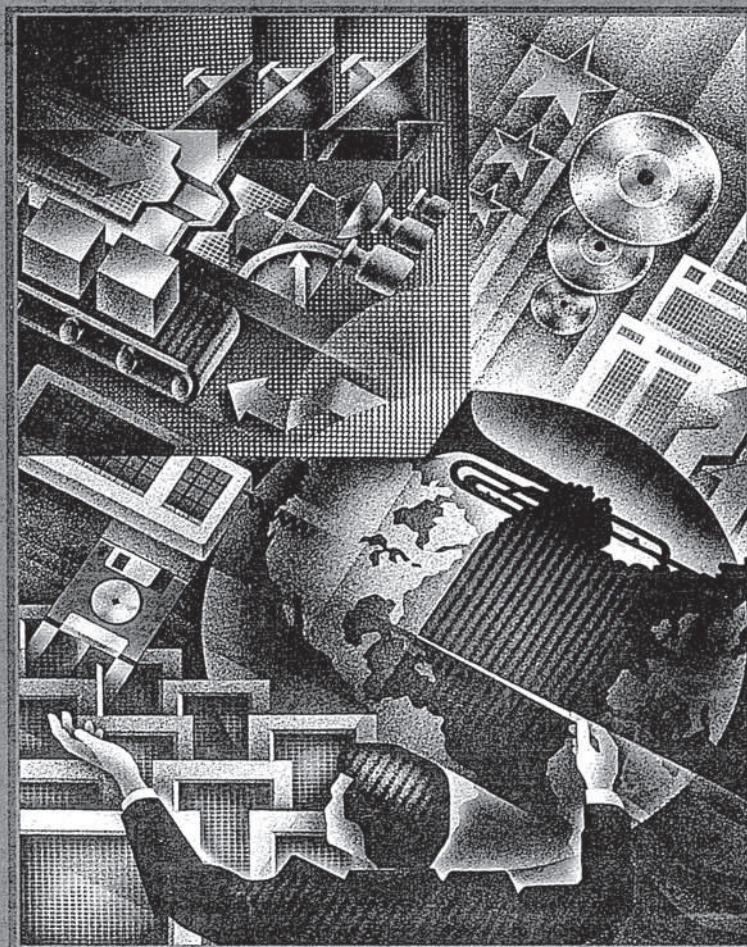


Designing the User Interface

*Strategies for Effective
Human-Computer
Interaction*



Third Edition

Ben Shneiderman

Designing the User Interface

*Strategies for Effective
Human-Computer Interaction*
Third Edition

Ben Shneiderman
The University of Maryland

 **ADDISON-WESLEY**

An imprint of Addison Wesley Longman, Inc.

Reading, Massachusetts • Harlow, England • Menlo Park, California
Berkeley, California • Don Mills, Ontario • Sydney • Bonn • Amsterdam
Tokyo • Mexico City

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and Addison-Wesley was aware of a trademark claim, the designations have been printed in initial caps or all caps.

Library of Congress Cataloging-in-Publication Data

Shneiderman, Ben.

Designing the user interface : strategies for effective human
-computer-interaction / Ben Shneiderman. -- 3rd ed.

p. cm.

Includes bibliographical references and index.

ISBN 0-201-69497-2

1. Human-computer interaction. 2. User interfaces (Computer
systems) I. Title.

QA76.9.H85S54 1998

004'.01'9--dc21

96-37974

CIP

Access the latest information about Addison-Wesley titles from our World Wide
Web site: <http://www.awl.com/cseng>

Chapter opener illustrations from art provided by Mark Kostabi. Reproduced
with permission.

Cover art © Boris Lyubner/SIS

Copyright © 1998 by Addison Wesley Longman, Inc.

Reprinted with corrections, March 1998.

All rights reserved. No part of this publication may be reproduced, stored in a
retrieval system, or transmitted, in any form or by any means, electronic,
mechanical, photocopying, recording, or otherwise, without the prior written
permission of the publisher. Printed in the United States of America.

6 7 8 9 10-MA-01009998

developers, in spite of limited success thus far. Natural-language interaction usually provides little context for issuing the next command, frequently requires *clarification dialog*, and may be slower and more cumbersome than the alternatives. Still, where users are knowledgeable about a task domain whose scope is limited and where intermittent use inhibits command-language training, there exist opportunities for natural-language interfaces (discussed at the end of Chapter 8).

Blending several interaction styles may be appropriate when the required tasks and users are diverse. Commands can lead the user to a form fillin where data entry is required, or menus can be used to control a direct-manipulation environment when a suitable visualization of actions cannot be found.

2.5 Principle 2: Use the Eight Golden Rules of Interface Design

Later chapters cover constructive guidance for design of direct manipulation, menu selection, command languages, and so on. This section presents underlying principles of design that are applicable in most interactive systems. These underlying principles of interface design, derived heuristically from experience, should be validated and refined.

1. *Strive for consistency.* This rule is the most frequently violated one, but following it can be tricky because there are many forms of consistency. Consistent sequences of actions should be required in similar situations; identical terminology should be used in prompts, menus, and help screens; and consistent color, layout, capitalization, fonts, and so on should be employed throughout. Exceptions, such as no echoing of passwords or confirmation of the delete command, should be comprehensible and limited in number.
2. *Enable frequent users to use shortcuts.* As the frequency of use increases, so do the user's desires to reduce the number of interactions and to increase the pace of interaction. Abbreviations, special keys, hidden commands, and macro facilities are appreciated by frequent knowledgeable users. Short response times and fast display rates are other attractions for frequent users.
3. *Offer informative feedback.* For every user action, there should be system feedback. For frequent and minor actions, the response can be modest, whereas for infrequent and major actions, the response should be more substantial. Visual presentation of the objects of interest provides a con-

venient environment for showing changes explicitly (see discussion of direct manipulation in Chapter 6).

4. *Design dialogs to yield closure.* Sequences of actions should be organized into groups with a beginning, middle, and end. The informative feedback at the completion of a group of actions gives operators the satisfaction of accomplishment, a sense of relief, the signal to drop contingency plans and options from their minds, and an indication that the way is clear to prepare for the next group of actions.
5. *Offer error prevention and simple error handling.* As much as possible, design the system such that users cannot make a serious error; for example, prefer menu selection to form fillin and do not allow alphabetic characters in numeric entry fields. If users make an error, the system should detect the error and offer simple, constructive, and specific instructions for recovery. For example, users should not have to retype an entire command, but rather should need to repair only the faulty part. Erroneous actions should leave the system state unchanged, or the system should give instructions about restoring the state.
6. *Permit easy reversal of actions.* As much as possible, actions should be reversible. This feature relieves anxiety, since the user knows that errors can be undone, thus encouraging exploration of unfamiliar options. The units of reversibility may be a single action, a data-entry task, or a complete group of actions such as entry of a name and address block.
7. *Support internal locus of control.* Experienced operators strongly desire the sense that they are in charge of the system and that the system responds to their actions. Surprising system actions, tedious sequences of data entries, inability or difficulty in obtaining necessary information, and inability to produce the action desired all build anxiety and dissatisfaction. Gaines (1981) captured part of this principle with his rule *avoid acausality* and his encouragement to make users the *initiators* of actions rather than the *responders* to actions.
8. *Reduce short-term memory load.* The limitation of human information processing in short-term memory (the rule of thumb is that humans can remember “seven-plus or minus-two chunks” of information) requires that displays be kept simple, multiple page displays be consolidated, window-motion frequency be reduced, and sufficient training time be allotted for codes, mnemonics, and sequences of actions. Where appropriate, online access to command-syntax forms, abbreviations, codes, and other information should be provided.

These underlying principles must be interpreted, refined, and extended for each environment. The principles presented in the ensuing sections

focus on increasing the productivity of users by providing simplified data-entry procedures, comprehensible displays, and rapid informative feedback that increase feelings of competence, mastery, and control over the system.

2.6 Principle 3: Prevent Errors

There is no medicine against death, and against error no rule has been found.

Sigmund Freud (Inscription he wrote on his portrait)

Users of word processors, spreadsheets, database-query facilities, air-traffic-control systems, and other interactive systems make mistakes far more frequently than might be expected. Card et al. (1980) reported that experienced professional users of text editors and operating systems made mistakes or used inefficient strategies in 31 percent of the tasks assigned to them. Brown and Gould (1987) found that even experienced authors made errors in almost half their spreadsheets. Other studies reveal the magnitude of the problem of—and the loss of productivity due to—user errors.

One way to reduce the loss in productivity due to errors is to improve the error messages provided by the computer system. Shneiderman (1982) reported on five experiments in which changes to error messages led to improved success at repairing the errors, lower error rates, and increased subjective satisfaction. Superior error messages were more specific, positive in tone, and constructive (telling the user what to do, rather than merely reporting the problem). Rather than using vague and hostile messages, such as SYNTAX ERROR or ILLEGAL DATA, designers were encouraged to use informative messages, such as UNMATCHED LEFT PARENTHESIS or MENU CHOICES ARE IN THE RANGE OF 1 TO 6.

Improved error messages, however, are only helpful medicine. A more effective approach is to prevent the errors from occurring. This goal is more attainable than it may seem in many systems.

The first step is to understand the nature of errors. One perspective is that people make mistakes or “slips” (Norman, 1983) that designers help them to avoid by organizing screens and menus functionally, designing commands or menu choices to be distinctive, and making it difficult for users to take irreversible actions. Norman offers other guidelines, such as do not have modes, do offer feedback about the state of the system, and do design for consistency of commands. Norman’s analysis provides practical examples and a useful theory.

Three techniques can reduce errors by ensuring complete and correct actions: correct matching pairs, complete sequences, and correct commands.

2.6.1 Correct matching pairs

A common problem is the lack of correct matching pairs. It has many manifestations and several simple prevention strategies. An example is the failure to provide the right parenthesis to close an open left parenthesis. If a bibliographic-search system allowed Boolean expressions such as `COMPUTERS AND (PSYCHOLOGY OR SOCIOLOGY)` and the user failed to provide the right parenthesis at the end, the system would produce a `SYNTAX ERROR` message or, more helpfully, a more meaningful message, such as `UNMATCHED LEFT PARENTHESES`.

Similarly, other marker pairs are required to delimit boldface, italic, or underscored text in word processors or web programming. If the text file contains `<B>This is boldface</B>`, then the three words between the markers appear in boldface. If the rightmost `</B>` is missing, additional text may be inadvertently made bold.

In each of these cases, a matching pair of markers is necessary for operation to be complete and correct. The omission of the closing marker can be prevented by use of an editor, preferably screen oriented, that puts both the beginning and ending components of the pair on the screen in one action. For example, typing a left parenthesis generates a left and right parenthesis and puts the cursor in between to allow creation of the contents. An attempt to delete one of the parentheses will cause the matching parenthesis (and possibly the contents as well) to be deleted. Thus, the text can never be in a syntactically incorrect form. Some people find this rigid approach to be too restrictive. For them a milder form of protection may be appropriate. For example, when the user types a left parenthesis, the screen displays in the lower-left corner a message indicating the need for a right parenthesis until that character is typed.

2.6.2 Complete sequences

Sometimes, an action requires several steps or commands to reach completion. Since people may forget to complete every step of an action, designers attempt to offer a sequence of steps as a single action. In an automobile, the driver does not have to set two switches to signal a left turn. A single switch causes both (front and rear) turn-signal lights on the left side of the car to flash. When a pilot throws a switch to lower the landing gear, hundreds of steps and checks are invoked automatically.

This same concept can be applied to interactive uses of computers. For example, the sequence of dialing up, setting communication parameters, logging on,

and loading files is frequently executed by many users. Fortunately, most communications-software packages enable users to specify these processes once and then to execute them by simply selecting the appropriate name.

Users of a word processor should be able to indicate that section titles are to be centered, set in uppercase letters, and underlined, without having to issue a series of commands each time they enter a section title. Then, if the user wants to change the title style—for example, to eliminate underlining—a single command will guarantee that all section titles are revised consistently.

As a final example, air-traffic controllers may formulate plans to change the altitude of a plane from 14,000 feet to 18,000 feet in two increments; after raising the plane to 16,000 feet, however, the controller may get distracted and may thus fail to complete the action. The controller should be able to record the plan and then have the computer prompt for completion.

The notion of complete sequences of actions may be difficult to implement because users may need to issue atomic actions as well as complete sequences. In this case, users should be allowed to define sequences of their own; the macro or subroutine concept should be available at every level of usage.

Designers can gather information about potential complete sequences by studying sequences of commands that people actually issue, and the patterns of errors that people actually make.

2.6.3 Correct commands

Industrial designers recognize that successful products must be safe and must prevent the user from making dangerously incorrect use of the product. Airplane engines cannot be put into reverse until the landing gear has touched down, and cars cannot be put into reverse while traveling forward at faster than five miles per hour. Many simpler cameras prevent double exposures (even though the photographer may want to expose a frame twice), and appliances have interlocks to prevent tampering while the power is on (even though expert users occasionally need to perform diagnoses).

The same principles can be applied to interactive systems. Consider these typical errors made by the users of command languages: They invoke commands that are not available, request files that do not exist, or enter data values that are not acceptable. These errors are often caused by annoying typographic errors, such as using an incorrect command abbreviation; pressing a pair of keys, rather than a desired single key; misspelling a file name; or making a minor error such as omitting, inserting, or transposing characters. Error messages range from the annoyingly brief ?

or WHAT?, to the vague UNRECOGNIZED COMMAND or SYNTAX ERROR, to the condemning BAD FILE NAME or ILLEGAL COMMAND. The brief ? is suitable for expert users who have made a trivial error and can recognize it when they see the command line on the screen. But if an expert has ventured to use a new command and has misunderstood its operation, then the brief message is not helpful. They must interrupt their planning to deal with correcting the problem—and with their frustration in not getting what they wanted.

Some systems offer automatic command completion that allows users to type just a few letters of a meaningful command. They may request the computer to complete the command by pressing the space bar, or the computer may complete it as soon as the input is sufficient to distinguish the command from others. Automatic command completion can save keystrokes and is appreciated by many users, but it can also be disruptive because the user must consider how many characters to type for each command, and must verify that the computer has made the completion that was intended.

A more effective preventative for errors is to apply direct-manipulation strategies that emphasize selection over command-language typing. The computer presents permissible commands, menu choices, or file names on the screen, and users select their choice with a pointing device. This approach is effective if the screen has ample space, the display rate is rapid, and the pointing device is fast and accurate.

2.7 Guidelines for Data Display

The separation between basic principles and more informal guidelines is not a sharp line. However, thoughtful designers can distinguish between psychological principles (Wickens, 1993; Bridger, 1995) and practical guidelines that are gained from experience with a specific application. Guidelines for display of data are being developed by many organizations. A guidelines document can help by promoting consistency among multiple designers, recording practical experience, incorporating the results of empirical studies, and offering useful rules of thumb (see Chapters 3 and 11). The creation of a guidelines document engages the design community in a lively discussion of input or output formats, command sequences, terminology, and hardware devices (Brown, 1988; Galitz, 1993). Inspirations for design guidelines can also be taken from graphics designers (Tufte, 1983, 1990, 1997; Mullet and Sano, 1995).

2.7.1 Organizing the display

Smith and Mosier (1986) offer five high-level objectives for data display that remain vital:

1. *Consistency of data display* During the design process, the terminology, abbreviations, formats, colors, capitalization, and so on should all be standardized and controlled by use of a written (or computer-managed) dictionary of these items.
2. *Efficient information assimilation by the user* The format should be familiar to the operator and should be related to the tasks required to be performed with these data. This objective is served by rules for neat columns of data, left justification for alphanumeric data, right justification of integers, lining up of decimal points, proper spacing, use of comprehensible labels, and appropriate measurement units and numbers of decimal digits.
3. *Minimal memory load on user* Users should not be required to remember information from one screen for use on another screen. Tasks should be arranged such that completion occurs with few actions, minimizing the chance of forgetting to perform a step. Labels and common formats should be provided for novice or intermittent users.
4. *Compatibility of data display with data entry* The format of displayed information should be linked clearly to the format of the data entry. Where possible and appropriate, the output fields should also act as editable input fields.
5. *Flexibility for user control of data display* Users should be able to get the information from the display in the form most convenient for the task on which they are working. For example, the order of columns and sorting of rows should be easily changeable by users.

This compact set of high-level objectives is a useful starting point, but each project needs to expand these into application-specific and hardware-dependent standards and practices. For example, these generic guidelines emerge from a report on design of control rooms for electric-power utilities (Lockheed, 1981):

- Be consistent in labeling and graphic conventions.
- Standardize abbreviations.
- Use consistent format in all displays (headers, footers, paging, menus, and so on).
- Present a page number on each display page, and allow actions to call up a page via entry of a page number.
- Present data only if they assist the operator.

- Present information graphically where appropriate by using widths of lines, positions of markers on scales, and other techniques that relieve the need to read and interpret alphanumeric data.
- Present digital values only when knowledge of numerical value is necessary and useful.
- Use high-resolution monitors and maintain them to provide maximum display quality.
- Design a display in monochromatic form using spacing and arrangement for organization and then judiciously add color where it will aid the operator.
- Involve users in the development of new displays and procedures.

Chapter 11 further discusses data-display issues.

2.7.2 Getting the user's attention

Since substantial information may be presented to users for the normal performance of their work, exceptional conditions or time-dependent information must be presented so as to attract attention (Wickens, 1992). Multiple techniques exist for getting attention:

- *Intensity* Use two levels only, with limited use of high intensity to draw attention.
- *Marking* Underline, enclose in a box, point to with an arrow, or use an indicator such as an asterisk, bullet, dash, plus, or X.
- *Size* Use up to four sizes, with larger sizes attracting more attention.
- *Choice of fonts* Use up to three fonts.
- *Inverse video* Use inverse coloring.
- *Blinking* Use blinking displays (2 to 4 hertz) with great care and in limited areas.
- *Color* Use up to four standard colors, with additional colors reserved for occasional use.
- *Color blinking* Use changes in color (blinking from one color to another) with great care and in limited areas.
- *Audio* Use soft tones for regular positive feedback and harsh sounds for rare emergency conditions.

A few words of caution are necessary. There is a danger in creating cluttered displays by overusing these techniques. Novices need simple, logically organized, and well-labeled displays that guide their actions. Expert users do not

need extensive labels on fields; subtle highlighting or positional presentation is sufficient. Display formats must be tested with users for comprehensibility.

Similarly highlighted items will be perceived as being related. Color coding is especially powerful in linking related items, but this use makes it more difficult to cluster items across color codes. User control over highlighting—for example, allowing the operator in an air-traffic-control environment to assign orange to images of aircraft above 18,000 feet—may provide a useful resolution to concerns about personal preferences. Highlighting can be accomplished by increased intensity, blinking, or other methods.

Audio tones can provide informative feedback about progress, such as the clicks in keyboards or ringing sounds in telephones. Alarms for emergency conditions do alert users rapidly, but a mechanism to suppress alarms must be provided. If several types of alarms are used, testing is necessary to ensure that users can distinguish among alarm levels. Prerecorded or synthesized voice messages are an intriguing alternative, but since they may interfere with communications among operators, they should be used cautiously.

2.8 Guidelines for Data Entry

Data-entry tasks can occupy a substantial fraction of the operator's time and are the source of frustrating and potentially dangerous errors. Smith and Mosier (1986) offer five high-level objectives for data entry:

1. *Consistency of data-entry transactions* Similar sequences of actions should be used under all conditions; similar delimiters, abbreviations, and so on should be used.
2. *Minimal input actions by user* Fewer input actions mean greater operator productivity and—usually—fewer chances for error. Making a choice by a single keystroke, mouse selection, or finger press, rather than by typing in a lengthy string of characters, is potentially advantageous. Selecting from a list of choices eliminates the need for memorization, structures the decision-making task, and eliminates the possibility of typographic errors. However, if users must move their hands from a keyboard to a separate input device, the advantage is defeated because home-row position is lost. Experienced users often prefer to type six to eight characters instead of moving to a lightpen, joystick, or other selection device.

A second aspect of this guideline is that redundant data entry should be avoided. It is annoying for users to enter the same information in two