees
— Meaningful groupings of items
— Meaningful sequencing of item
— Brief items, begin with keyword
— Consistent grammar, layout, terminology
— Type-ahead, jump-ahead, or other short-cuts
— Jumps to previous and main menu
— Consider: online help, novel selection mechanisms, response time, display rate, screen size

Many new ideas and clever variations of menus are being developed: pull-down or pop-up menus with mouse selection, circular-shaped pie menus to minimize cursor motion, iconic menus to speed recognition, dynamic menus whose contents vary, and audio menus for speech systems. Researchers are developing cognitive models of menu usage, comparing deep to broad menus, studying menu use under stress, and studying new pointing devices.

2.2. Form fill-in
In a form fill-in interaction the users' main task is to provide data in labelled fields clustered on one or more screens. Sometimes the data items are merely binary choices (Yes/No or Male/Female) or selections from short lists (days of the week or a set of colours), but they can also be taken from large domains (personal names or chemical formulae) or may be essentially unbounded (explanatory paragraphs or meteorological data). Of course, it is possible to replace form fill-in with a series of menu choices, but this strategy can become extremely tedious.

Keyboards or other input devices are effective ways of inputting lengthy data fields. The keyboard may be seen as a continuously displayed menu that permits rapid selection using a well-learned skill. Increasingly data entry is handled more automatically as is done in reading magnetic stripes on charge cards to get names and account numbers or grocery packages with bar codes. Form fill-in requires that users learn to use the keyboard with tab, cursor control, and backspace keys. If the users are competent keyboard operators and field labels are meaningful then form fill-in is an effective approach that produces a quite rapid rate of data entry, a moderate level of errors, and good user satisfaction.

There has been very little research in the design of form fill-in systems but there are several sources of design guidelines (Galitz 1985, Smith and Mosier 1987):

Form fill-in design guidelines
— Meaningful title
— Comprehensible instructions
— Logical grouping and sequencing of fields
— Visually appealing layout of the form
— Familiar field labels
— Consistent terminology and abbreviations
— Visible space and boundaries for data entry fields
— Convenient cursor movement
— Error correction for individual characters and entire fields
— Error messages for unacceptable values
Better user interfaces

— Optional fields should be marked
— Explanatory messages for fields

Research issues include very basic questions of appropriate layout, justification, spacing, grouping, sequencing, handling of optional fields, and online help. A combination of form fill-in with menu selection might help the novice user. For example, the list of departure cities might be made available when the cursor is moved to that field; then the user could select a city instead of typing it and it would be displayed in the field. A useful conjecture, that needs validation, is that as user experience increases, the density of fields on a single screen can be increased while the field labels can be shortened.

2.3. Command language

Knowledgeable frequent users do not want to be distracted by having to locate an item in a list, nor do they want to have to view and move a cursor over a form. They can manipulate the possibilities in their mind and want concise notations for issuing commands with modest informative feedback. These ‘power’ users want to be able to put several commands on a single line and even create new commands that encapsulate the work of several frequently used command sequences. The learning time may be days or weeks, but since usage is frequent the benefits of concision are great. Often command languages emerge from familiar notations such as mathematics, boolean logic, or music, but many command languages have been created to deal with novel task domains such as text and string manipulation, hotel reservations, or information retrieval. There has been much research in command language design for text editors and other applications. Consistency in structure, meaningfulness, orderly abbreviations, small number of commands, and congruent pairs of commands have been shown to be important determinants of rapid learning, rapid use, low error rates, high satisfaction, and easy retention over time:

Command language guidelines
— Create explicit model of objects and actions
— Choose meaningful, specific, distinctive names
— Try for hierarchical structure
— Provide consistent structure (hierarchy, argument order, action-object)
— Support consistent abbreviation rules (prefer truncation to one letter)
— Offer frequent users the capability to create macros
— Consider command menus on high-speed displays
— Limit number of commands and ways of accomplishing a task

Command language research focuses on the choice of command names, sequencing of arguments, usage of abbreviations, cognitive models of objects and actions, command completion strategies, error correction, and consistency rules.

2.4. Natural language

Many computer scientists and computer users propose that the ‘ultimately desirable’ way of using a computer is through natural language interaction. They argue that people already know their own natural language and therefore learning to use the computer would be simplified if the computer accepted natural language input. This vision has propelled many researchers over the years as they tried to make natural language interaction systems for programming, database retrieval, or expert systems
usage. There are commercial products such as INTELLECT (Artificial Intelligence Corporation) or Q & A (Symantec) but the success could only be considered to be modest.

Relieving the user of the burdens of syntax is only a small part of the problem in using a computer and is of benefit only to novice and intermittent users. The much more complex part is understanding the computer concepts and the task domain. Relief from syntactic details does not ensure that the users will know that files must be saved before quitting or that stock market purchases must be in blocks of 100 shares.

Furthermore natural language interaction systems constantly confront the user with the problem of uncertainty about whether a particular input will be acceptable to the machine. This inhibits planning ahead since users must always be ready to engage in clarification dialogue. Improved designs will mitigate this problem and speech input might reduce the penalty of typing, but the users will still be struggling to discover the syntactic and semantic scope of the system. Therefore, this author is sceptical that natural language interaction will become widely used. The reader should be cautioned that there are still many people who hold a more optimistic view. Natural language generation and natural language parsing of textual databases are likely to be more successful applications of the technology of computational linguistics.

Finally, the special case of interactive fiction and adventure games should be cited as a highly successful application of natural language interaction. The success stems from the fact that in this situation part of the challenge and fun is to discover the syntactic and semantic bounds of the program.

Natural language interaction guidelines
— Reducing syntactic load is not enough
— Computer and task semantics should be represented
  predicate calculus, boolean algebra, set theory,
  normalization theory, database entities and values,
  permissible operations and constraints
— Make syntactic and semantic scopes visible
— Therefore, NLI might work best for:
  • Users who are knowledgeable about task domain
  • Intermittent users who cannot retain syntax
  • Users with moderate computer skill
  • Good typists (until speech I/O improves)

Researchers continue to develop natural language parsers and refine attempts at natural language interaction. However, there is a great need for empirical studies to identify the successful aspects and to guide refinement.

2.5. Direct manipulation
The first four interaction styles emerged from a 'teletype mentality' or at best a text-oriented view of the task. However, many designers are breaking away from the constraints of teletype and keyboard designs and employing screen-oriented, form-oriented, visual and graphic approaches with pointing devices such as arrow keys, a mouse, touchscreens, or a graphics tablet. These new designs often depend on bit-mapped high resolution displays, but the concepts of direct manipulation can be applied with simple text screens and a simple pointing device. The key to direct
manipulation designs is to create a visual representation of the 'world of action' that includes selectable displays of the objects and actions of interest. Then with pointing, zooming, and panning the user can rapidly perform operations, see the results immediately and reverse operations if necessary. These ideas have been applied in the Xerox STAR, Apple Macintosh, videogames, many word processors, and other applications (Shneiderman 1983, 1987). The benefits include relatively rapid learning with high retention over time and high user satisfaction. Errors can often be prevented because the representation shows the users the impossibility of performing a task (the Pacman can’t go through walls) and because typographic errors are eliminated when the user selects from a set of displayed objects. Exploration is often encouraged in direct manipulation environments, especially when reversibility of action is ensured.

There are disadvantages in current direct manipulation designs. It is often difficult to create programs and recording the history of a session can be troublesome. Developing direct manipulation systems is difficult with current software tools, but some improvements are expected. There is some concern also for visually impaired users. For some tasks, frequent users may prefer command language approaches (unseen manipulation) because they prefer not to be distracted by having to locate objects and actions on the screen and then select them.

The guidelines for direct manipulation with some of the benefits and concerns might be stated as:

**Direct manipulation guidelines**
- Create visual representation of the 'world of action'
  - Objects and Actions are shown
  - Tap analogical reasoning
- Rapid, incremental, and reversible actions
- Replace typing with pointing/selecting
- Make results of actions immediately visible

<table>
<thead>
<tr>
<th>Benefits</th>
<th>Concerns</th>
</tr>
</thead>
<tbody>
<tr>
<td>Control/display compatibility</td>
<td>Increased system resources, possibly</td>
</tr>
<tr>
<td>Less syntax reduces error rates</td>
<td>Some actions may be cumbersome</td>
</tr>
<tr>
<td>Faster learning and higher retention</td>
<td>Macro techniques are often weak</td>
</tr>
<tr>
<td>Encourages exploration</td>
<td>History/tracing may be difficult</td>
</tr>
<tr>
<td></td>
<td>Visually impaired have more difficulty</td>
</tr>
</tbody>
</table>

This researcher sees great opportunities in applying direct manipulation and in studying its benefits. There are many domains in which direct manipulation is still under-used. While videocassette player designers have finally caught on to the idea of 'on-screen programming' they can do much better as can designers of other home electronic equipment, commercial applications, programming languages and environment, medical equipment, etc. Still there are many problems that invite research: how to do programming by direct manipulation, how to store session histories, how to measure and predict performance, and when is direct manipulation slower than command typing or menu selection.

3. **Choosing an interaction style**

Like good cooking, good user interface design is a blend of measurable aspects (calorie counts, fat percentages, vitamin levels) with subjective issues (smell, texture,
colour, taste) and stylistic variations (Szechuan, French, Northern Italian) plus contemporary fashion (nouvelle cuisine, par-boiled vegetables, rare meat). Designers absorb each problem situation thoroughly, then apply reason and intuition combined with experience to produce a solution. Some designers favour menu selection or command language for all situations because they are familiar with the strategies for design and implementation. In other cases the availability of form management software may sway designers to depend heavily on this approach. As a design field, human-computer interaction is moving beyond depending only on the intuition of designers; empirical evidence is accumulating to guide designers. Still interface design is a new field and much work remains to be done before reliable designs for multiple situations are available.

3.1. Factors influencing choices
If the task has a high degree of data entry, then form-fill is strongly recommended, although command language can be used. If a familiar notation is already available to the users, then that notation should be converted into a compact and hopefully easy to learn command language. If exploration is important, as in circuit or automobile design, then direct manipulation can provide a flexible environment with minimum distraction from the task.

User skill level has a profound influence on the selection of an interaction style. With novices, menu selection or some form of direct manipulation present familiar words or visual representations of the task domain. These styles require limited keyboard skills and computer knowledge. More knowledgeable users can work more rapidly with a form fill-in approach. Knowledgeable intermittent users can work with any style. As frequency of use and proficiency increases, users will seek more rapid and productive styles with less to look at on the screen. Expert users expect short response time and macro facilities to encapsulate frequent sequences of actions into a single command.

In addition to the interaction style, there are a host of variables that have to be adjusted as a function of the users' knowledge and frequency of use. For example, novices need meaningful labels, while knowledgeable intermittent users prefer abbreviations so that more information can be shown in a single display. Expert frequent users may prefer to eliminate some labels and use concise codes, thereby speeding up screen scanning and permitting still denser displays. Similarly, the desire for informative feedback when actions are taken generally decreases with increased expertise and frequency of use. These rules describe some of the design features that vary with the users' profile. There are certainly other features and exceptions to these rules.

3.2. Blending interaction styles
The five primary interaction styles are meant for exposition and education. In real systems, designers blend styles where appropriate. For example, in a menu selection system, the date may still be requested as a fill-in item, because menu selection would be preposterous. Similarly, personal names are usually handled as a form fill-in item. For a short list, say two to 12 items, if the display rate is reasonably fast, a menu selection strategy is often the best. As the list gets longer the designer must weigh the users' familiarity with the items, the potential and seriousness of an error, the speed of interaction required, and the length of the items. All but the final criteria are difficult to assess quantitatively, leaving the designer with the responsibility for making a tough judgment. A few versions might be built and subjected to empirical testing.
Sometimes a blend of interaction styles can help resolve the problem. For example, an airline reservations system user may be presented with a form fill-in for the departure and arrival cities. Those who wish to can type in the city name or the short three letter airport code if they know it. Alternatively, the placement of the cursor in the field may generate a pop-up window with a menu of cities that can be selected by pointing (mouse or arrow keys). Then the selected city name is automatically displayed in the form. Form fill-in can be nicely supplemented by pop-up menus for less knowledgeable users.

Similarly, command language systems can be supplemented by a form fill-in or menu selection strategy for less knowledgeable users. For example, many text editors (e.g. Wordstar or Finalword) start with a command language approach that has a hierarchical structure. After pressing the first key for the first part of the command, the user can immediately type more characters to complete the command, but if they are less familiar with the commands a menu will appear automatically after 4-6 seconds. Some users prefer to make the menus visible all the time, others prevent them from ever appearing.

A quite similar approach, called command menus, is nicely implemented in LOTUS 1-2-3. The top line of the screen shows a menu with items that are selected by arrow keys or typing the first letter. Then another one-line menu appears, or arguments are typed in, going down as many as five levels in the tree structure. Novice users walk slowly through the menus while experts type ahead rapidly enough that the menus appear on the screen for only a fraction of a second. Another blending approach is to offer a form fill-in and allow the knowledgeable user to type a long sequence of commands in the field. Direct manipulation systems often revert to menu selection approaches when a visual representation cannot be found. Some direct manipulation systems offer a menu of visual items, such as a colour or texture palette.

Tennant et al. (1983) offer an imaginative blend of natural language and menu selection. The users select from a menu of syntactically and semantically acceptable sentence fragments. As each choice is made, the permissible sentence fragments change. With a rapid display and pointing device, the user can quickly assemble a syntactically and semantically correct query for a database. These are but a few possibilities. Identifying useful and appropriate blends for different users in different situations is an important goal for researchers and system designers.

4. Conclusions
Designers of interactive systems have had to work from their own experience and intuition, validating their designs by costly and time consuming iterative testing. There is some hope that we can more precisely understand when to apply a particular interaction style and how to refine it so that it produces rapid user performance, low error rates, high satisfaction, ease in learning, and ease in retention over time. Blends of interaction styles can be very effective in serving a range of users and in dealing with certain interaction tasks. A massive programme of empirical research will be necessary to guide this research, but the payoff in more reliable and effective designs can be enormous. Finally, a useful psychological theory, tied to the interaction styles would be a tremendous benefit.

Acknowledgments
Some of the material for this article was prepared for a lecture at Loughborough University that will be published in Brian Shackel (ed). Human Factors for Informatics Usability (1988). Jean Gasen and John Kohl provided useful comments on the draft version.
References and bibliography

General


Menu selection


PERLMAN, G., 1984, Making the right choices with the menus, *INTERACT '84, First IFIP International Conference on Human-Computer Interaction* (North-Holland, Amsterdam), 291–295.


Form fill-in


Better user interfaces


Command language


Natural language


Better user interfaces

Direct manipulation


Le recours massif aux ordinateurs a ouvert une voie nouvelle aux sciences ergonomiques. Cette revue de la question recommande trois points d'appui pour la conception de l'interface utilisateur-ordinateur: les documents de mode d'emploi, les commandes du système et l'évaluation répétitive à partir d'essais. Elle présente également cinq styles interactifs de base: la sélection du menu, le dialogue, le langage des commandes, le langage naturel et la manipulation directe. L'auteur recommande une plus grande prise en compte de la manipulation directe, car elle rend visible les objets et les actions. Les actions sont évoquées par la sélection ou le pointage et leur impact est immédiatement visible et réversible.


計算機の広範囲な使用は人間工学にとって新しい次元の用途を開いた。本論文はユーザー・インタフェース設計過程を支える3本の柱、すなわち、ガイドライン、ユーザ・インタフェース管理システム、反復試験用使用方法ラボ等を推奨する。次に5つの主要対話方式、すなわち、メニュー選択、書式記入、コマンド言語、自然言語、直接操作を述べる。目的と動作が見え、動作の選択またはポインタングによって引き起こされ、影響が直ちに見え、逆にできる直接操作にもっと関心を払うことを勧める。