

ESD-TR-83-122

MTR-8857

12

DESIGN GUIDELINES
FOR
THE USER INTERFACE TO COMPUTER-BASED INFORMATION SYSTEMS

By
S. L. SMITH
A. F. AUCELLA

MARCH 1983

Prepared for
DEPUTY FOR ACQUISITION LOGISTICS AND TECHNICAL OPERATIONS
ELECTRONIC SYSTEMS DIVISION
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE
Hanscom Air Force Base, Massachusetts



DTIC
APR 26 1983
H

Project No. 522A

Prepared by

THE MITRE CORPORATION
Bedford, Massachusetts

Contract No. F19628-82-C-0001

Approved for public release;
distribution unlimited.

ADA127345

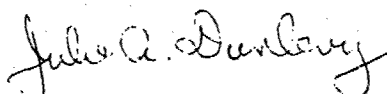
DTIC FILE COPY

When U.S. Government drawings, specifications, or other data are used for any purpose other than a definitely related government procurement operation, the government thereby incurs no responsibility nor any obligation whatsoever; and the fact that the government may have formulated, furnished, or in any way supplied the said drawings, specifications, or other data is not to be regarded by implication or otherwise, as in any manner licensing the holder or any other person or corporation, or conveying any rights or permission to manufacture, use, or sell any patented invention that may in any way be related thereto.

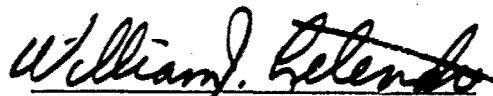
Do not return this copy. Retain or destroy.

REVIEW AND APPROVAL

This technical report has been reviewed and is approved for publication.

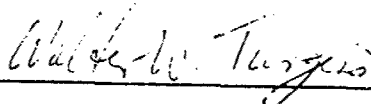


JULIE A. DUNLEVY, 1Lt, USAF
Project Manager



WILLIAM J. LETENDRE
Program Manager, Computer
Resource Management Technology

FOR THE COMMANDER



WALTER W. TURGISS
Director, Engineering and Test
Deputy for Acquisition Logistics
and Technical Operations

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER ESD-TR-83-122	2. GOVT ACCESSION NO. AD A127345	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) DESIGN GUIDELINES FOR THE USER INTERFACE TO COMPUTER-BASED INFORMATION SYSTEMS	5. TYPE OF REPORT & PERIOD COVERED	
	6. PERFORMING ORG. REPORT NUMBER MTR-8857	
7. AUTHOR(s) SIDNEY L. SMITH, ARLENE F. AUCELLA	8. CONTRACT OR GRANT NUMBER(s) F19628-82-C-0001	
9. PERFORMING ORGANIZATION NAME AND ADDRESS The MITRE Corporation Burlington Road, Bedford, MA 01730	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS Project No. 522A	
11. CONTROLLING OFFICE NAME AND ADDRESS Deputy for Acquisition Logistics & Technical Operations, Electronic Systems Division, AFSC, Hanscom AFB, MA 01731	12. REPORT DATE March 1983	
	13. NUMBER OF PAGES 288	
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)	15. SECURITY CLASS. (of this report) UNCLASSIFIED	
	15a. DECLASSIFICATION/DOWNGRADING SCHEDULE	
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited.		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) COMPUTER-BASED SYSTEMS DESIGN GUIDELINES INFORMATION SYSTEMS USER-SYSTEM INTERFACE		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) The user-system interface (USI) to computer-based information systems can require a sizable investment in software design. In current practice there is still no coherent methodology for design of USI software, but efforts to establish design guidelines are continuing. This report revises and enlarges previous compilations of guidelines (Smith, 1982b), and proposes a total of 580 guidelines for design of USI software in six functional areas: data entry, data display, sequence control, user guidance, data transmission, and data protection.		

Acknowledgments

This report has been prepared by The MITRE Corporation under Project No. 522A. The contract is sponsored by the Electronic Systems Division, Air Force Systems Command, Hanscom Air Force Base, Massachusetts.

Accession For	
NTIS GRA&I	<input checked="" type="checkbox"/>
DTIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
By _____	
Distribution/	
Availability Codes	
Dist	Avail and/or
	Special
A	

DZIS
COPY
INSPECTED
2

TABLE OF CONTENTS

	<u>Page</u>
INTRODUCTION	1
SECTION 1 DATA ENTRY	13
1.0 General	15
1.1 Position Designation	25
1.2 Direction Designation	31
1.3 Text	31
1.4 Data Forms	32
1.5 Tabular Data	41
1.6 Graphic Data	43
1.7 Data Validation	44
1.8 Other Data Processing	47
1.9 Design Change	49
SECTION 2 DATA DISPLAY	51
2.0 General	55
2.1 Data Type	58
2.1.1 Text	58
2.1.2 Data Forms	66
2.1.3 Tables	70
2.1.4 Graphics	74
2.1.5 Combination	74
2.2 Data Selection	75
2.3 Data Aggregation	76
2.4 Display Generation	78
2.5 Display Partitioning	79
2.6 Display Density	82
2.7 Display Coding	84
2.8 Display Coverage	96
2.9 Display Update	100
2.10 Display Suppression	102
2.11 Design Change	102

TABLE OF CONTENTS
(Continued)

	<u>Page</u>
SECTION 3 SEQUENCE CONTROL	103
3.0 General	107
3.1 Dialogue Type	116
3.1.1 Question and Answer	117
3.1.2 Form Filling	117
3.1.3 Menu Selection	118
3.1.4 Function Keys	131
3.1.5 Command Language	136
3.1.6 Query Language	142
3.1.7 Natural Language	144
3.1.8 Graphic Interaction	144
3.2 Transaction Selection	145
3.3 Interrupt	151
3.4 Context Definition	154
3.5 Error Management	157
3.6 Alarms	162
3.7 Design Change	163
SECTION 4 USER GUIDANCE	165
4.0 General	169
4.1 Status Information	179
4.2 Routine Feedback	182
4.3 Error Feedback	187
4.4 Job Aids	194
4.5 User Records	203
4.6 Design Change	206
SECTION 5 DATA TRANSMISSION	207
5.0 General	209
5.1 Data Type	211
5.2 Source	213
5.3 Destination	214
5.4 Transmission Control	216
5.5 Feedback	218
5.6 Queuing	219
5.7 Record Keeping	221
5.8 Design Change	221

TABLE OF CONTENTS
(Concluded)

	<u>Page</u>
SECTION 6 DATA PROTECTION	223
6.0 General	229
6.1 User Identification	233
6.2 Data Access	235
6.3 Data Entry/Change	238
6.4 Data Transmission	245
6.5 Loss Prevention	247
6.6 Design Change	254
 REFERENCES	 255
 GLOSSARY	 267
 INDEX	 271

INTRODUCTION

In computer-based information systems developed by the Air Force Electronic Systems Division (ESD), special attention must be given to design of the user-system interface (USI). Guidelines for the improvement of USI design are being compiled as a continuing effort under ESD sponsorship in Air Force Program Element PE 64740F. Three previous ESD reports have dealt with this subject (Smith, 1980; 1981a; 1982b).

This present report adapts and enlarges previously published material, and proposes a more comprehensive set of guidelines for design of USI software in computer-based information systems. Discussion of information systems is couched in general terms throughout this report, to acknowledge that USI design problems and their possible solutions are common to many different systems. Guidelines for USI design proposed here will be evaluated in their specific application to ESD development of Air Force command, control and communication systems.

INFORMATION SYSTEMS

Current use of computers covers a broad range of applications. At one extreme are large, general-purpose computer installations, used by different people for different purposes. Users are expected to provide exact instructions (a computer program) in order to request data processing. Many users have programming skills. All users must have some knowledge of the system and its capabilities.

At the other extreme are the small, special-purpose computers used increasingly as components within larger systems. Such component computers may regulate the ignition of an automobile, or the tuning of a television set. Their net effect is to help make system operation easier. The user does not interact directly with component computers, and indeed may not even know they are there.

Between these extremes of general-purpose and component computers, there are a variety of applications in which computers are used to support what are commonly called information systems. Information systems are task-oriented rather than general-purpose, being designed and dedicated to help perform defined jobs. Applications range from relatively simple data entry and retrieval (e.g., airline reservations) through more complex monitoring and control tasks (inventory control, process control, air traffic control), to jobs requiring long-term analysis and planning. Military command systems fall within this range of applications.

The users of information systems must interact with a computer in some explicit fashion to accomplish the data handling tasks needed to get their jobs done. The users of information systems differ in ability, training and job experience. Users may be keenly concerned with task performance, but they probably have little knowledge of (or interest in) their computer tools. Design of the user-system interface must take these human factors into account.

USER-SYSTEM INTERFACE (USI)

What is the user-system interface? In accord with recommended usage (Smith, 1982a), here the phrase is defined broadly to include all aspects of system design that affect a user's participation in data handling transactions. Directly observable aspects of the USI include the physical work environment and the hardware facilities at the user's work station. Such physical aspects, sometimes called the man-machine interface, have been the subject of conventional human engineering study, where the concern is for illumination, seating, workplace arrangement, keyboard layout, display contrast, symbol size, etc. Good design of the physical workplace is important, of course, but by itself is not sufficient to ensure effective job performance.

Also important are less tangible aspects of information system design -- the ways in which data are stored and manipulated, including paper files and forms, if any, and the procedures and processing logic that govern data handling transactions. Forms, procedures and logic involve software design, the design of computer programs to permit hardware and paper to be used in conjunction with automated data processing.

If the USI is conceived in these broad terms, to encompass all factors influencing user-system interaction, then to say that the USI is critical to successful system operation is to state the obvious. In any automated information system, whether its work stations are used for data input, calculation, planning, management or control, effective USI design is required for effective performance. Task analysis, review of operating procedures, equipment selection, workspace configuration, and especially the design of USI software -- all must be handled with care.

USI SOFTWARE

The critical role of USI software in system design poses a special challenge to human factors specialists, recognized a decade ago by Parsons:

. . . what sets data processing systems apart as a special breed? The function of each switch button, the functional arrangement among the buttons, the size and distribution of elements within a display are established not in the design of the equipment but in how the computer is programmed. Of even more consequence, the 'design' in the programs establishes the contents of processed data available to the operator and the visual relationships among the data. In combination with or in place of hardware, it can also establish the sequence of actions which the operator must use and the feedback to the operator concerning those actions.

(1970, page 169)

Current interest in USI software design is reflected in phrases such as "software psychology" (cf. Shneiderman, 1980). But USI design cannot be the concern only of the psychologist or the human factors engineer. It is a significant part of information system design that must engage the attention of system developers, designers, and ultimately system users as well.

Not only is USI software design critical to system performance, it can also represent a sizable investment of programming effort during initial system development, and during subsequent system modification ("software maintenance") to accommodate changing operational requirements thereafter. Just how sizable is that investment? The answer, of course, will depend upon the particular type of information system being considered.

In a recent survey (Smith, 1981b), people involved in the design of information systems were asked to estimate the percent of operational software devoted to implementing the USI. Overall, the average estimate was that USI design comprises a bit more than 30 percent of operational software. Estimates for individual systems ranged from 3 to 80 percent, reflecting our common experience that some computer systems require a much higher investment in USI design than others, depending upon their purpose.

DESIGN DEFICIENCIES

Given the broad definition of the user-system interface proposed here, it is obvious that deficiencies in USI design can result in degraded system performance. To be sure, users can sometimes compensate for poor design with extra effort. Probably no single USI design flaw, in itself, will cause system failure. But there is a limit to how well users can adapt to a poorly designed interface. As one deficiency is added to another, the cumulative

negative effects may eventually result in system failure, poor performance, and/or user complaints.

Outright system failure may sometimes result in abandoned installations, or decreasing use of an automated information system where use is optional. There may be retention of (or reversion to) manual data handling procedures, with little use of automated capabilities. When a system fails in this way, the result is disrupted operation, wasted time, effort and money, and failure to achieve the potential benefits of automated data handling.

In a constrained environment, such as that of many military and commercial information systems, the user may have little choice but to make do with whatever interface design is provided. Here the symptoms of poor USI design may appear in degraded system performance. Frequent and/or serious errors in data handling may result from confusing USI design. Tedious user procedures may slow data processing, resulting in longer queues at the checkout counter, the teller's window, the visa office, the truck dock, or any other workplace where the potential benefits of computer support are outweighed by an unintended increase in human effort.

In situations where degradation in system performance is not so easily measured, symptoms of poor USI design may appear as user complaints. The system may be described as hard to learn, or clumsy, tiring and slow to use. The user's view of the system is conditioned chiefly by experience with its interface. If USI design is unsatisfactory, the user's image of the system will be negative, regardless of any niceties of internal computer processing.

USI design deficiencies arise from problems in system development. Different portions of the USI design may be implemented by different people, who share no common view of desired operational procedures. One result is design inconsistency. When users must handle different tasks differently, or even different transactions within the same task differently, an unnecessary burden is imposed on both learning and use of the system.

In some applications, system developers must worry about design inconsistency among different systems. Within a large industrial firm, or within a government agency or a military service, many different information systems may be developed for different purposes in different parts of the organization. Examples of inter-system inconsistencies of USI design have been documented in a study of battlefield information systems, sponsored by the Army Research Institute (Sidorsky and Parrish, 1980).

Where USI design is established independently for each system, the result can be to impose a burden of incompatible habits on the user who must move from one system to another. Conversely, some standardization of USI design across systems might help to reduce user learning problems and improve user performance.

DESIGN PRACTICE

Looking at current design practice, it seems fair to characterize present methods of USI software design as art rather than science, depending more upon individual judgment than systematic application of knowledge (Ramsey and Atwood, 1979; 1980). As an art, USI design is best practiced by experts, by specialists experienced in the human engineering of computer systems. But such experts are not always available to help guide system development, and it is clear that they cannot personally guide every step of USI design. What seems needed is some way to embody expert judgment in the form of explicit guidelines for USI design. Unfortunately, there has been little effective guidance for USI design in recent practice.

For military information systems, Military Specification MIL-H-48655B (1979) calls for a system development sequence starting with requirements analysis, functional specification and design verification. The actual course of USI software design has sometimes departed from this desired sequence. There may be no explicit attempt to determine USI requirements. Specifications may include only rudimentary references to USI design, with general statements that the system must be "easy to use". In the absence of more effective guidance, both the design and implementation of USI software may become the responsibility of programmers unfamiliar with operational requirements. USI software may be produced slowly, while detection and correction of design flaws occur only after system prototyping, when software changes are difficult to make.

Human engineering standards and design handbooks have been of little use to the software designer. The recently published human factors design handbook by Woodson (1981) is typical. Its nearly 1000 pages include only three pages of general material on information processing, and there is no reference to computer systems in its index.

MIL-STD-1472B (1974), for many years a major human engineering design standard for military system procurement, was concerned almost exclusively with hardware design and physical safety. In 1981, MIL-STD-1472 was published in a revised "C" version. That new military standard includes nine pages dealing with USI software

design, in Section 5.15 titled "Personnel-Computer Interface", and that material will be expanded in future revisions. Thus a beginning has been made, but much more is needed. There are still no comprehensive guidelines available for USI software design. The question is, can needed USI design guidance be developed?

USI DESIGN GUIDELINES

Until several years ago, there had been no serious attempt to integrate the scattered papers, articles and technical reports that constitute the literature of user-computer interaction. A first step was made, under sponsorship of the Office of Naval Research, in compilation of an extensive bibliography on this subject (Ramsey, Atwood and Kirshbaum, 1978). A significant follow-on effort culminated in publication by Ramsey and Atwood (1979) of a comprehensive summary of this literature.

In reviewing the literature, it becomes apparent that most published reports dealing with the user-computer interface describe applications rather than design principles. The ONR bibliography cited above includes 564 items, but identifies only 17 as offering design guidelines. A popular book on the design of user-computer dialogues offers stimulating examples, covering a range of on-line applications, but is disappointing in its failure to emphasize design principles (Martin, 1973).

Although accepted principles for USI design have not been available, some work toward guidelines development has been accomplished. In the past decade, as increasing experience has been gained in the use of on-line computer systems, a number of experts have attempted to set forth principles ("guidelines", "ground rules", "rules of thumb") for design of the user-computer interface. That work has not produced a comprehensive set of guidelines, but it does offer a foundation on which to build. In aggregate, the evidence of concern for USI design principles is encouraging, but there is still much to be learned.

Military agencies, of course, are not the only organizations seeking guidelines for USI design. There is increasing interest in this topic within industrial and commercial organizations, and throughout the general community of people who develop and use information systems. David Penniman, writing for the User On-Line Interaction Group of the American Society for Information Sciences, cites the need for "an interim set of guidelines for user interface design based on available literature and pending the development of better guidelines as our knowledge increases" (1979, page 2). Penniman goes on to remind us that interim guidelines are better than no guidelines at all.

In a survey of people concerned with USI design (Smith, 1981b), respondents generally support Penniman's activist position. Given a choice between trying to develop a complete set of USI guidelines now, when many of them must be based on judgment rather than experimental data, or else accepting only a partial set of guidelines based on evaluated research, most respondents would go with judgment now.

It is clear, of course, that system developers cannot wait for future research data in making present design decisions. To meet current needs, a number of in-house handbooks have recently been published to guide USI design within particular organizations (NASA, 1979; Galitz, 1980; Brown, Burkleo, Mangelsdorf, Olsen and Williams, 1981; Parrish, Gates, Munger, Grimma and Smith, 1982). These in-house guidelines draw heavily from earlier publications, especially the influential IBM report by Engel and Granda (1975), usually modified by the authors' own good judgment. They will help system developers until such time as a comprehensive USI design standard becomes available.

The ESD/MITRE compilation of USI design guidelines has been built on the work of our predecessors, and will help support the work of followers to come. Each year our compilation has grown larger. In our last report (Smith, 1982b) there were 375 guidelines. In this present report there are 580. Future reports will include even more. This present compilation, although still by no means complete, represents the most comprehensive published guidance available for USI software design, and for that reason is recommended as a reference in current system acquisition programs.

GUIDELINES ORGANIZATION

The present total of 580 guidelines constitutes a formidable compendium, posing serious problems in deciding how to organize guidelines material. In the numbered sections of this report, guidelines have been organized within six functional areas:

<u>Section</u>	<u>USI Functions</u>	<u>Number of Guidelines</u>
1	Data Entry	98
2	Data Display	141
3	Sequence Control	145
4	User Guidance	94
5	Data Transmission	35
6	Data Protection	67

Within each functional area, guidelines are grouped by more specific functions, which are listed in the table of contents for this report. The overall organization of guidelines follows the structure established in a checklist of functional capabilities that has been developed for USI requirements definition (see Smith, 1982b).

Organizing guidelines in relation to USI functions offers several advantages. This organization could facilitate association of guidelines with required functions, and tailoring guidelines to the differing functional requirements of different systems. As a designer considers each specified function, pertinent guidelines can be readily referenced. Conversely, the designer can quickly determine where guidance is not relevant to functional requirements.

Functional organization of guidelines also shows us those USI functions for which little design guidance is yet available, to help direct future work in guidelines development. In this past year we have tried more to extend the range of guidelines coverage than to fill in the missing pieces. There are some functions such as text entry and graphical data entry that are still not covered at all, and other functions where coverage is weak. In future work we hope to remedy those deficiencies.

A notable limitation of the present guidelines is the almost exclusive concern with electronic visual displays as the input/output medium for user-system interaction. Additional guidelines will be needed for auditory displays. Additional guidelines would also be needed for USI designs employing a teletype or reactive typewriter, with paper printouts rather than electronic displays.

The design of USI software is necessarily dependent in some degree on the means chosen for hardware implementation. That is acknowledged in the contingent wording of some of the guidelines listed here. And certainly there are hardware features implied in some of these guidelines, including function keys for ENTER, CONFIRM, BACKUP, CANCEL, DITTO, PRINT, etc., plus various tab keys for controlling cursor position in USI designs involving form-filling dialogues. For the most part, however, the guidelines are stated in terms of software function rather than hardware specification. Such functional guidelines could be amplified to include more explicit recognition of hardware features, if that should prove appropriate in system acquisition.

One significant weakness in current design guidelines is that they are based on opinion, judgment, accumulated experience, rather than on quantitative performance measures. The designer is given

good advice but not told the consequences of ignoring it. Until research on user-system interaction catches up with application, which may take a long time, this weakness cannot be remedied.

GUIDELINES FORMAT

Each section of the guidelines begins with a general discussion of design principles. General principles, however, can only guide the designer a little way. Exhortations to provide flexibility, to preserve context, to ensure compatibility between input and output, are not enough. More specific guidance is needed, and the guidelines that follow the introductory discussion are couched in more specific terms.

No matter how specifically guidelines are stated, however, they will still be subject to misinterpretation. Details of wording and format will affect the understanding and use of design guidelines (Rogers and Pegden, 1977). For that reason, therefore, the format adopted in this report for presentation of guidelines deserves some further description.

Guidelines are numbered sequentially within function, in order to permit convenient referencing. Each guideline is stated as a single sentence. In some instances a stated guideline is amplified by one or more examples. Where validity of a guideline is contingent upon special factors, exceptions may be noted. Where further clarification seems needed, further comments may be added. Where a guideline is derived from a particular source, a reference note is added. (Those references are listed at the end of this report.) Finally, where guidelines are specifically related to one another, cross references to other guidelines are given.

One consequence of all this notation is that a guideline that began as a simple statement can end up filling half a page. As the bulk of guidelines material increases, good format becomes even more critical for efficient guidelines use.

In comparison with previous reports, some new formatting features have been added to the present guidelines compilation. Numeric function codes are highlighted in running headers, to help readers find guidelines pertinent to any particular USI function. Each guideline has been given a short title, to help readers find guidelines relating to more specific topics, a good idea adopted from the guidelines format used by Brown et al. (1981).

At the end of this report, following the reference list, there is a glossary defining word usage in the guidelines, an attempt to

encourage consistent terminology in USI design. There is also a topical index, to help readers look up guidelines on a particular subject independently of the imposed functional organization of guidelines material.

FUTURE DEVELOPMENT

Future efforts at MITRE will be directed toward revising and expanding the USI design guidelines presented here. Critical review is needed. If many of our present guidelines are based on judgment, then a broad range of judgment will be needed to assess them. Improvement of design guidelines will require a collaborative effort with other people working on user interface design.

Readers are encouraged to recommend changes to the guidelines proposed here, by making copies of the suggestion sheet that is appended as the last page in this report. Do some of the guidelines seem wrong? Are some unclear? Can you suggest additions or changes to the guidelines, better wording, further examples, exceptions, references to supporting data? If so, please contribute your ideas.

In addition to revision and expansion of the guidelines, it is necessary to assess their usefulness in practical application to USI design. Trial application of USI design guidelines is presently being explored in Air Force system acquisition programs. If you find an opportunity to apply these guidelines for USI design in your own system development work, please contact the authors of this report, so that we can share information.

Meanwhile, any final evaluation of USI design guidelines, by system developers, designers and users, still lies in the future. Whatever guidelines are proposed at this stage can only be regarded as tentative, to be used on an interim basis until experience is gained in their application.

Once a comprehensive set of guidelines has been prepared and evaluated, the appropriate form for codification will probably be as a design handbook. Such a handbook could be referenced for optional design guidance in system development, or perhaps adopted by some organizations as an agreed standard for in-house system design. With increased experience in its use, a USI design handbook might become accepted as a more general design standard, to be used for formal design evaluation as well as for guidance.

An accepted standard will not in itself guarantee good design. This is true in the case of human engineering standards for hardware design (Rogers and Armstrong, 1977), and will presumably prove true

for future design standards pertaining to USI software. That is no reason to abandon standards. It simply means that they must be used with care, with continual evaluation of their application.

The potential value of standards for interface design is generally recognized in system engineering, where communication standards are needed to ensure the effective linking of systems, and of system components. USI design standards may prove equally important for the effective linking of a system with its users.

There is one important factor to consider in this regard. Design standards for hardware interfaces may change as each new generation of equipment becomes available. But people change relatively little from one generation to the next, in terms of their basic information processing capabilities and limitations. This relative invariance of people with respect to technology poses both problems and advantages. The most significant advantage is this: if USI guidelines can be expressed in terms of basic human capabilities rather than transient technology, a design standard of enduring value will be established.

SECTION 1

DATA ENTRY

Data entry refers to input by the user of data items to be processed. Command inputs or option selections intended to control data processing are considered separately in the discussion of sequence control (Section 3). Data entry is heavily emphasized in tasks related to clerical jobs, and many other tasks involve data entry to some degree. Because data entry is so common, because the requirements of data entry seem to be readily understood, and because inefficiencies caused by poorly designed data entry are so apparent, many of the published recommendations for good USI design deal with this topic.

Design of data input transactions is necessarily influenced by hardware selection. For that reason, design guidelines for input devices receive considerable attention. A notable example is standardization of keyboard layouts. Future technological advances in input hardware may well influence the design of data entry tasks, presaged perhaps by the current advocacy of voice input. But the major need in information systems is for consistently good software design. It is in improving the logic of data entry that the chief gains can be made, and it is here that design guidance should prove most helpful.

The general objectives of designing data entry functions are to establish consistency of data entry transactions, to minimize input actions and memory load on the user, to ensure compatibility of data entry with data display, and to provide flexibility of user control of data entry. Some of these ideas seem so basic that they are seldom expressed as explicit design principles.

With regard to minimizing input actions, here is an example: a user should not have to enter the same data twice. That is probably something every designer knows, even if it is sometimes forgotten. A corollary is this: a user should not have to enter a data item already entered by another user. That seems to be good common sense, although one could imagine occasional exceptions to the rule when cross validation of data inputs is required.

How can duplicative data entry be avoided in practice? The solution lies in designing the USI (i.e., programming the computer) to maintain context. Thus when a user identifies a particular category of interest, for example a squadron of aircraft, the computer should be able to access all previously entered data relevant to that squadron and not require the user to enter such data again.

In repetitive data entry transactions the user should have some means of establishing context, for example by defining default entries for selected data items, in effect telling the computer those items will stay the same until the default value is changed or removed. If the user enters one item of data about a particular squadron, it should be possible to enter a second item immediately thereafter without having to re-identify that squadron.

Context should also be preserved to help speed correction of input errors. One significant advantage of on-line data entry is the opportunity for immediate computer validation of user inputs, with timely feedback so that the user can correct detected errors while that set of entries is still fresh in his mind and/or while documented source data are still at hand. Here the computer should preserve the context of each data entry transaction, saving correct items so that the user does not have to enter those again while changing incorrect items.

Preservation of context is, of course, important in all aspects of user-system interaction, with implications for data display, sequence control and user guidance, as well as for data entry. The importance of context is emphasized again in the discussion of those other functional areas.

Another important design concept is that of flexibility. The idea that USI design should adapt flexibly to user needs is often expressed. The means of achieving such flexibility should be spelled out in USI guidelines. For data entry functions it is important that the pacing of inputs be controlled flexibly by the user. Tasks where the pacing of user inputs is set by a machine (for example, keying ZIP codes at an "automated" post office) are stressful and error-prone.

Aside from flexibility in pacing, the user will often benefit from having some flexible choice in the ordering of inputs. Although this kind of flexibility is related to the topic of sequence control, it merits discussion here as well. What is needed for USI design is some sort of suspense file(s) to permit flexible ordering of user inputs, including temporary omission of unknown items, backup to correct mistaken entries, cancellation of incomplete transactions, etc.

As noted earlier, the user may also benefit from flexibility in defining default options to simplify data entry during a sequence of transactions. Some systems include only those defaults anticipated by the designers, which may not prove helpful to the user in a particular instance. These general concepts are represented in the specific design guidelines for data entry functions proposed in the following pages.

DATA ENTRY

- Objectives: Consistency of data entry transactions
Minimal entry actions by user
Minimal memory load on user
Compatibility of data entry with data display
Flexibility for user control of data entry

DATA ENTRY **General 1.0**

-1 Entry via Primary Display -1

When data entry is a significant part of a user's task, it should be accomplished via the user's primary display.

Example: Entry via typewriter is acceptable only if the typewriter itself, under computer control, is the primary display medium.

Comment: When the primary display is basically formatted for other purposes, such as a graphic display for process control, a separate "window" on the display may have to be reserved for data entry.

-2 Displayed Feedback -2

Keyed entries should always appear in the display.

Exception: Passwords and other secure entries.

Reference: EG 6.3.7; MS 5.15.3.9.4.2.

See also: 4.2-1.

-3 Single Method of Entry

-3

Data entry transactions, and associated displays, should be designed so that the user can stay with one method of entry as long as necessary for the data entry task, before having to shift to another.

Example: Shifts from lightpen to keyboard entry and then back again should be minimized.

Comment: This, like other guidelines here, assumes a task-oriented user, busy or even overloaded, who needs efficiency of data entry.

Reference: EG 6.1.1.

-4 Data Entry by Character Replacement

-4

Keyed data entry on an electronic display should generally be accomplished by direct character replacement, in which keyed entries replace underscores (or other delimiter symbols) in defined data fields.

Exception: For general text entry, no field delimiters are needed. In effect, keyed text entries can replace nothing (null characters).

Comment: This feature permits display formats providing explicit user guidance as to the location and extent of data entry fields.

-5 Consistent Method for Data Change

-5

For data change, a consistent method should be adopted, in which new entries replace previous entries (including default values, if any) either by direct character substitution, or by insertion and deletion.

Example: Text editing can be handled either way.

Comment: In many cases, direct modification of displayed data will reduce keying and permit more compact display formats. There may be some risk of user confusion, however, in replacement of an old value with a new one, during the transitional period when the item being changed is seen as a composite beginning with the new value and ending with the old.

Comment: In some applications it may help the user to key a new entry directly above or below display of the prior entry it will replace, if that is done consistently. Here the user can compare values before confirming entry of the new data and deletion of the old.

-6 User-Paced Data Entry

-6

Whenever possible, data entry should be user-paced, depending upon the user's needs, attention span and time available, rather than computer processing or external events.

Comment: When user pacing does not seem feasible, the general approach to task allocation and USI design should be reconsidered.

-7 Computer Processing Delay

-7

Data entry should not be slowed or paced by delays in computer response; for normal operation, delays or lockouts should not exceed 0.2 seconds.

Example: Key press followed by display of symbol.

Comment: This recommendation is intended to ensure efficient operation in routine, repetitive data entry tasks. Somewhat longer delays may be tolerable in special circumstances, perhaps to reduce variability in computer response, or perhaps in cases where data entry comprises a relatively small portion of the user's task.

Reference: EG Table 2.

See also: 3.0-9.

-8 Explicit ENTER Action

-8

Data entry should always require an explicit ENTER action, and not be accomplished as a side effect of some other action.

Example: Returning to a menu of control options should not by itself result in computer processing of data just keyed onto a display.

Comment: This practice permits the user to review data and correct errors before computer processing, particularly helpful when data entry is complex and/or difficult to reverse.

Comment: In text entry, users might be expected to detect and correct 90 percent of their keying errors before taking a final ENTER action.

Reference: Neal and Emmons, 1982.

See also: 3.0-5, 4.0-2, 6.0-3, 6.3-8.

-9 ENTER Key Labeling

-9

An ENTER key should be explicitly labeled to indicate its function to the user.

Example: The ENTER key should not be labeled in terms of mechanism, such as CR or RETURN or XMIT.

Comment: For a computer-naive user, the label should perhaps be even more explicit, such as ENTER DATA. Ideally, one consistent ENTER label would be adopted for all systems and so become familiar to all users.

Comment: Some other label might serve as well, if it were used consistently. In some current systems the ENTER key is labeled GO, implying a generalized command to the computer, "Go off and process it."

Reference: PR 3.3.9.

See also: 4.0-13.

-10 Displayed Feedback for Changing Data

-10

When a user requests change (or deletion) of a stored data item that is not currently being displayed, then both the old and new values should be displayed so that the user can confirm or nullify the change before the transaction is completed.

Comment: This practice will tend to prevent inadvertent change, and is particularly useful in protecting delete actions. Like other recommendations intended to reduce error, it assumes that accuracy of data entry is worth extra keying by the user. For some tasks, that may not be true.

See also: 6.3-19, 6.5-10.

-11 Short Data Items

-11

Ideally, the length of an individual data item should not exceed 5-7 characters.

Exception: Meaningful words and general textual material.

Comment: For coded data, longer strings exceed the user's memory span, inducing errors in both data entry and data review.

Reference: BB 2.5.2; EG 6.3.3.

-12 Partitioning of Data Items

-12

When a long data item must be entered, it should be partitioned into shorter symbol groups for both entry and display.

Example: A 10-digit telephone number can be entered as three groups, NNN-NNN-NNNN.

-13 Entry Sequence for Partitioned Data Items

-13

When portions of a long data item are highly familiar, ideally those should be entered last.

Exception: But not if that sequence would violate a functional requirement, such as initial keying of area code in telephone numbers, or if common usage puts the familiar first.

Comment: This practice will reduce the load on the user's short-term memory. The point here is really to enter the unfamiliar items, those difficult to remember, first before they are forgotten.

Reference: EG 6.3.4.

-14 Minimal Keying

-14

Minimize data entry keying by abbreviating lengthy data items, when that can be done without ambiguity.

Comment: Some flexibility should be provided for users of different ability. Novice and/or occasional users may prefer to make full-form entries, while experienced users will learn and benefit from appropriate abbreviations.

Reference: BB 6.4.1; EG 6.3.5.

-15 Consistent Abbreviation Rule

-15

When abbreviations are used to shorten data entry, those abbreviations should follow some consistent rule that can be explained to the user.

Example: Simple truncation is probably the best choice.

Comment: It is important for both encoding and decoding abbreviations that the user know what the rule is. Truncation provides inexperienced users with a straightforward and highly successful method for generating abbreviations, and is a rule that can be easily explained. Moreover, truncation works at least as well as more complicated rules involving word contraction with omission of vowels, etc.

Reference: Moses and Ehrenreich, 1981.

-16 Abbreviation Length

-16

Abbreviations should be of fixed length.

Comment: Desirable length will depend upon the vocabulary size of words to be abbreviated. For a vocabulary of 75 words, 4-letter abbreviations will suffice. For smaller vocabularies, shorter abbreviations can be used.

Reference: Moses and Ehrenreich, 1981.

-17 Special Abbreviations

-17

Special abbreviations (i.e., those not formed by consistent rule) should be used only when required for clarity.

Comment: Special abbreviations will be needed to distinguish between words whose abbreviations by rule are identical, or when abbreviation by rule forms another word, or when the special abbreviation is already familiar to system users. Such special cases should represent less than 10 percent of all abbreviations used.

Reference: Moses and Ehrenreich, 1981.

-18 Minimal Deviation from Abbreviation Rule

-18

When an abbreviation must deviate from the consistent rule, the extent of deviation should be minimized.

Example: Letters in the truncated form should be changed one at a time until a unique abbreviation is achieved.

Reference: Moses and Ehrenreich, 1981.

-19 Resolving Ambiguous Abbreviation

-19

When abbreviated data entries are not recognized, the computer should apply data validation routines and interrogate the user as necessary to resolve any ambiguity.

See also: 6.0-7.

-20 Distinctive Abbreviation

-20

When abbreviations or other codes are used to shorten data entry, they should be designed to be meaningful and distinctive in order to avoid potentially confusing similarity.

Example: BOS vs. LAS is good; but LAX vs. LAS risks confusion.

-21 Codes Entered by Menu Selection**-21**

When arbitrary codes must be entered, menu selection should be considered as an appropriate dialogue mode, rather than user-composed entry.

Comment: Menu selection does not require the user to remember codes.

Reference: Siebel, 1972; Gade, Fields, Maisano, Marshall and Alderman, 1981.

See also: Section 3.1.3.

-22 Single Keying for Alphabetic Data**-22**

When alphabetic data entry is required, the user should be able to enter each letter with a single stroke of an appropriately labeled key.

Comment: More complex double-keying methods will require special user training, and will risk frequent data entry errors.

Comment: Software might be provided to interrogate the user to resolve any ambiguities in data entry resulting from hardware limitations, such as when each key of a panel used primarily for numeric entry is labeled with several letters.

Reference: Smith and Goodwin, 1971a; Butterbaugh, 1982.

-23 Minimal Shift Keying**-23**

Special characters requiring shift keying should be avoided insofar as possible.

Comment: Conversely, keyboard designers should put frequently used special characters where they can be easily keyed.

Reference: EG 6.3.12.

-24 Entry of Leading Zeros

-24

Entry of leading zeros should be optional for general numeric data.

Example: If a user enters "56" in a field that is four characters long, the system should recognize the entry rather than requiring that "0056" be entered.

Exception: Special cases such as entry of serial numbers or other numeric identifiers.

Reference: BB 6.2.3; EG 6.3.11.

-25 Entry of Blanks

-25

There should be no distinction between single and multiple blanks in data entry; in particular, the user should not have to count blanks.

Comment: People cannot be relied upon to pay careful attention to such details. The computer should handle them automatically, e.g., ensuring that two spaces follow every period in text entry (if that is the desired convention), and spacing other data items to whatever format has been defined.

See also: 3.1.5-14.

-1 Distinctive Cursor

-1

Position designation on an electronic display should be accomplished by means of a movable cursor with distinctive visual features (shape, blink, etc.).

Exception: When position designation involves only selection among displayed alternatives, then some form of highlighting selected items might be used instead of a separately displayed cursor.

Comment: If multiple cursors are used (e.g., one for alphanumeric entry, one for tracking, one for line drawing), they should be visually distinctive from one another.

See also: 4.0-12.

-2 Non-Obscuring Cursor

-2

The cursor should be designed so that it does not obscure any other character displayed in the position designated by the cursor.

-3 Fine Positioning Cursor

-3

When fine accuracy of positioning is required, as in some forms of graphic interaction, the displayed cursor should include a point designation feature.

Example: A cross may suffice (like cross-hairs in a telescope), or perhaps a notched or V-shaped symbol (like a gun sight).

-4 Explicit Activation

-4

Actual entry ("activation") of a designated position should be accomplished by an explicit user action distinct from cursor placement.

Exception: Tracking tasks and other situations where the need for rapid entry may override the need to reduce entry errors.

Reference: MS 5.15.3.9.2.4; Albert, 1982.

See also: 6.0-3.

-5 Rapid Feedback

-5

Computer acceptance of a designated position should be signaled by direct feedback to the user within 0.2 seconds.

Example: Almost any consistently programmed display change will suffice, perhaps brightening or flashing a selected symbol; in some applications it may be desirable to provide an explicit message indicating that a selection has been made.

Reference: EG Table 2; MS 5.15.1.4.a.

See also: 4.2-2, 4.2-10.

-6 Consistent Starting Point

-6

If there is a predefined HOME position for the cursor, which is usually the case, that position should be consistent on displays of a given type.

Example: HOME might be in the upper left corner of a text display, or at the first field in a form-filling display, or at the center of a graphic display.

Comment: The HOME position of the cursor should also be consistent in different windows/sections of a partitioned display.

See also: 4.4-12.

-7 Cursor Control Response

-7

For arbitrary position designation, the cursor control should permit both fast movement and accurate placement.

Comment: Rough positioning should take no more than 0.5 seconds for a displacement of 20-30 cm on the display. Fine positioning may require incremental stepping of the cursor, or a control device incorporating a large control/display ratio for small displacements, or a selectable vernier mode of control use.

Reference: EG 6.1.

-8 Stable Cursor

-8

The displayed cursor should be stable, i.e., should remain where it is placed until moved by the user (or computer) to another position.

Comment: Some special applications, such as aided tracking, may benefit from computer-controlled cursor movement. The intent of the recommendation here is to avoid unwanted "drift".

Reference: EG 6.1.

-9 Incremental Cursor Positioning

-9

When cursor positioning is incremental by discrete steps, the step size of cursor movement should be consistent in both right and left directions, and both up and down directions.

-10 Variable Step Size

-10

When character size is variable on the display, incremental cursor positioning should have a variable step size corresponding to the size of currently selected characters.

-11 Proportional Spacing

-11

If proportional spacing is used for displayed text, the computer should be programmed to make necessary adjustments automatically when the cursor is being positioned for data entry or data change.

Example: Editing text composed for type setting.

Exception: Manual override may help the user in special cases where automatic spacing is not wanted.

Comment: The user cannot be relied upon to handle proportional spacing accurately.

-12 Continuous Cursor Positioning -12

Continuous position designation, such as used for entry of track data, should be accomplished by continuously operable controls (e.g., thumb wheel for one dimension, joystick for two dimensions) rather than by incremental, discrete key actions.

-13 Direct Pointing -13

When position designation is the sole or prime means of data entry, as in selection of displayed alternatives, cursor placement should be accomplished by a direct-pointing device (e.g., lightpen) rather than by incremental stepping or slewing controls (e.g., keys, joystick, etc.).

Reference: Albert, 1982.

-14 Pointing Area for Option Selection -14

In selection of displayed alternatives, the acceptable area for cursor placement should be made as large as consistently possible, including at least the area of the displayed label plus a half-character distance around the label.

Comment: The larger the effective target area, the easier the pointing action will be, and the less risk of error in selecting the wrong label by mistake.

Reference: EG 2.3.13, 6.1.3.

-15 Cursor Control by Keyboard -15

When position designation is required in a task emphasizing keyed data entry, cursor movement should be controlled by some device integral to the keyboard (function keys, joystick, "cat", etc.) rather than by a separately manipulated device (lightpen, "mouse", etc.).

-16 Minimal Use of Multiple Cursors -16

Multiple cursors should be displayed only when justified by careful task analysis.

Comment: Multiple cursors may confuse a user, and so require special consideration if advocated in USI design.

-17 Distinctive Control of Multiple Cursors -17

If multiple cursors are controlled by a single device, then a clear signal must be provided to indicate to the user which cursor is currently under control.

-18 Compatible Control of Multiple Cursors -18

If multiple cursors are controlled by different devices, their separate controls should be compatible in operation.

Reference: Morrill and Davies, 1961.

-19 Automatic Cursor Positioning -19

On initial appearance of a form-filling data entry display, the cursor should be placed automatically at the first character position of the first entry field.

See also: 1.4-25, 4.4-12.

-20 Minimal Cursor Positioning -20

Displays for form-filling data entry should be designed so as to minimize user actions required for cursor movement from one entry field to the next.

-21 Sequential Cursor Positioning -21

Sequential cursor positioning in predefined areas, such as displayed data entry fields, should be accomplished by programmable tab keys.

Comment: Automatic cursor advance is generally not desirable.

See also: 1.4-13.

-22 Display Format Protection

-22

Areas of a display not needed for data entry (such as labels and blank spaces) should be made inaccessible to the user, under computer control, so that the cursor does not have to be stepped through blank areas nor are they sensitive to pointing actions.

Exception: When it is expected that a user may have to modify display formats, such automatic format protection can be handled as a general default option subject to user override.

Comment: Mechanical overlays on the display should not be used for format protection.

Reference: EG 7.5; PR 3.3.2.

See also: 1.4-7, 2.0-6, 6.2-5.

-23 Data Entry Independent of Cursor Placement

-23

An ENTER action for multiple data items should result in entry of all items, regardless of where the cursor is placed on the display.

Comment: A user may choose to move the cursor back to correct earlier data items, and cannot be relied upon to move the cursor forward again. The computer should ignore cursor placement in such cases.

See also: 6.5-9.

-1 Keyed Entry of Quantified Direction

-1

When designation of direction (azimuth, bearing, heading, etc.) is based on already quantified data, then keyed entry should be used.

-2 Analog Entry of Estimated Direction

-2

When direction designation is based on graphic representation, then some "analog" means of entry should be provided, such as vector rotation on the display, and/or a suitably designed rotary switch.

Example: Heading estimation for displayed radar trails.

Exception: When approximate direction designation will suffice, for just eight cardinal points, keyed entry can be used.

Comment: In matching graphic display, an entry device providing a visual analog will prove both faster and more accurate.

Reference: Smith, 1962a.

(Guidelines for text entry are deferred pending further study.)

-1 Single Entry of Related Data

-1

In a form-filling dialogue, entering logically related items should be accomplished by a single, explicit action at the end, rather than by separate entry of each item.

Comment: This practice permits user review and possible data correction prior to entry, and also clarifies for the user just when grouped data are processed. It will also permit efficient cross validation of related data items by the computer.

Comment: Depending on form design, this practice might involve entering the entire form, or entry by page or section of a longer form. Just where entry is required should be indicated to the user.

See also: 1.0-8., 6.3-9, 6.3-18.

-2 Flexible Data Entry

-2

When multiple data items are entered as a single transaction, as in form filling, the user should be allowed to RESTART, CANCEL, or BACKUP and change any item before taking a final ENTER action.

Reference: BB 6.9; MS 5.15.1.2.4, 5.15.3.9.4.1.

See also: 3.5-2, 6.0-6, 6.3-10.

-3 Minimal Use of Delimiters

-3

Whenever possible, multiple data items should be entered without the need for special separators or delimiters, either by keying into predefined entry fields or by including simple spaces between sequentially keyed items.

-4 Standard Delimiter Character

-4

When a field delimiter must be used for data entry, a standard character should be adopted for that purpose; slash (/) is recommended.

-5 Prompting Data Entry**-5**

For all dialogue types involving prompting, data entries should be prompted explicitly by displayed labels for data fields, and/or by associated user guidance messages.

Example: (Good) NAME: _ _ _ _ _

ORGANIZATION: _ _ / _ _

PHONE: _ _ _ - _ _ _ _

(Bad) NAME, ORGANIZATION AND PHONE

_ _ _ _ _
_ _ _ _ _
_ _ _ _ _

Reference: BB 6.1.6

See also: 4.0-14.

-6 Consistent Labeling of Data Fields**-6**

Field labels should consistently indicate what data items are to be entered.

Example: A field labeled NAME should always require name entry, and not sometimes require something different like elevation.

-7 Protected Field Labels**-7**

In ordinary use, field labels should be protected and transparent to keyboard control, so that the cursor skips over them when spacing or tabbing.

Reference: PR 3.3.2, 4.8.1.

See also: 1.1-22, 2.0-6, 6.2-5, 6.3-3.

-8 Delineation of Data Fields

-8

Special characters should be used to delineate each data field; a broken-line underscore is recommended.

Example: (Good) Enter account number: _ _ _ _ _

(Bad) Enter account number:

Comment: Such implicit prompts help reduce data entry errors by the user.

Reference: BB 6.2.1; EG 6.3, 6.3.1; PR 4.8.1; Savage, Habinek and Blackstad, 1982.

See also: 1.0-4, 4.4-11.

-9 Length of Data Fields

-9

Implicit prompting by field delineation should indicate a fixed or maximum acceptable length of the entry.

Example: (Good) Enter ID: _ _ _ _ _

(Bad) Enter ID (11 characters):

Comment: Prompting by delineation is more effective than simply telling the user how long an entry should be. Underscoring gives a direct visual cue as to the number of characters to be entered, and the user does not have to count them.

Comment: Similar implicit cues should be provided when data entry is prompted by auditory displays. Tone codes can be used to indicate the type and length of expected data entries (Smith and Goodwin, 1970).

Reference: BB 6.2.1; EG 6.3; PR 4.8.2.

See also: 4.4-11.

-10 Required and Optional Data Fields

-10

Field delineation cues should distinguish required from optional entries.

Example: A broken underscore might be used to indicate required entries, with a dotted underscore to indicate optional entries:

LICENSE NUMBER: _ _ _ _ _

MAKE:

YEAR/MODEL:

Reference: BB 6.6; PR 4.8.6.

See also: 4.4-11.

-11 Automatic Justification of Variable-Length Entries

-11

When item length is variable, the user should not have to justify an entry either right or left, and should not have to remove any unused underscores; computer processing should handle those details automatically.

Reference: BB 6.2.2; EG 6.3.2.

-12 Tabbing to Data Fields

-12

When multiple items (especially those of variable length) will be entered by a skilled touch typist, each data field should end with an extra (blank) character space; software should be designed to prevent keying into a blank space, and an auditory signal should be provided to alert the user when that happens.

Comment: This will permit consistent use of tab keying to move from one field to the next.

Reference: PR 4.9.1.

See also: 1.1-21.

-13 Consistent Labeling Format**-13**

When data fields are distributed across a display, a consistent format should be adopted for relating labels to delineated entry areas.

Example: The label might always be to the left of the field; or the label might always be immediately above and left-justified with the beginning of the field.

Comment: Such consistent practice will help the user distinguish labels from data in distributed displays.

See also: 4.0-10.

-14 Distinctive Labels**-14**

Labels for data fields should be distinctively worded, so that they will not be readily confused with data entries, labeled control options, guidance messages, or other displayed material.

See also: 4.0-11.

-15 Label Coding**-15**

When displayed data forms are crowded, auxiliary coding should be adopted to distinguish labels from data.

Example: A recommended practice is to display fixed, familiar labels in dim characters, with data entries bright.

Comment: For novice users, it may sometimes be helpful to have brighter labels, if that could be provided as a selectable option.

Reference: PR 3.3.2.

See also: 2.1.2-7, 4.0-11.

-16 Informative Labels

-16

In labeling data fields, only agreed terms, codes and/or abbreviations should be used.

Example:

(Good)

WEEK: MONTH: YEAR:

SOCIAL SECURITY NUMBER: - - - - - - - -

(Bad)

DATECODE: - - - -

SSAN: - - - - - -

Comment: Do not create new jargon; if in doubt, pretest all proposed wording with a sample of qualified users.

Reference: BB 6.1.5; PR 4.5.6.

See also: 2.1.1-18, 4.0-14.

-17 Label Punctuation as Entry Cue

-17

The label for each entry field should end with a special symbol, signifying that an entry may be made.

Example: A colon is recommended for this purpose, e.g.,

NAME: - - - - - -

Comment: A symbol should be chosen that can be reserved exclusively for prompting user entries, or else is rarely used for any other purpose.

Reference: BB 6.5.

See also: 4.4-11.

-18 Data Format Cueing in Labels

-18

Labels for data fields may incorporate additional cueing of data formats when that seems helpful.

Example: DATE (M/D/Y): / /

Example: DATE: / /
 M M D D Y Y

Reference: PR 4.8.9.

See also: 4.0-14.

-19 Labeling Units of Measurement

-19

When a measurement unit is consistently associated with a particular data field, it should be displayed as part of the fixed label rather than entered by the user.

Example: COST: \$

Example: SPEED (MPH):

Reference: PR 4.8.11.

See also: 4.0-14.

-20 Variable Units of Measurement

-20

When alternative measurement units are acceptable, then space should be provided in the data field for user entry of a unit designator.

Example: DISTANCE: (MI/KM)

Reference: PR 4.8.11.

See also: 4.0-14.

-21 Familiar Units of Measurement

-21

Data should be entered in units that are familiar to the user.

Example: (Good) SPEED LIMIT: ___ ___ miles per hour

FUEL USE: ___ ___ . ___ miles per gallon

(Bad) SPEED LIMIT: ___ ___ feet per second

FUEL USE: . ___ ___ gallons per minute

Comment: Data conversion, if necessary, should be handled by the computer.

Reference: BB 6.3.

See also: 4.0-17.

-22 Compatible Formats for Data Entry and Display

-22

The display format for data entry should be compatible with whatever format is used for display output, scanning and review of the same data; item labels and ordering should be preserved consistently from one display to the other.

Comment: When a display format optimized for data entry seems unsuited for data display, or vice versa, some compromise format should be designed taking into account the relative functional importance of data entry and data review in the user's task.

See also: 2.3-2, 2.5-1, 4.0-5.

-23 Format Compatible with Source Documents

-23

When data entry involves transcription from source documents, form-filling displays should match (or be compatible with) paper forms; in a question-and-answer dialogue, the sequence of entry should match the data sequence in source documents.

Comment: When paper forms are not optimal for data entry, consider revising the layout of the paper form.

Comment: When data entries must follow an arbitrary sequence of external information (e.g., keying telephoned reservation data), some form of command language dialogue should be used instead of form filling, to identify each item as it is entered so that the user does not have to remember and re-order items.

Reference: BB 2.8.9; PR 4.8.3, 4.8.5, 4.10.7.

See also: 2.3-2, 2.5-1, 4.0-5.

-24 Format Follows Logical Sequence

-24

If no source document or external information is involved, the ordering of multiple-item data entries should follow the logical sequence in which the user can be expected to think of them.

Comment: Alternatively, data entry can sometimes be made more efficient by placing all required fields before any optional fields.

Reference: BB 6.6; PR 4.8.5.

See also: 2.3-2.

-25 Automatic Cursor Positioning

-25

When a form for data entry is displayed, the cursor should be positioned automatically in the first entry field.

Exception: If a data form is regenerated following an entry error, the cursor should be positioned in the first field in which an error has been detected.

Reference: PR 4.9.1.

See also: 1.1-19, 4.4-12.

-1 Tables for Related Data Sets

-1

When sets of data items must be entered sequentially, in a repetitive series, a tabular format where data sets are keyed row by row should be used.

Exception: When the items in each data set will exceed the display capacity of a single row, tabular entry will usually not be desirable.

Comment: Row-by-row entry will facilitate comparison of related data items, and permit potential use of a DITTO key for easy duplication of repeated entries.

Reference: PR 4.8.4.

-2 Informative Labels

-2

Column headers and row labels should be worded informatively, so as to help guide data entry.

See also: 4.0-14.

-3 Distinctive Labels

-3

Column headers and row labels should be formatted distinctively, so as to distinguish them from data entries.

See also: 4.0-11.

-4 Consistent Labeling Format

-4

When tabular formats are used for data entry, column labels should be left-justified with the leftmost position beginning column entries.

Comment: This consistent practice will prove especially helpful when columns vary in width.

See also: 4.0-5.

-5 Automatic Justification of Table Entries -5

Justification of tabular data entries should be handled automatically by the computer; the user should not have to enter any leading blanks or other extraneous formatting characters.

Example: If a user enters "56" in a field four characters long, the system should not interpret "56 ____" as "5600".

Reference: BB 6.2.3.

See also: 1.0-24

-6 Justification of Numeric Entries -6

It should be possible for the user to make numeric entries (e.g., dollars and cents) as left-justified, but they should be automatically justified with respect to a fixed decimal point when a display of those data is subsequently regenerated for review by the user.

Reference: PR 4.8.10.

-7 Entry of Duplicative Data -7

For entry of tabular data, when vertical repetition of entries is frequent the user should be provided a DITTO key to speed entry of duplicative data.

-8 Row Scanning Cues -8

For dense tables, those with many row entries, some extra visual cue should be provided to guide the user accurately across columns.

Example: A blank line after every fifth row is recommended. Alternatively, adding dots between columns at every fifth row may suffice.

Comment: This practice is probably more critical for accurate data review and change than it is for initial data entry, but is desirable in the interest of compatible display formats.

See also: 2.1.3-9.

(Guidelines for graphic data entry are deferred pending further study.)

-1 Automatic Data Validation

-1

Software for automatic data validation should be incorporated to check any item whose entry and/or correct format or content is required for subsequent data processing.

Comment: Do not rely on the user always to make correct entries. When validity of data entries can be checked automatically, such computer aids will help improve accuracy of data entry.

Comment: Some data entries, of course, may not need checking, or may not be susceptible to computer checking, such as free text entries in a COMMENT field.

Reference: MS 5.15.1.2.2; PR 4.12.4.

See also: 6.3-17, 6.3-18.

-2 Deferral of Required Data Entry

-2

When required data entries have not been entered, but can be deferred, data validation software should signal that omission to the user, permitting either immediate or delayed entry of missing items.

Reference: PR 4.8.7.

-3 Explicit Deferral Action

-3

When entry of a required data item is deferred, the user should have to enter a special symbol in the data field to indicate that the item has been temporarily omitted rather than ignored.

Reference: PR 4.8.7, 4.12.2.

See also: 4.0-2.

-4 Sequential Validation of Repetitive Entries

-4

In a repetitive data entry task, data validation for one transaction should be completed, and the user allowed to correct errors, before another transaction can begin.

Comment: This is particularly important when the user is transcribing data from source documents, so that detected entry errors can be corrected while the relevant document is still at hand.

See also: 3.5-13, 6.3-11.

-5 Optional Item-by-Item Validation

-5

If item-by-item data validation within a multiple-entry transaction is provided, it should only be as a selectable option.

Comment: This capability will sometimes help a novice user, who may be uncertain about what requirements are imposed on each data item; but it may slow a skilled user if the computer processing delays next item entry.

Reference: EG 6.3.9, 7.1.

See also: 4.3-10.

-6 User Definition of Default Values

-6

When helpful default values for data entry cannot be predicted by USI designers, which is often the case, the user (or perhaps some authorized supervisor) should have a special transaction to define, change or remove default values for any data entry field.

-7 Display of Default Values

-7

On initiation of a data entry transaction, currently defined default values should be displayed automatically in their appropriate data fields.

Comment: The user should not be expected to remember them.

Comment: It may be helpful to mark or highlight default values in some way to distinguish them from new data entries.

See also: 4.4-7, 6.3-7.

-8 User Confirmation of Default Entries

-8

User acceptance of a displayed default value for entry should be accomplished by simple means, such as by a single confirming key action, or simply by tabbing past the default field.

Comment: Similar techniques should be used in tasks involving user review of previously entered data.

See also: 6.3-7.

-9 Temporary Replacement of Default Values

-9

A user should be able to replace any data entry default value with a different entry, without necessarily changing the default definition for subsequent transactions.

-1 Computer Generation of Routine Data

-1

A user should not be required to enter "bookkeeping" data that the computer could determine automatically.

Example: A user generally should not have to identify his work station to initiate a transaction, nor include other routine data such as transaction sequence codes.

Comment: Complicated data entry routines imposed in the interest of security may hinder the user in achieving effective task performance; other means of ensuring data security should be considered.

See also: 6.1-1, 6.3-16.

-2 Automatic Entry of Redundant Data

-2

A user should not be required to enter redundant data already accessible to the computer.

Example: The user should not have to enter both an item name and identification code when either one defines the other.

Exception: As needed for resolving ambiguous entries, for user training, or for security (e.g., user identification).

Comment: Verification of previously entered data is often better handled by review and confirmation rather than by re-entry.

Reference: EG 6.3.10.

See also: 6.3-16.

-3 User Review of Prior Entries

-3

Data entries made in one transaction should be retrieved by the computer when relevant to another transaction, and displayed for user review if appropriate.

Comment: The user should not have to enter such data again.

Reference: BB 6.4.2.

See also: 6.3-15.

-4 Automatic Computation of Derived Data

-4

Whenever needed, automatic computation of derived data should be provided, so that a user does not have to calculate and enter any number that can be derived from data already accessible to the computer.

See also: 6.3-16.

-5 Automatic Cross-File Updating

-5

Whenever needed, automatic cross-file updating should be provided, so that a user does not have to enter the same data twice.

Example: Assignment of aircraft to a mission should automatically indicate that commitment in squadron status files as well as in a mission assignment file.

See also: 6.3-16.

-1 Flexible Design for Data Entry

-1

When data entry requirements may change, which is often the case, some means should be provided for the user (or an authorized supervisor) to make necessary changes to data entry procedures, entry formats, data validation logic, and other associated data processing.

SECTION 2

DATA DISPLAY

Data display, i.e., some kind of output from a computer to its users, is needed for all data handling tasks. Data display is emphasized particularly in monitoring and control tasks. Included as data display may be hardcopy printouts as well as more mutable electronic displays. Also included are auxiliary displays and signaling devices, including voice output, which may alert the user to unusual conditions. Displays specifically intended to guide the user in his interaction with the system are discussed separately under the topic of sequence control (Section 3).

In general, it may be said that rather less is known about data display, and information assimilation by the user, than about data entry. In current information system design, display formatting is an art. Guidelines are surely needed.

Here again some general concepts deserve emphasis, including the importance of context and flexibility. Data displays must always be interpreted in the context of task requirements and user expectations. An early statement of the need for relevance in data display seems valid still:

When we examine the process of man-computer communication from the human point of view, it is useful to make explicit a distinction which might be described as contrasting "information" with "data." Used in this sense, information can be regarded as the answer to a question, whereas data are the raw materials from which information is extracted. A man's questions may be vague, such as, "What's going on here?" or "What should I do now?" Or they may be much more specific. But if the data presented to him are not relevant to some explicit or implicit question, they will be meaningless. . . .

What the computer can actually provide the man are displays of data. What information he is able to extract from those displays is indicated by his responses. How effectively the data are processed, organized, and arranged prior to presentation will determine how effectively he can and will extract the information he requires from his display. Too frequently these two terms data and information are confused, and the statement, "I need more information," is assumed to mean, "I want more symbols." The reason for the statement, usually, is that

the required information is not being extracted from the data. Unless the confusion between data and information is removed, attempts to increase information in a display are directed at obtaining more data, and the trouble is exaggerated rather than relieved.

(Smith, 1963b, pages 296-297)

Certainly this distinction between data and information should be familiar to psychologists, who must customarily distinguish between a physical stimulus (e.g., "intensity" of a light) and its perceived effect ("brightness"). The distinction is not familiar to system designers, however, although the issue itself is often addressed. In the following description of what has been called the "information explosion", notice how the terms data and information are used interchangeably, confounding an otherwise incisive (and lively) analysis:

The sum total of human knowledge changed very slowly prior to the relatively recent beginnings of scientific thought. But it has been estimated that by 1800 it was doubling every 50 years; by 1950, doubling every 10 years; and by 1970, doubling every 5 years. . . . This is a much greater growth rate than an exponential increase. In many fields, even one as old as medicine, more reports have been written in the last 20 years than in all prior human history. And now the use of the computer vastly multiplies the rate at which information can be generated. The weight of the drawings of a jet plane is greater than the weight of the plane. The computer files of current IBM customer orders contain more than 100 billion bits of information -- more than the information in a library of 50,000 books.

For man, this is a hostile environment. His mind could no more cope with this deluge of data, than his body could cope with outer space. He needs protection. The computer -- in part the cause of the problem -- is also the solution to the problem. The computer will insulate man from the raging torrents of information that are descending upon him.

The information of the computerized society will be gathered, indexed, and stored in vast data banks by the computers. When man needs a small item of information he will request it from the computers. The machines, to satisfy his need, will sometimes carry on a simple dialogue with him until he obtains the data he wants.

With the early computers, a manager would often have dumped on his desk an indigestible printout -- sometimes several hundred pages long. Now the manager is more likely to request information when he needs it, and receive data about a single item or situation on a screen or small printer.

It is as though man were surviving in the depths of this sea of information in a bathyscaphe. Life in the bathyscaphe is simple, protected as it is from the pressure of the vast quantities of data. Every now and then man peers through the windows of the bathyscaphe to obtain facts that are necessary for some purpose or other. The facts that he obtains at any one time are no more than his animal brain can handle. The information windows must be designed so that man, with his limited capabilities, can locate the data he wants and obtain simple answers to questions that may need complex processing.

(Martin, 1973, page 6)

Somehow a means must be found to provide and maintain context in data displays so that the user can find the information he needs for his job. Task analysis may point the way here, indicating what data are relevant to each stage of task performance. Design guidelines must emphasize the value of displaying no more data than the user needs, maintaining consistent display formats so that the user always knows where to look for different kinds of information, and using consistent labeling to help the user relate different data items, on any one display and from one display to another.

Detailed user information requirements will vary from time to time, however, and may not be completely predictable in advance, even from a careful task analysis. Here is where flexibility is needed, so that data displays can be tailored on-line to user needs. Such flexibility is sometimes provided through optional category selection, display offset and expansion features. If such options for display coverage are available, the user may be able to adjust his processing of data outputs in a way analogous to self pacing of data inputs.

In tasks where a user must both enter and retrieve data, which is often the case, the formatting of data displays should be compatible with the methods used for data entry. As an example, if data entry is accomplished via a form-filling dialogue, with a particular format for data fields, subsequent retrieval of that data set should produce an output display with the same format, especially if the user is expected to make changes to displayed

data, and/or additional entries. Where compaction of data output is required for greater efficiency, to review multiple data sets in a single display frame, the displayed items should retain at least an ordering and labeling compatible with those fields used previously for data entry.

Display design should also be compatible with dialogue types used for sequence control, and with hardware capabilities. Where user inputs are made via menu selection, using a pointing device like a lightpen, then display formats should give prominence (and adequate separation) to the labeled, lightpennable options. Location of multi-function keys at the display margin, to be labeled on the adjacent portion of the display itself, may provide flexibility for both data entry and sequence control, but will necessarily constrain the formatting of displays for data output.

These general concepts underlie many of the specific guidelines for data display that are proposed in the following pages.

DATA DISPLAY

Objectives: Consistency of data display
Efficient information assimilation by the user
Minimal memory load on user
Compatibility of data display with data entry
Flexibility for user control of data display

DATA DISPLAY

General 2.0

-1 Only Necessary Data Displayed

-1

Displayed data should be tailored to user needs, providing only necessary and immediately usable information at any step in a transaction sequence.

Comment: When user needs cannot be exactly anticipated by the designer, provision should be made for on-line tailoring of displays by the user, including data selection, display coverage and display suppression.

Reference: EG 3.1.4, 3.3.1; MS 5.15.2.3.

See also: 4.0-3.

-2 Data Displayed in Usable Form

-2

Data should be displayed to the user in directly usable form.

Example: (Too cryptic) Error 459 in column 64.

(Better) Error 459, character in NAME entry cannot be recognized.

Comment: The user should not be required to transpose, compute, interpolate, translate displayed data into other units, or refer to documentation to determine the meaning of displayed data.

Reference: BB 3.3; EG 3.3.4; MS 5.15.2.8, 5.15.4.9.

See also: 4.4-1.

-3 Consistent Data Display

-3

Data should be displayed consistently, following standards and conventions familiar to the user.

Example: If users work with metric units of measurement, do not display data in English units, or vice versa.

Example: Computer time records that are not in directly usable format should be converted for display, to a conventional 12-hour (AM/PM) clock or a 24-hour clock, in local time or whatever other time standard is appropriate to user needs.

Comment: Adopt a consistent standard when no specific user convention has been established.

Reference: BB 3.4; EG 2.2.4.

See also: 4.0-16.

-4 User Control of Data Display

-4

An experienced user should be provided means to control the amount, format, and complexity of displayed data.

Reference: EG 3.4.2.

-5 User Changes to Displayed Data

-5

Data displays should permit direct user change (replacement) of displayed items, or entry of new items, when that represents an efficient transaction sequence.

Comment: Some consistent formatting cue, perhaps different initial cursor placement, should be provided to inform the user when displayed data can or cannot be changed.

Reference: PR 4.4.

See also: 6.2-3.

-6 Protection of Displayed Data**-6**

When protection of displayed data is essential, maintain computer control over the display and do not permit a user to change controlled items.

Comment: Never assume compliance with instructions by the user, who may attempt unwanted changes by mistake, or for curiosity, or to subvert the system.

Reference: EG 3.4.8.

See also: 1.1-22, 1.4-7, 6.2-4. 6.3-3.

-7 Remembering Displayed Data**-7**

If data must be remembered from one display to another, no more than 4-6 items should be displayed.

Comment: Better yet, do not require the user to rely on memory, but recapitulate needed items on the succeeding display.

Reference: EG 2.3.15.

-1 Consistent Display Format

-1

When textual material is formatted, as in structured messages, headers, etc., consistent format should be maintained from one display to another.

Comment: A missing item should be shown as a labeled blank, rather than being omitted from a standard format.

-2 Mixed Case for Displayed Text

-2

Running text (prose) should be displayed conventionally, in mixed upper and lower case.

Exception: An item intended to attract the user's attention, such as a label or title, may be displayed in upper case.

Exception: Upper case should be used when lower case letters will have decreased legibility, which is true on display terminals that cannot show descenders for lower case letters.

Comment: Normal reading of text is easier with conventional use of capitalization, i.e., to start sentences, indicate proper nouns and acronyms, etc.

Reference: BB 2.6; EG 3.4.3.

-3 Left Justify Displayed Text

-3

Displayed text should be left justified to maintain constant spacing between words, leaving right margins ragged if that is the result.

Exception: When right justification can be achieved by variable spacing, maintaining constant proportional differences in spacing between and within words, and consistent spacing between words in a line.

Comment: Reading is easier with constant spacing, which outweighs the debatable esthetic advantage of an even right margin achieved at the cost of uneven spacing. Uneven spacing is a greater problem with narrow column formats than with wide columns. Uneven spacing handicaps poor readers more than good readers.

Reference: PR 4.5.1, 4.10.5; Gregory and Poulton, 1970; Campbell, Marchetti and Mewhort, 1981.

-4 Minimal Hyphenation of Displayed Text

-4

In textual material, words should be displayed intact wherever possible, with minimal breaking by hyphenation between lines.

Comment: Text is more readable if the entire word is on one line, even if that makes the right margin more ragged.

Reference: BB 3.2; EG 2.2.10; MS 5.15.4.9.g.

-5 Separation of Paragraphs

-5

Displayed paragraphs of text should be separated by at least one blank line.

Reference: EG 2.3.4.

-6 Sentences End with Period

-6

In textual display, every sentence should end with a period.

Reference: EG 2.2.13.

-7 Sentences Begin with Main Topic -7

In textual display, the main topic of each sentence should be placed near the beginning of the sentence.

Reference: BB 3.8.2.

-8 Simple Sentence Structure -8

In textual display, short, simple sentences should be used.

Comment: Long sentences with multiple clauses may confuse the user. Consider breaking long sentences into two or more shorter statements.

Reference: BB 3.8, 3.8.1; EG 2.2.12; Wright and Reid, 1973.

See also: 4.3-5.

-9 Concise Wording of Displayed Text -9

When speed of display output for textual material is slower than the user's normal reading speed, an extra effort should be made to word the text concisely.

Comment: Assume a normal average reading speed of 250 words per minute.

Reference: EG 3.3.7.

See also: 4.3-5.

-10 Affirmative Sentences

-10

In textual display, affirmative statements should be used rather than negative statements.

Example: (Good) Clear the screen before entering data.

(Bad) Do not enter data before clearing the screen.

Comment: Tell the user what to do, rather than what to avoid.

Reference: BB 3.8.3.

See also: 4.0-18.

-11 Active Voice

-11

In textual display, sentences in the active voice should be used rather than passive voice.

Example: (Good) Clear the screen by pressing RESET.

(Bad) The screen is cleared by pressing RESET.

Comment: Active voice sentences are generally easier to understand.

Reference: BB 3.8.5.

See also: 4.0-19

-12 Temporal Sequence

-12

In textual display, sentences describing a sequence of events should be phrased with a corresponding word order.

Example: (Good) Enter LOGON before running programs.

(Bad) Before running programs, enter LOGON.

Comment: Temporal order is clearer. Reverse order may confuse the user.

Reference: BB 3.8.6.

See also: 4.0-20.

-13 List Format

-13

When listing textual material, or other data, each item should start on a new line, i.e., the list should be a single column.

Example: (Good) Major USI functional areas include
Data Entry
Data Display
Sequence Control
User Guidance
Data Transmission
Data Protection

(Bad) Major USI functional areas include Data
Entry, Data Display, Sequence Control, User
Guidance, Data Transmission, and Data
Protection

Exception: Multiple columns of data should be used where that facilitates comparison of corresponding data sets, as in tabular displays.

Exception: Listing in multiple columns may be considered where shortage of display space dictates a compact format.

Comment: Single-column list format will facilitate rapid, accurate scanning.

Reference: BB 2.3.2, 2.9.2; EG 2.3.5.

-14 Logical List Ordering

-14

Lists within text should be ordered by some logical principle; long lists, with more than seven items, should be ordered alphabetically.

Reference: EG 2.3.1.

See also: 2.3-6.

-15 List Ordering in Multiple Columns

-15

If a list is displayed in multiple columns, item ordering should be vertical within each column, considered sequentially from left to right.

-16 Listed Items Grouped by Difficulty -16

For material that must be remembered in a displayed statement or list, the hardest items should be put at the beginning and the easiest items in the middle.

Reference: EG 3.3.3, 3.3.5.

-17 Immediate Action Items Listed Last -17

Material that need be recalled only for the next immediate user action should be put at the end of a statement or list.

Reference: EG 3.3.5.

-18 Familiar Terminology -18

In text displays and labels, word usage should incorporate familiar terms and the technical jargon of the user, and avoid the unfamiliar jargon of the interface designer and programmer.

Comment: When in doubt, pretest the meaning of words for prospective users, to ensure that there is no ambiguity.

Reference: BB 2.2.2, 3.7.1, 3.7.2, 3.7.4; EG 3.4.5, 4.2.13; PR 4.5.6.

See also: 1.4-16, 4.0-16, 4.0-17, 4.3-3.

-19 Consistent Wording -19

In text displays and labels, word usage should be consistent, particularly for technical terms.

Example: The word "screen" should not be used to mean "display frame" in one place, and "menu selection option" in another.

Comment: Standard terminology should be defined and documented for reference by interface designers and by users.

Reference: BB 2.2.2; EG 3.4.5, 4.2.13; MS 5.15.1.3.1.

See also: 4.0-17.

-20 Consistent Wording Across Displays

-20

Wording of text and labels should be consistent from one display to another.

Example: The title of a display should be identical to the menu option used to request the display.

Reference: BB 3.7.3.

See also: 4.0-16.

-21 Contractions

-21

In text display and labels, use distinct words rather than contractions or combined forms, especially in phrases involving negation.

Example: (Good) "will not", "not complete"

(Bad) "won't", "incomplete"

Comment: This practice will help the user understand the sense of the message.

Reference: BB 3.1.4; EG 2.2.15.

-22 Minimal Abbreviation

-22

In text display and labels, complete words should be used in preference to abbreviations.

Exception: Abbreviations may be displayed if they are significantly shorter, save needed space, and will be understood by the prospective users.

Exception: When abbreviations are required (or useful) for data entry, then corresponding use of those abbreviations in data display may help a user learn them for data entry.

Reference: BB 3.1.1; EG 4.1.3; MS 5.15.1.3.2.

-23 Consistent Abbreviation

-23

In text display and labels, when words are abbreviated, the designer should ensure that abbreviations are consistent in form, and that abbreviations of different words are distinguishable.

Reference: BB 3.1, 3.1.2; EG 4.1.3; MS 5.15.1.3.3;
PR 4.5.6.

See also: 1.0-14 thru 1.0-18.

24 Special Abbreviations Marked

-24

If an abbreviation deviates from the consistent form, it should be specially marked whenever it is displayed.

Reference: Moses and Ehrenreich, 1981.

-25 Dictionary of Abbreviations

-25

When abbreviations are used, a dictionary of abbreviations should be available for on-line user reference.

Reference: BB 3.1.3.

See also: 4.4-16.

-26 Punctuation of Abbreviations

-26

In text display and labels, abbreviations and acronyms should not include punctuation.

Example: (Good) "USAF"

(Bad) "U.S.A.F."

Exception: Punctuation should be retained when needed for clarity, e.g., "4-in. front dimension" rather than "4 in front dimension".

Reference: BB 2.3.4; EG 2.2.14.

-1 Labeling Data Fields

-1

When data items are displayed in a distributed format, each field should have an associated label to identify it.

Comment: Do not assume that the user can identify individual data fields because of past familiarity. Context may play a significant role: 617-271-7768 might be recognized as a telephone number if seen in a telephone directory, but might not be recognized as such in an unlabeled display.

Reference: BB 2.8.7; EG 2.2.16; MS 5.15.4.9.i.

See also: 4.0-14.

-2 Wording of Data Field Labels

-2

A data field label should be a descriptive title, phrase or word, consistently positioned adjacent to (above, or to the left of) a displayed item, or group of items.

Comment: Labels should be worded carefully to assist a new user in scanning the display and assimilating information quickly.

Comment: Labels may be worded as a heading or title reflecting the question for which an answer is sought in the data that follow.

Reference: BB 2.9.1; EG 3.2, 3.2.4; MS 5.15.2.10.

See also: 2.1.2-8.

-3 Distinctive Wording of Labels

-3

Field labels should be distinctive from one another in wording, to aid user discrimination.

Reference: BB 3.5; EG 3.2.3; MS 5.15.2.10.c.

-4 Consistent Construction for Labels

-4

Consistent grammatical construction should be used for labels, and for items in a list.

Example: Do not use single words or phrases for some items and short sentences for others.

Reference: BB 3.8.4.

See also: 4.0-21.

-5 Distinctive Format for Labels

-5

Labels should be distinctive in format/positioning to help distinguish them from displayed data and other types of displayed material (e.g., error messages).

Reference: EG 3.2.3; MS 5.15.2.10.a.

See also: 4.0-11.

-6 Spacing of Labels

-6

Labels and their associated data fields should be separated by at least one space in the display.

Reference: BB 2.9.5; EG 2.3.8.

-7 Optional Highlighting for Labels

-7

An option should be provided to highlight labels for a new user, or to dim labels for an experienced user.

Reference: EG 3.2; MS 5.15.2.10.b.

See also: 1.4-22.

-8 Consistent Formatting of Labels

-8

In data forms, labels and data fields should be consistently formatted, and aligned to minimize search time by the user.

Example: In a numbered list, vertically formatted, the data items should start in a fixed column position on the display.

Reference: EG 2.3.7, 2.3.9.

See also: Section 2.1.3.

-9 Labeling Units of Measurement

-9

The units of measurement for displayed data should be included either in the label or as part of each data item.

Reference: BB 2.8.8.

See also: 2.1.3-2.

-10 Consistent Format Across Displays

-10

The ordering and layout of corresponding data fields should be consistent from one display to another.

Reference: BB 2.8.3.

See also: 2.1.3-11.

-11 Consistent Format Within Data Fields

-11

The detailed internal format of frequently used data fields should be consistent from one display to another.

Example: Telephone numbers should be consistently hyphenated, as 213-394-1811.

Example: Time records might be consistently formatted with colons, as HH:MM:SS, or HH:MM, or MM:SS.S, whatever is appropriate.

Example: Date records might be consistently formatted with slashes, as MM/DD/YY.

Comment: The convention chosen should be that familiar to the prospective users. For European users, the formatting of telephone numbers and of dates is customarily different than suggested in the examples above. For military users, date/time data are frequently combined in a familiar special format. For many user groups, time records are kept on a 24-hour clock, which should be acknowledged in display formatting.

Reference: EG 2.2.17; MS 5.15.1.3.4.

-12 Partitioning Long Data Items

-12

Long data items of arbitrary alphanumeric characters should be displayed in groups of three or four separated by a blank.

Exception: Words should, of course, be displayed intact, whatever their length.

Comment: Hyphens may be used instead of blanks where that is customary. Slashes are less preferred for separating groups, since they are more easily confused with alphanumerics.

Comment: Grouping should follow convention where a common usage has been established, as in the NNN-NN-NNNN of social security numbers.

Reference: BB 2.4.1; EG 2.2.2; MS 5.15.4.9.a.

See also: 1.0-11.

-1 Labeling Tables

-1

In tabular displays, columns and rows should be labeled following the same guidelines proposed for labeling the fields of data forms.

Reference: BB 2.8.7.

See also: Section 2.1.2.

-2 Labeling Units of Measurement

-2

In tabular displays, the units of displayed data should be consistently included in the column labels, or following the first row entry.

Example: (Good)	<u>Time</u>	<u>Velocity</u>	<u>Temperature</u>
	(s)	(m/s)	(°C)
	5	8	25
	21	49	29
	43	87	35

(Also acceptable)	<u>Time</u>	<u>Velocity</u>	<u>Temperature</u>
	5 s	8 m/s	25 °C
	21	49	29
	43	87	35

Reference: BB 2.8.8.

See also: 2.1.2-9.

-3 Justification of Numeric Data

-3

Columns of numeric data should be displayed right-justified, or justified with respect to a fixed decimal point.

Reference: BB 2.4.2, 2.4.3; EG 2.3.9; MS 5.15.4.9.d;
PR 4.8.10, 4.10.6.

See also: 1.5-6.

-4 Justification of Alphabetic Listings

-4

Lists of alphabetic data should be vertically aligned with left justification to permit rapid scanning; indentation can be used to indicate subordinate elements in hierarchic lists.

Example:	(Good)	APL	(Bad)	APL
		COBOL		COBOL
		FORTRAN		FORTRAN
		PL1		PL1

Exception: Numbers should be right-justified.

Exception: A short list, of just 4-5 items may be displayed horizontally on a single line, in the interests of compact display format, if that is done consistently.

Reference: BB 2.3.1; EG 2.2.8, 2.2.11; MS 5.15.4.9.d, 5.15.4.9.e.

-5 List Organization

-5

Data lists should be organized in some recognizable order, whenever feasible, to facilitate scanning and assimilation.

Example: Dates may be ordered chronologically, names alphabetically.

Reference: EG 2.2.3, 2.3.1; MS 5.15.4.9.b.

See also: 3.1.3-13.

-6 Labeling Listed Items

-6

When listed data are labeled by number, letter, etc., the list format should be distinctive from lists of menu options.

Reference: EG 2.2.7.

See also: 3.1.3-12.

-7 Numbering Listed Items

-7

When listed items are labeled by number, the numbering should start with "1", and not "0".

Comment: In counting, people start with "one"; in measuring, start with "zero".

Reference: EG 2.2.6.

-8 Hierarchic Numbering

-8

For hierarchic lists with compound numbers, the complete numbers should be used, rather than omitting the repeated elements.

Example: (Good)

- 2.1 Position Designation
 - 2.1.1 arbitrary positions
 - 2.1.1.1 discrete
 - 2.1.1.2 continuous
 - 2.1.2 predefined positions
 - 2.1.2.1 HOME
 - 2.1.2.2 other

(Bad)

- 2.1. Position Designation
 - 1. arbitrary positions
 - 1 discrete
 - 2 continuous
 - 2. predefined positions
 - 1 HOME
 - 2 other

Comment: Although implicit numbering, as in the "bad" example, may be acceptable for tasks involving perception of list structure, complete numbering will be better for tasks requiring search and identification of individual items in the list.

Reference: Smith and Aucella, 1982.

-9 Row Spacing

-9

In dense tables with many rows, a blank line (or some other distinctive feature) should be inserted after every fifth row as an aid for horizontal scanning.

See also: 1.5-8.

- 10 Column Spacing -10
- When data are displayed in more than one column, the columns should be separated by at least 3-4 spaces if right-justified, and by at least 5 spaces otherwise.
- Reference: EG 2.3.6.
- 11 Consistent Column Spacing Across Displays -11
- Column spacing should be consistent from one display to another.
- Reference: BB 2.8.3.
- See also: 2.1.2-10
- 12 Column Assignment in Reference Tables -12
- When tables are used for referencing purposes, such as an index, the indexed material should be displayed in the left column, the material most relevant for user response in the next adjacent column, and associated but less significant material in columns further to the right.
- Reference: Hamill, 1980.
- 13 Table Format for Item Comparison -13
- Items that must be compared on a character-by-character basis should be displayed with one directly above the other.
- Reference: MS 5.15.4.6.2.d.

-1 Graphic Display for Data Comparison

-1

When users must scan and compare sets of data quickly, items should be displayed in an ordered graphic format, with backup display of raw data available as a user-selected option.

Comment: People cannot readily assimilate and compare detailed sets of raw data.

Reference: EG 2.2.9; MS 5.15.4.9.f.

-2 Graphic Displays for Monitoring

-2

Graphic displays should be used, rather than alphanumeric, when a user must monitor changing data in any critical task involving qualitative distinction between normal and abnormal conditions.

Comment: It is preferable, of course, to program the computer to handle data monitoring, where that is feasible, and to signal detected abnormalities to the user's attention.

Reference: Hanson, Payne, Shiveley and Kantowitz, 1981; Tullis, 1981.

-3 Standardized Graphics Symbology

-3

Graphic symbols should be standardized in meaning within a system, and among systems having similar operational requirements.

Reference: MS 5.15.2.6.

-1 Display of Mixed Data Types

-1

When tables and/or graphics are combined with text, each figure should be placed immediately following its first reference in the text.

Comment: People may not look at a figure if it is displayed in a location separated from its reference.

Reference: Whalley and Fleming, 1975.

-1 Display Identification Label

-1

When the user participates in selection of data for display, each display should have a unique identifying label, an alphanumeric code or abbreviation that can facilitate display requests by the user.

Comment: The display identification label should be short enough (3-7 characters) or meaningful enough to be remembered easily. Where flexibility is desired, it may be good practice to let each user assign names to the particular sets of data that constitute commonly used displays.

Reference: BB 2.2.3.

See also: 4.2-4.

-2 Consistent Position for Display Label

-2

The identifying label used for display selection should be displayed prominently in a consistent location.

Comment: The top left corner of the display is recommended for this purpose.

Reference: BB 2.2.3.

See also: 2.5-1, 4.0-5, 4.2-4.

-1 Consistent Data Grouping

-1

Grouped data should be arranged in the display with consistent placement of items, so that user detection of similarities, differences, trends and relationships is facilitated.

Example:

<u>Cost</u>			<u>Output</u>		
<u>Actual</u>	<u>Predicted</u>	<u>Difference</u>	<u>Actual</u>	<u>Predicted</u>	<u>Difference</u>
947	901	+ 46	83	82	+ 1
721	777	- 56	57	54	+ 3
475	471	+ 4	91	95	- 4

Reference: BB 2.8.6; Tullis, 1981.

-2 Data Grouped by Sequence of Use

-2

Displayed data should be grouped by some logical principle, such as sequence of use, where the spatial (or temporal) order is that in which data items are usually encountered.

Example: Data in an electronic display should match the order of items in an associated paper data form.

Reference: BB 2.8.1; PR 4.10.7.

See also: 1.4-23, 1.4-24, 1.4-25.

-3 Data Grouped by Function

-3

Displayed data should be grouped by some logical principle, such as by function, where data items associated with a particular question or purpose are located adjacent to one another in the display.

Reference: BB 2.8.1; Tullis, 1981.

-4 Data Grouped by Importance

-4

Displayed data should be grouped by some logical principle, such as importance, where data items providing the most significant information, and/or requiring immediate response, are grouped at the top of the display.

Reference: BB 2.8.1; Tullis, 1981.

-5 Data Grouped by Frequency

-5

Displayed data should be grouped by some logical principle, such as frequency, where the most frequently used data items are presented at the top of the display.

Comment: Principles of data grouping also apply to the display/listing of control options.

Comment: These principles for data grouping in display formatting are essentially the same as those recommended for display/control layout in equipment design.

Reference: BB 2.8.1.

See also: 3.1.3-13.

-6 Data Grouped Alphabetically or Chronologically

-6

When there is no appropriate logic for grouping data by sequence, function, frequency or importance, some other principle should be adopted, such as alphabetical or chronological grouping.

Reference: BB 2.8.2.

See also: 2.1.1-14.

-1 Response Time to Display Request

-1

System response to simple requests for data display should take no more than 0.5-1.0 second.

Example: This response time should apply when the user requests the next page of a multi-page display, or when a display begins to move in response to a scrolling request.

Comment: Responses to requests for new displays may take somewhat longer, perhaps 2-10 seconds, particularly if the user perceives such a request to involve more complicated operations, such as accessing different files, transforming data, etc.

Reference: F; Table 2.

See also: 4.2-2.

-2 Printing Displays Locally

-2

When displayed data are of potential long-term interest, there should be an easy means for the user to request local printing of a hard copy, within security restraints.

Comment: USI design should not require the user to rely on memory. Optional printout permits the user to record data from one display to compare with another, and so deal with situations where the system designer has not anticipated the need for such comparison.

Comment: The user should not have to take notes or transcribe displayed data manually. That practice under-utilizes the data handling potential of the computer, and risks transcription errors by the user.

Reference: BB 1.7; EG 4.2.14; MS 5.15.4.8; PR 4.10.1.

See also: 6.2-6, 6.4-5.

-3 Passive Printing

-3

The contents of a display should not change as a result of a user request for printout.

Reference: EG 4.2.14; MS 5.15.4.8.

-1 Consistent Display Format

-1

A consistent organization for the location of various display features should be adopted as a standard format to be used, insofar as possible, for all displays.

Example: One location might be used consistently for a display title, another area might be reserved for data output by the computer, and other areas dedicated to display of control options, instructions, error messages, and user command entry.

Comment: Consistent display formats are needed to establish and preserve user orientation. There is no fixed display format that is optimum for all data handling applications, which will vary in their requirements. However, once a suitable format has been devised, it should be maintained as a pattern to ensure consistent design of other displays.

Reference: BB 2.1, 2.8.4; EG 2.3, 2.3.3; MS 5.15.4.6.1.d.

See also: 4.0-5.

-2 Display Pages

-2

When a display output contains too much data for presentation in a single screen, the data should be partitioned automatically into separately displayable pages, each page with appropriate labeling to show its relation to the others.

Comment: Convenient control procedures should also be provided to let a user move easily from one page to another.

See also: 2.8-6.

-3 Clarity of Display Format

-3

Means should be devised to make established display formats clearly perceptible to the user.

Example: Different display areas, or "windows", can be separated by spacing (where space permits); outlining can also be used to separate different areas, so that displayed data are distinct from control options, instructions, etc.

Reference: BB 2.8.5; EG 2.3; MS 5.15.4.6.2.

See also: 4.0-11.

-4 Window Size

-4

When a display window must be used for data scanning, the window size should be greater than one line.

Reference: Elkerton, Williges, Pittman and Roach, 1982.

-5 Distinctive Display Formats

-5

Established display formats should be changed only as necessary to distinguish one task or activity from another.

Comment: The objective is to develop display formats that are consistent with accepted usage and existing user habits.

Reference: EG 2.2.5.

See also: 4.0-5.

-6 Integrated Data Display

-6

The body of the display, when used for data output, should be formatted to present data coherently, and usually should not be partitioned into many small windows.

Reference: EG 2.3.2.

-7 Display Titles

-7

Every display should begin with a title or header, describing briefly the contents or purpose of the display; the title should be separated by one blank line from the body of the display.

Reference: BB 2.1.1, Table 1; PR 4.5.2.

See also: 4.2-4.

-8 Command Entry at Bottom of Display

-8

The last several lines at the bottom of every display should be reserved for status and error messages, prompts and command entry.

Comment: Assuming that the display is mounted physically above the keyboard, which is the customary placement, the user can look back and forth from keyboard to display more easily when prompts and entry area are at the bottom of the display.

Reference: BB 6.1.2; PR 4.5.3; Granda, Teitelbaum and Dunlap, 1982.

See also: 3.1.5-2, 4.0-10.

-1 All Necessary Information Displayed

-1

Ideally, each display should provide the user all of the information needed at that point in the transaction sequence.

Example: Header information should be retained, or re-generated, when paging /scrolling data tables.

Comment: The user should not have to remember information from one display to the next.

Reference: BB 4.3.4; EG 2.3.15.

See also: 2.0-1, 2.8-1, 4.0-3, 4.4-1.

-2 Only Relevant Information Displayed

-2

Ideally, each display should provide the user only the information essential at that point in the transaction sequence, and not be overloaded with extraneous data.

Example: (Good) Select data to be displayed: _ _

CODE	DATA TYPE
su	Summary
d	Detailed list
se	Sequences

(Bad) Select data to be displayed: _ _

CODE	DATA TYPE	DATE IMPLEMENTED
su	Summary	5-17-82
d	Detailed list	7-14-82
se	Sequences	9-25-82

Comment: Extraneous data will prevent or slow user assimilation of needed information. Where user information requirements cannot be accurately determined in advance of interface design, and/or are expected to be variable, on-line user options should be provided for data selection, display coverage and suppression.

Reference: BB 2.7, 2.8.10; EG 3.1.4; MS 5.15.2.3; Tullis, 1981.

See also: 2.0-1, 4.0-3.

-3 Display One Item per Line

-3

For simple user-system dialogues, each line of a display should provide a single item of information.

Exception: Form filling.

Reference: PR 4.10.5.

-1 Highlighting Critical Data

-1

Important data items requiring user attention should be highlighted on the display with some form of auxiliary coding, particularly when they appear infrequently.

Example: Such items might include recently changed data, or discrepant data exceeding acceptable limits, or data failing to meet some other defined criteria.

Comment: Position coding might suffice, i.e., displaying important items consistently in a particular location, as when an error message appears in a space otherwise left blank. But auxiliary codes may still be needed to highlight important items, even if they are positioned consistently.

Reference: EG 2.1.3, 2.3.12; MS 5.15.4.6.1.

See also: 4.0-15.

-2 Display Coding by Data Category

-2

Display coding should be used in applications where the user must distinguish rapidly among different categories of displayed items, particularly when those items are distributed in an irregular way on the display.

-3 Alphanumeric Coding

-3

Alphanumeric characters can be used in effectively unlimited combinations, and should be considered for auxiliary coding in display applications where basic data presentation is not already alphanumeric (e.g., graphics).

Reference: EG Table 1.

-4 Consistent Case in Alphabetic Coding

-4

When using alphabetic codes, a consistent convention should be adopted that all letters shall either be upper case or else lower case.

Comment: For data display, upper case labels will be somewhat more legible. For data entry, computer logic should not distinguish between upper and lower case codes, because the user will find it hard to remember any such distinction.

Reference: BB 2.3.3.

-5 Alphanumeric Grouping

-5

When codes combine letters and numbers, characters of each type should be grouped together rather than interspersed.

Example: Letter-letter-number ("HW5") will be read and remembered more accurately than letter-number-letter ("H5W").

Comment: Unfortunately, there are common instances in which this recommendation is ignored, such as the coding of English and Canadian postal zones.

Reference: BB 2.5.1.

-6 Meaningful Codes

-6

Meaningful codes should be adopted in preference to arbitrary codes.

Example: A three-letter mnemonic code (DIR = directory) is easier to remember than a three-digit numeric code.

Reference: BB 3.6.2.

-7 Familiar Coding Conventions

-7

Display (and data entry) codes should be assigned so as to conform with conventional population stereotypes, accepted abbreviations, and general user expectations.

Example: Use M for "male", F for "female", rather than arbitrary digits 1 and 2. In color coding, use red for danger.

Reference: BB 2.3.5.

See also: 2.7-28, 4.0-7.

-8 Definition of Display Codes

-8

When codes are assigned special meaning in a display, a definition should be provided at the bottom of the display.

Example: The legend on a map is a common example.

Comment: The definition should replicate the code, i.e., display the symbol, line width, etc., being defined. For a color code, each definition should be displayed in the appropriate color, e.g., "RED = hostile" in red.

Reference: BB 7.6.1.

See also: 4.4-17.

-9 Code Length

-9

When arbitrary codes must be remembered by the user, they should be no longer than 4-5 characters.

Comment: When a code is meaningful, such as a mnemonic abbreviation or a word, it can be longer.

Reference: BB 2.5.2.

-10 Consistent Coding Across Displays

-10

Symbols, and other codes as well, should be assigned to have consistent meanings from one display to another.

Comment: When coding is not consistent, the user's task of display interpretation may be made more difficult than if no auxiliary coding were used at all.

Reference: BB 3.6.1, 7.6.2.

See also: 2.1.1-24, 4.0-8.

-11 Special Symbols

-11

Special symbols, such as asterisks, arrows, etc., should be considered for drawing attention to selected items in alphanumeric displays.

Comment: Symbols chosen for such an "alerting" purpose should not be used for other purposes in the display.

See also: 4.3-17.

-12 Spacing of Marker Symbols

-12

When a special symbol is used to mark a word, it should be separated from the beginning of the word by a space.

Comment: A symbol immediately adjacent to the beginning of a word will impair legibility.

Reference: Noyes, 1980.

-13 Shape Coding

-13

Geometric shapes should be considered for discriminating different categories of data on graphic displays.

Comment: Approximately 15 different shapes can be distinguished readily. If that "alphabet" is too small, it may be possible to use component shapes in combination, as in some military symbol codes.

Reference: EG Table 1.

-14 Consistent Shape Coding

-14

When shape coding is used, the assignment of codes should be consistent for all displays, and based upon an established standard.

Comment: Although shape codes can often be mnemonic in form, their interpretation will generally rely on learned association as well as immediate perception. Existing user standards must be taken into account by the display designer.

Reference: MS 5.15.4.6.1.e.

See also: 4.0-8.

-15 Coding by Line Length

-15

A special form of shape coding, using lines of varying length, should be considered for applications involving spatial categorization in a single dimension.

Example: The length of a displayed vector might be used to indicate distance or speed.

Comment: Perhaps four lengths can be reliably distinguished in practical use. Long lines will add clutter to a display, but may be useful in special applications.

Reference: EG Table 1.

-16 Coding by Line Direction

-16

A special form of shape coding, using lines of varying direction, should be considered for applications involving spatial categorization in two dimensions.

Example: The angle of a displayed vector might be used to indicate direction, i.e., heading or bearing.

Comment: Users can make fairly accurate estimates of angles for lines displayed at ten-degree intervals.

Reference: Smith, 1962a.

-17 Line Coding

-17

Auxiliary methods of line coding should be considered for graphics applications, including variation in line type (e.g., solid, dashed, dotted) and line width ("boldness").

Comment: Perhaps 3-4 line types might be readily distinguished, and 2-3 line widths.

Reference: EG 2.3.

-18 Line Positioning

-18

When a line is added simply to mark or emphasize a displayed item, it should be placed under the designated item.

Comment: A consistent convention is needed to prevent ambiguity in the coding of vertically arrayed items; underlining is customary, and does not detract from word legibility.

Comment: For words from the Roman alphabet, underlining probably detracts from legibility less than overlining.

-19 Size Coding

-19

Size coding, i.e., varying the size of displayed alphanumeric and other symbols, should be considered only for applications where displays are not crowded.

Comment: Perhaps as many as five sizes might be used for data categorization, but two or three will probably prove the practical limit except for printed displays.

Reference: EG Table 1; MS 5.15.4.6.1.e.

-20 Differences in Size Codes

-20

When size coding is used, a larger symbol should be at least 1.5 times the height of the next smaller symbol.

Reference: MS 5.15.4.6.1.e.

-21 Brightness Coding

-21

Differences in brightness of displayed symbols should be considered for many display applications as a two-valued code.

Example: A data form might combine bright data items with dim labels to facilitate display scanning.

Comment: Perhaps as many as four brightness levels might be used, but at some risk of reduced legibility for the dimmer items.

Reference: EG 2.1.4, Table 1; MS 5.15.4.6.1.b.

-22 Brightness Inversion

-22

When a capability for brightness inversion is available, i.e., where bright characters on a dark background can be changed under computer program control to dark on light (or vice versa), this should be considered for use in highlighting displayed items that require user attention.

Reference: PR 3.3.4.

-23 Color Coding

-23

Color coding should be considered for applications where the user must distinguish rapidly among several categories of data, particularly when data items are dispersed on the display.

Example: Different colors might be used effectively in a position display to distinguish friendly, unknown and hostile aircraft tracks, or alternatively to distinguish among aircraft in different altitude zones.

Comment: Color is a good auxiliary code, where a multi-color display capability is available. A color code can be overlaid directly on alphanumerics and other symbols without significantly obscuring them. Color coding permits rapid scanning and perception of patterns and relationships among dispersed data items.

Comment: Perhaps as many as 11 different colors might be reliably distinguished, or even more for trained observers; but as a practical matter it will prove safer to use no more than five or six.

Reference: BB 7.2; EG Table 1; MS 5.15.4.6.1.f; Smith, 1963a; Smith and Thomas, 1964; Smith, Farquhar and Thomas, 1965.

-24 Conservative Use of Color

-24

Color coding should be used conservatively, with relatively few colors to designate critical categories of displayed data.

Comment: Arbitrary use of many colors may cause displays to appear "busy" or cluttered, and may reduce the likelihood that relevant color coding on other displays will be interpreted appropriately and quickly by the user.

Reference: BB 7.1.

-25 Color Coding Formatted Displays

-25

Color coding should be applied as an additional aid to the user on displays that have already been formatted as effectively as possible in a single color.

Comment: Do not use color coding in an attempt to compensate for poor display format; redesign the display instead.

Reference: BB 7.3.

-26 Redundant Color Coding

-26

When color coding is used, it should be redundant with some other feature in data display, such as symbology.

Comment: Displayed data should provide necessary information even when viewed at a monochromatic display terminal, or hard-copy printout, or when viewed by a user with defective color vision.

Reference: BB 7.4.

-27 Assignment of Color Codes

-27

When color coding is used, each color should represent only one category of displayed data.

Comment: Color will prove the dominant coding dimension on a display. If several different types of data are displayed in red, say, they will have an unwanted visual coherence that may hinder proper assimilation of information by the user.

Reference: BB 7.6.1; Smith and Thomas, 1964.

-28 Conventional Color Coding

-28

Color coding should be consistent with conventional associations with particular colors.

Example: In a display of accounting data, negative numbers might be shown as red, corresponding to the customary use of red ink for that purpose.

Example: Red is associated with danger (in our society), and is an appropriate color for alarm conditions. Yellow is associated with caution, and might be used for alerting messages or to denote changed data. Green is associated with normal "go ahead" conditions, and might be used for routine data display. White is a color with neutral association, which might be used for general highlighting.

Comment: Other associations can be learned by the user if color coding is applied consistently.

Reference: BB 7.7.1, 7.7.2, 7.7.3; MS 5.15.4.6.1.f.

See also: 2.7-7, 4.0-7, 4.0-8, 4.3-17.

-29 Limited Use of Blue

-29

The color blue should be used only for background features in a display, and not for critical data.

Example: Blue might be used in shading background areas in graphic displays, where its lower apparent brightness could possibly be of benefit.

Comment: The human eye is not equally sensitive to all colors, nor are its optics color-corrected. Blue symbols appear dimmer than others, and are more difficult to focus.

Reference: BB 7.6, 7.7.5.

-30 Blink Coding

-30

Blink coding should be considered for applications where a displayed item implies an urgent need for user attention.

Comment: Blinking symbols are effective, if used sparingly, in calling the user's attention to displayed items of unusual significance. Blinking characters may have somewhat reduced legibility, and may cause visual fatigue with over-use.

Comment: Perhaps as many as four levels might be reliably distinguished, but it will probably prove safer to use blinking as a two-level code, i.e., blinking versus non-blinking.

Reference: EG Table 1; MS 5.15.4.6.1.a; Smith and Goodwin, 1971b; Smith and Goodwin, 1972.

See also: 2.7-31, 4.3-17.

-31 Blinking Marker Symbols

-31

When blink coding is used to mark a data item that must be read, an extra symbol such as an asterisk should be added as a blinking marker, rather than blinking the item itself.

Comment: This practice will draw attention to an item without detracting from its legibility.

Reference: Smith and Goodwin, 1971b.

See also: 2.7-30.

-32 Blink Rate

-32

When blink coding is used, the blink rate should be in the range of 2-3 Hz, with a minimum duty cycle (ON interval) of 50 percent.

Comment: Although equal ON and OFF intervals are often specified, an effective code can probably be provided even when the OFF interval is considerably shorter than the ON (perhaps a wink, rather than a blink), as in occulting lights used for Navy signaling.

-33 Coding with Texture, Focus, Motion

-33

Visual dimensions that should be considered for special display coding applications include variation in texture, focus and motion.

Comment: Texture can be useful for area coding in graphic displays. Only two levels of focus are feasible, clear and blurred, with the risk that blurred items will be illegible. Perhaps 2-10 degrees of motion might be distinguished, in display applications where motion is an appropriate and feasible means of coding.

Reference: EG 2.3.

-34 Distinctive Auditory Coding

-34

For auditory displays, distinctive sounds should be used to code items requiring special user attention.

Example: A variety of signals might be available, including sirens, bells, chimes, buzzers, and tones of different frequency.

Comment: Tones may be presented in sequence to enlarge the signal repertoire.

Reference: Smith and Goodwin, 1970.

See also: 4.3-17.

-35 Voice Coding

-35

For auditory displays with voice output, different voices should be considered for use in distinguishing different categories of data.

Comment: At least two voices, male and female, could be readily distinguished, and perhaps more depending upon fidelity of auditory output, and listening conditions.

Reference: Smith and Goodwin, 1970.

-1 Integrated Display

-1

Whenever possible, all data relevant to the user's current transaction should be included in one display page (or "frame").

Comment: Do not rely on the user to remember data accurately from one display to the next.

Reference: EG 3.4.4.

See also: 2.0-1, 2.6-1, 4.0-3, 4.4-1.

-2 Paging/Scrolling

-2

When requested data exceeds the capacity of a single display frame, the user should be provided easy means to move back and forth among relevant displays, by paging or scrolling.

Example: Dedicated function keys might be provided for paging/scrolling forward and back.

Comment: Paging/scrolling is acceptable when the user is looking for a specific data item, but not when the user must discern some relationship among separately displayed sets of data.

Reference: BB 4.4.1, 4.4.2; EG 6.3.8; MS 5.15.4.9.j.

-3 Paging/Scrolling Continued Lists

-3

When a list of numbered items exceeds one display page, and must be paged/scrolling for its continuation, items should be numbered continuously in relation to the first item in the first display.

Reference: EG 2.3.10.

-4 Paging/Scrolling Continued Tables

-4

When a tabular display must be paged/scrolling for its continuation, column headings and row labels should be preserved in each viewed portion of the display.

-5 Annotating Continued Displayed Data

-5

When lists or tables are of variable length, and may extend beyond the limits of a single display page, their continuation and ending should be explicitly noted on the display.

Example: Incomplete lists might be marked "continued on next page", or simply "continued". Concluding lists might add a note "end of list".

Exception: Short lists whose conclusion is evident from the display format need not be annotated in this way.

Reference: BB 2.9.6.

See also: 4.2-5.

-6 Display Page Numbering

-6

When display output contains more than one page, the notation "page x of y" should appear on each display.

Comment: A recommended format is to put this note immediately to the right of the display title. With such a consistent location, the page note might be displayed in dimmer characters. Leading zeros should not be used in the display of page numbers.

Reference: PR 4.5.5, 4.10.4.

See also: 2.5-2, 4.2-5.

-7 Consistent Scrolling or Windowing

-7

When scrolling is used, a consistent orientation should be adopted in USI design as to whether 1) data are conceived to move behind a fixed display window, commonly called "scrolling"; or 2) the display window is conceived to move over a fixed array of data, here called "windowing".

Comment: A user can adapt to either concept, if it is maintained consistently. "Windowing" is the more natural concept for inexperienced users, causing fewer errors, and hence is the preferred option when other considerations are equal.

Reference: BB 4.4.8; Bury, Boyle, Evey and Neal, 1982.

-8 Windowing with Free Cursor Movement

-8

In applications where a cursor is moved freely within a page of displayed data, "windowing" should be selected rather than "scrolling" as the conceptual basis of display movement.

Example: Full-screen editing.

Comment: Since displayed data will be perceived as fixed during cursor movement, considerations of joint compatibility suggest that displayed data remain conceptually fixed during window "movement". Indeed, it may be possible to use the same arrow-labeled keys to control both cursor movement and "windowing".

Reference: Morrill and Davies, 1961.

-9 Consistent Windowing Orientation

-9

When a "windowing" orientation is maintained consistently, the wording of scroll functions should refer to the display page (or window) and not to the displayed data.

Example: The command "Up 10" should mean that ten lines of data will disappear from the bottom of the display, and ten earlier lines will appear at the top.

-10 Consistent Scrolling Orientation

-10

When a "scrolling" orientation is maintained consistently, the wording of scroll functions should refer to the data being displayed, and not to the display page or window.

Example: "Roll up 5 lines" should mean that the top five lines of data will disappear from the display, and five new lines will appear at the bottom.

Reference: EG 2.3.16.

-11 Wording of Scrolling or Windowing Functions

-11

When the user may be exposed to different systems adopting different usage, any reference to scroll functions should avoid wording that implies spatial orientation (e.g., "up", "down"), and instead consistently use functional terms such as "forward" and "back" (or "next" and "previous") to refer to movement within a displayed data set.

Comment: In that event, control of scroll functions should be implemented by keys marked with arrows, avoiding verbal labels altogether.

-1 Automatic Display Update

-1

In accord with operational requirements, the user should be able to request automatic update (computer regeneration) of displayed data, and control the update rate.

-2 Display Rate for Changing Data

-2

Changing data values that the user must read accurately should be updated only in a fixed position on the display, and no faster than one per second.

Reference: MS 5.15.4.5.1.

-3 Display Rate for Changing Data

-3

Changing data values that the user need read only approximately to identify the general nature of data change should be updated no faster than five per second.

Reference: MS 5.15.4.5.2.

-4 Time Compression and Expansion

-4

Graphic displays in which a user must visually integrate changing patterns should be updated at a rate matching the user's data handling abilities.

Comment: Slowly developing patterns may be seen more easily with time compression, i.e., with rapid (and repetitive) display of sequentially stored data frames. Fast changing data may require time expansion, i.e., slowed motion, to detect patterns.

Comment: Similar considerations may apply to auditory displays, where speeding or slowing sound signals may aid pattern recognition.

Reference: MS 5.15.4.5.3.

-5 Display Freeze**-5**

When a display is automatically updated, the user should be able to stop the process ("freeze", "stop action") at any point, to examine changed data more deliberately.

Comment: For some applications, it may also prove helpful if the user can step incrementally forward or back in the time sequence, frame by frame.

Reference: MS 5.15.4.5.4.

-6 Annotating Display Freeze**-6**

When an updated display is frozen, some appropriate label should be added to remind the user of that status.

Reference: MS 5.15.4.5.5.

See also: 4.4-10.

-7 Updating Display Freeze**-7**

When a display being updated in real time is frozen, the user should be warned if some significant (but not displayed) change is detected in the computer processing of new data.

Reference: MS 5.15.4.5.4.

See also: 4.4-10.

-8 Ending Display Freeze**-8**

When a display being updated in real time is frozen, resumption of display update should be at the current real-time point unless otherwise specified by the user.

Comment: In some applications, a user might wish to resume display update at the point of stoppage, and so display change would thenceforth lag real-time data change. As a general practice, however, such an option risks confusion.

Reference: MS 5.15.4.5.4.

-1 **Temporary Suppression of Displayed Data****-1**

When standard data displays are used for special purposes, the user should be provided means of temporarily suppressing the display of data not needed for the current task.

Comment: Data selections made originally for one purpose may not be appropriate for another. When task requirements shift rapidly, it may be more efficient to suppress temporarily the display of unneeded data categories, rather than to re-generate a display with different selection criteria.

See also: 2.0-1.

-2 **Restoring Display of Suppressed Data****-2**

When data can be temporarily suppressed from display, the user should be provided means for rapid restoration of the display to its complete, originally selected form.

Comment: In some applications, it may be desirable to make restoration of suppressed data automatic, after expiration of a predetermined time-out, rather than relying on the user to remember to do it.

-1 **Flexible Design for Data Display****-1**

When data display requirements may change, which is often the case, some means should be provided for the user (or an authorized supervisor) to make necessary changes to display functions.

Comment: Display characteristics that may need to be changed include those represented in these guidelines, namely, the types of data that are displayed, the selection and aggregation of displayed data, the formats for display partitioning, plus changes in display density, coding, coverage, update and suppression logic.

SECTION 3

SEQUENCE CONTROL

Sequence control refers to the logic and means by which inputs and outputs are linked to become coherent transactions, and which govern the transitions from one transaction to the next. General design objectives are consistency of control actions, minimized control and minimized memory load on the user, with flexibility of sequence control to adapt to different user needs. Techniques of sequence control require explicit attention in USI design, and a number of published guidelines bear on this topic.

The importance of good design for controlling user interaction with a system is emphasized by Brown, Burkleo, Mangelsdorf, Olsen and Williams:

One of the critical determinants of user satisfaction and acceptance of a computer system is the extent to which the user feels in control of an interactive session. If the user cannot control the direction and pace of the interaction sequence, he is likely to feel frustrated, intimidated, or threatened by the computer system. His performance may suffer or he may avoid using the system at all.

(1981, page 4-1)

Complete user control of the interaction sequence and its pacing is not always possible, of course, particularly in applications using computer aids for monitoring and process control. The actions of an air traffic controller, for example, are necessarily paced in some degree by the job to be done. As a general principle, however, it is the user who should decide what needs doing and when to do it.

A fundamental decision in USI design is selection of the dialogue type(s) that will be used to implement sequence control. Here dialogue refers to the sequence of transactions which mediate user-system interaction. USI design will often involve a mixture of two or more dialogue types, since different dialogues are appropriate to different jobs and different kinds of users. Recognition of appropriate dialogue types at the outset of system development will facilitate USI design and help ensure the effectiveness of future system operation.

The selection of dialogue types based on anticipated task requirements and user skills seems straightforward, at least for

simple cases. Computer-initiated question-and-answer dialogues are suited to routine data entry tasks, where data items are known and their ordering can be constrained, and provides explicit prompting for unskilled, occasional users. Form-filling dialogues permit somewhat greater flexibility in data entry, but may require user training. When data entries must be made in arbitrary order, perhaps mixed with queries as in making airline reservations, for example, then some mixture of function keys and coded command language will be required for effective operation, implying a moderate to high level of user training.

One important aspect of dialogue choice is that different types of dialogue imply differences in system response time for effective operation. In a repetitive form-filling dialogue, for example, the user may accept relatively slow computer processing of a completed form. If the computer should take several seconds to respond, the user probably can take that time to set one data sheet aside and ready another. But several seconds delay in a menu selection dialogue may prove intolerable, especially where the user must make an extended sequence of selections in order to complete an action.

To categorize these differences, an estimate of the implied requirement for user training and for system response time is given below for eight general dialogue types. Cumulative experience and specific requirements of a particular task may modify such estimates. But the general principle illustrated here, that one design choice implies others, must be taken into account in USI specification.

<u>Dialogue Type</u>	<u>Required User Training</u>	<u>Required System Response Time</u>
Question and Answer	Little/None	Moderate
Form Filling	Moderate/Little	Slow
Menu Selection	Little/None	Very Fast
Function Keys	High/Moderate	Fast
Command Language	High	Fast
Query Language	High/Moderate	Moderate
Natural Language	Moderate (potentially little)	Fast
Interactive Graphics	High	Very Fast

Flexibility and context are important in sequence control as in other aspects of USI design. Ideal flexibility would permit the user to undertake whatever task or transaction is needed, at any time. Although this may not always prove feasible, the USI designer should try to provide the maximum possible user control of the on-line transaction sequence. As a simple example, a user who is scanning a multi-page data display should be able to go either forward or back at will. If the USI design only permits stepping forward, so that the user must cycle through the entire display set to reach a previous page, that design is inefficient. The user should also be able to interrupt display scanning at any point to initiate some other transaction. Such simple flexibility is relatively easy for the designer to achieve, and indeed is commonly provided.

More difficult are transactions that involve potential change to stored data. Here again the user will need flexibility in sequence control, perhaps wishing to back up in a data entry sequence to change previous items, or to cancel and restart the sequence, or to abort the sequence altogether and escape to some other task. The USI designer can provide such flexibility through use of suspense files and other special programmed features. This flexibility requires extra effort from the designer and programmer. But that extra effort is made only once, and is a worthwhile investment on behalf of future users who may interact with their computer system for months or even years.

In one respect, flexibility of sequence control has pitfalls. Just as users can make mistakes in data entry, so also will users make mistakes in sequence control. The USI designer must try to anticipate user errors and ensure that potentially irreversible actions are difficult to take. In most data entry tasks, for example, simple keying of data items should not in itself initiate computer processing. The user should have to take some further, explicit action to ENTER the data. The USI should be designed to protect the user from the consequences of inadvertently destructive actions. Any large-scale erasure or deletion of data, for example, should require some sort of explicit user confirmation, being accomplished as a two-step process rather than a single transaction. (This provides a software analogy to the physical barriers sometimes used to protect critical hardware controls from accidental activation.) Some well-designed systems go a step further and permit the user to reverse a mistaken action already taken.

One form of flexibility frequently recommended is the provision of alternate modes of sequence control for experienced and inexperienced users. In a command-language dialogue, optional guidance might be provided to prompt a beginner step by step in: the

composition of commands, whereas an experienced user might enter a complete command as a single complex input. Some such flexibility in USI design is surely desirable -- to interpret halting, stepwise control inputs, as well as fluent, coherent commands.

More generally, however, it may be desirable to include redundant modes of sequence control in USI design, perhaps involving combinations of different dialogue types. As an example, menu selection might be incorporated to provide easy sequence control for beginners, but every display frame might also be formatted to include a standard field where an experienced user could enter complete commands more efficiently. Examples of this approach have been provided by Palme (1979).

Another way to provide flexibility in sequence control is through specific tailoring of display formats. Consider, for example, a dialogue type in which sequence control is exercised through lightpen selection among displayed command options. For any particular display frame it might be possible to display just three or four options most likely to be selected by a user at that point in the task sequence, plus a general purpose OPTIONS selection that could be used to call out a display of other (less likely) commands. Thus, on the first page of a two-page display set, one of the likely commands would be NEXT PAGE; but on the second page that command would be replaced by its more likely complement, PREV PAGE.

This approach illustrates two design ideas. The first comes close to being a general rule for sequence control: make the user's most frequent transactions the easiest to accomplish. The second idea is the reliance on context to improve flexibility. These general ideas concerning sequence control are reflected in the specific design guidelines proposed in the following pages.

SEQUENCE CONTROL

Objectives: Consistency of control actions
Minimal control actions by user
Minimal memory load on user
Compatibility with user needs
Flexibility of sequence control

SEQUENCE CONTROL

General 3.0

-1 Flexible Sequence Control

-1

Flexible means of sequence control should be provided so that the user can accomplish necessary transactions involving data entry, processing, retrieval and transmission, or can obtain guidance as needed in connection with any transaction.

Example: In scanning a multi-page display the user should be able to go forward or back at will; if USI design permits only forward steps, so that the user must cycle through the entire display series to reach a previous page, that design is deficient.

Comment: Necessary transactions should be defined in task analysis prior to software design.

Reference: PR 4.0.

-2 Minimal User Actions

-2

Control entries should be simplified to the maximum extent possible, particularly for real-time tasks requiring fast user action, and should permit completion of a transaction sequence with the minimum number of control entries consistent with user abilities.

Example: The user should be able to print a display directly without having to take a series of other actions first, such as calling for the display to be filed, specifying a file name, then calling for a print of that named file.

Example: For long, multi-page displays, it should be possible to request a particular page directly, without having to take repetitive PAGE FORWARD actions.

Comment: Shortcuts via direct commands should be provided for experienced users, to by-pass intervening steps that might provide a more easily learned sequence for beginners. The computer should be programmed to handle intervening steps automatically, informing the user what has been done if that seems necessary.

Reference: BB 4.4.1, 4.5; MS 5.15.2.7.

See also: 4.0-22.

-3 Ease/Difficulty of User Control

-3

The ease of sequence control should match desired ends; frequent or urgent actions should be easy to take, whereas potentially destructive actions should be made sufficiently difficult to require explicit user attention.

Comment: Perhaps "difficult" is the wrong word here. If a destructive action is made different or distinctive in some way, so that it will not be taken by mistake, then perhaps it is not necessary that it be made difficult as well.

Reference: EG 4.0, 4.1.2.

See also: 3.5-9, 3.6-4, 4.3-16, 4.3-17, 6.5-5.

-4 Control Matched to User Skill

-4

Sequence control should be compatible with user skills, permitting simple step-by-step actions for beginners, and efficiently coded commands for experienced users.

Comment: This will generally require a mix of dialogue types.

See also: 4.4-26, and Section 3.1.

-5 Control by Explicit User Action

-5

In most on-line information handling systems, sequence control should result from explicit user entries rather than occur as an automatic consequence of computer processing.

Example: The computer should not interrupt user data entry to require immediate correction of any entry error, but instead should wait until the user signals completion of the transaction.

Exception: Routine, repetitive transactions in which successful completion of one may lead automatically to initiation of the next.

Exception: Automated process control applications where emergency conditions may take precedence over current user transactions.

Comment: In general, computer detection of problems with current user entries can be negotiated at the conclusion of a transaction, before it is implemented. Computer detection of other problems can be signaled by alarms or advisory messages so that the user can choose when to deal with them.

See also: 1.0-8, 1.1-5, 1.4-1, 4.0-2, 6.0-3, 6.3-8.

-6 User Initiative in Sequence Control**-6**

Although the user-system dialogue is necessarily limited by the computer, software design should permit initiative and control by the user; the USI designer should anticipate all possible user actions and their consequences, and should provide appropriate options in every case.

Comment: In particular, a dialogue should never reach a dead end with no further action available to the user; if the user makes an entry inappropriate (or unrecognizable) to current processing logic, the result should simply be an advisory message indicating the nature of the problem and the available options as to what can be done next.

Reference: BB 4.2; PR 2.2.

See also: 4.4-5.

-7 User-Paced Sequence Control**-7**

Whenever possible, control entries should be self-paced, depending upon the user's needs, attention span and time available, rather than computer processing or external events.

Comment: When user-pacing does not seem feasible, the general approach to task allocation and USI design should be reconsidered.

See also: 1.0-6.

-8 Computer Response Time**-8**

The speed of computer response to user entries should be appropriate to the transaction involved; in general, the response should be faster for those transactions perceived by the user to be simple.

Example: Computer response to a predictable control entry, such as NEXT PAGE, should be within 0.5-1.0 second; response to other simple entries should be within 2.0 second; error messages should be displayed within 2-4 second.

Reference: Miller, 1968.

See also: 4.2-2.

-9 Computer Processing Delay

-9

Control entries by a user should not be delayed or paced by delays in computer response.

Comment: It is recommended that control delays or lockouts not exceed 0.2 seconds. In some applications, however, longer delay may be tolerable, particularly if that has the effect of reducing variability in computer response time.

Reference: MS 5.15.3.3.

See also: 1.0-7.

-10 Keyboard Locked during Delay

-10

If further user entries must be delayed pending completion of computer processing, the keyboard should be automatically locked until the user can begin a new transaction; keyboard lock should be accompanied by disappearance of the cursor from the display and (especially if infrequent) by some more specific indicator such as an auditory signal.

Comment: Absence of a cursor is not in itself a sufficient indicator of keyboard lockout. Auditory signals will be particularly helpful to skilled touch typists, who may not look at the display during a repetitive data transcription.

Comment: Following keyboard lockout, computer readiness to accept further entries should be signaled to the user.

Comment: In some cases, it may be desirable to provide the user with an auxiliary means of control entry, such as a special function key, to abort a transaction causing extended keyboard lockout.

See also: 3.3-6, 4.1-4.

-11 Signaling Completion of Delayed Processing

-11

When execution of a control entry is delayed, the computer should give the user some positive indication when processing is subsequently completed, the outcome, and the implied need for further user actions if any.

Reference: BB 4.3.1; MS 5.15.1.4.c.

See also: 4.2-3.

-12 Feedback for Control Entries

-12

All control entries made by a user should be acknowledged unambiguously by the computer, either by their immediate execution, or else by some immediate message indicating that execution is in progress or deferred or that the control entry requires correction or confirmation.

Example: In particular, the absence of computer response is not an acceptable means of indicating that a control entry is being processed.

Comment: "Immediate" as used here is subject to interpretation in relation to the response time requirements of different dialogue types.

Reference: BB 4.3.2; EG 4.2.5; MS 5.15.3.2, 5.15.3.4.

See also: 4.2-1, 4.2-7, and Section 3.1.

-13 Feedback in Natural Form

-13

Feedback for control entries should generally consist of changes in state or value of displayed elements affected by the control action, in an expected or natural form.

Reference: MS 5.15.3.4.1.

See also: 4.2-1.

-14 Feedback for Data Entry

-14

In data entry tasks accomplished as a single, discrete transaction, successful entry should be signaled by a confirmation message without any other display change.

Comment: This follows the general recommendation for sequence control that the user should leave one transaction and choose the next by explicit action.

Reference: MS 5.15.1.2.7.d.

See also: 4.2-1.

-15 Feedback for Repetitive Data Entry

-15

In repetitive data entry tasks, accomplished in a continuing sequence of transactions, successful entry should be signaled by regeneration of the data entry display, automatically removing the just entered data in preparation for the next entry.

Comment: This represents an exception to the general principle of sequence control by explicit user choice, in the interest of efficiency.

Comment: An explicit message confirming successful data entry can be added in cases where that seems helpful.

Reference: EG 4.2.10.

See also: 4.2-1.

-16 Consistent User Actions

-16

Sequence control actions should be consistent in form and consequences throughout USI design; similar means should be employed to accomplish similar ends, from one transaction to the next, and from one task to another.

Comment: In particular, there should be some standard, consistent routine for the user to initiate and complete task sequences.

See also: 4.0-1.

-17 Displayed Context

-17

If the consequences of a given control entry will differ depending upon context established by a prior action, then some appropriate means of context definition should be displayed in advance to the user.

Comment: Do not rely on the user always to remember prior actions, nor to understand their current implications.

See also: 4.4-10.

-18 Logical Transaction Sequences

-18

The design of linked transaction sequences should be based on task analysis, i.e., should represent a logical unit or subtask from the viewpoint of the user.

Comment: A logical unit to the user is not necessarily the same as a logical unit of the computer software that mediates the transaction sequence.

Reference: PR 5.1.

See also: 4.0-1.

-19 Distinctive Display for Sequence Control

-19

Displays should be designed so that features relevant to sequence control are distinctive in position and/or format.

Comment: Relevant features include displayed options, command entry areas, prompts, advisory messages, and other displayed items (titles, time signals, etc.) whose changes signal the results of control entries.

See also: 4.0-5.

-20 Simultaneous Users

-20

When two or more users must interact with the system simultaneously, control entries by one should not interfere with those of another.

Comment: This requires careful USI design for applications where joint, coordinated actions must be made by a group of users.

Reference: MS 5.15.2.5.

See also: 6.5-2.

-21 Consistent Terminology for Sequence Control

-21

In instructional material, and in on-line messages to the user, consistent terminology should be adopted to refer to control entries.

Example: Various words and phrases might be used, such as "control input", "command entry", "instruction", "request", "function call", etc. The practice adopted in these guidelines is to call general sequence control actions "control entry". More specific terminology is sometimes used here, such as "command entry" for keyed control entries composed by the user, "code entry" for keyed selections from displayed menus, etc.

See also: 4.0-6.

-1 Dialogue Matched to User and Task

-1

Choice of dialogue type(s) and design of sequence control dialogue should take into account user characteristics and task requirements.

Example: When data entries must be made in arbitrary order, perhaps mixed with queries (as in making flight reservations), then some mixture of function keys and coded command entries will be required for effective operation, implying a moderately high level of user training.

Comment: The simple dictum is, "Know the user." If user characteristics are variable, which is often the case, then a variety of dialogue types should be provided.

See also: 3.0-4, 4.4-26.

-2 Speed of Computer Response

-2

The speed of computer response to user entries should be appropriate to the type of dialogue; in general, the response to menu selections, function keys, and most entries during graphic interaction should be immediate.

Comment: It is generally thought that maximum acceptable computer response for menu selection by lightpen is 1.0 second; for key activation is 0.1 second; for cursor positioning by lightpen (as in graphic line drawing) 0.1 second.

Comment: If computer response time will be slow, other dialogue types should be considered by the USI designer.

Reference: Miller, 1968.

See also: 4.2-2.

-1 Question-and-Answer Dialogue -1

Question-and-answer dialogue should be considered primarily for routine data entry tasks, where data items are known and their ordering can be constrained, where the user will have little or no training, and where computer response is expected to be moderately fast.

Comment: Brief question-and-answer sequences can be used to supplement other dialogue types for special purposes, such as for log-on routines, or for resolving ambiguous control or data entries.

-1 Form-Filling Dialogue -1

Form-filling dialogue should be considered when some flexibility in data entry is needed, such as the inclusion of optional as well as required items, where users will have moderate training, and/or where computer response may be slow.

See also: Section 1.4.

-1 Menu Selection

-1

Menu selection should be considered for tasks such as scheduling and monitoring that involve little entry of arbitrary data, where users may have relatively little training, and where computer response is expected to be fast.

Comment: Menu selection is, of course, a generally good means of mediating control entries by untrained users, in conjunction with other dialogue types for other task requirements.

Comment: When display output is slow, as for a printing terminal, or for an electronic display constrained by a low-bandwidth channel, it may be tiresome for a user to wait for display of menu options.

-2 Single Selection per Menu

-2

Each menu display should require just one selection by the user.

Comment: Novice users will be confused by any more complicated procedure, such as a "Chinese menu" requiring one choice from Column A, two from Column B, etc.

Reference: PR 4.6.5.

-3 Menu Selection by Pointing

-3

When menu selection is the primary means of sequence control, and especially if extensive lists of control options must be displayed, then selection should be accomplished by direct pointing (e.g., by lightpen).

See also: 1.1-13.

-4 Dual Activation for Pointing

-4

If menu selection is handled by pointing, dual activation should be provided, the first action to designate (position a cursor on) the selected option, followed by a separate action to make an explicit control entry.

Comment: The two actions should be compatible in their design implementation. If the cursor is positioned by keying, then an ENTER key should be used to signal control entry. If the cursor is positioned by lightpen, then a dual-action "trigger" on the lightpen should be provided for positioning and control entries.

Comment: This recommendation assumes that accuracy in selection of control entries is more important than speed. In some applications that may not be true. USI design will involve a trade-off considering the criticality of wrong entries, ease of recovery from wrong entries, and user convenience in making selections.

See also: 1.0-8, 3.0-5, 6.0-3.

-5 Menu Selection by Keyed Entry

-5

When menu selection is a secondary (occasional) means of control entry, and/or only short option lists are needed, then selection may be accomplished by keyed entry of corresponding codes, or by other means such as programmed multi-function keys labeled in the display margin.

-6 Standard Area for Code Entry

-6

When menu selection is accomplished by code, that code should be keyed into a standard command entry area (window) in a fixed location on all displays.

Example: In a customary terminal configuration, with the display located above the keyboard, command entry should be in the bottom line of the display.

Comment: In effect, the command entry area should be positioned to minimize user head/eye movement between the display and the keyboard.

Comment: For experienced users, coded menu selections can be keyed in a standard area identified only by its consistent location and use; if the system is designed primarily for novice users, that entry area should be given an appropriate label such as ENTER CHOICE HERE: ____.

Reference: MS 5.15.4.6.1.d; PR 4.6.3.

See also: 4.0-5.

-7 Wording of Menu Options

-7

Menu options should be worded so as to permit direct selection by pointing or by code entry, rather than worded as questions.

Example: For option selection by pointing,

(Good) +PRINT

(Bad) PRINT?

Example: For option selection by code entry,

(Good) p = Print

(Bad) Print? (Y/N)

Reference: PR 4.6.8.

See also: 3.1.3-12, 4.0-21.

-8 Explicit Option Display

-8

When control entries will be selected from a discrete set of options, then those options should be displayed at the time of selection.

Reference: MS 5.15.3.5.

See also: 3.0-6, 4.4-5.

-9 Options Worded as Commands

-9

If menu selection is used in conjunction with (as an alternative to) command language, then displayed options should be worded in terms of recognized commands or command elements.

Comment: Where appropriate, sequences of menu selections should be displayed in an accumulator until the user signals entry of a completely composed command.

Comment: This practice will speed the transition for a novice user, relying initially on sequential menu selection, to become an experienced user composing coherent commands without such aid.

See also: 4.0-6.

-10 Letter Codes for Menu Selection**-10**

If menu selections must be made by keyed codes, then each code should be the initial letter (or letters) of the displayed option label, rather than an arbitrary number.

Example: (Good) Enter sex (M/F):

(Bad) Enter sex:

1 Male
2 Female

Exception: Options might be numbered when a logical order or sequence is implied.

Exception: When menu selection is from a long list, line number might be an acceptable alternative to letter codes.

Comment: Letters are easier than numbers for touch typists; options can be reordered on a menu without changing letter codes; it is easier to memorize meaningful names than numbers, and so letter codes can facilitate a potential transition from menu selection to command language when those two dialogue types are used together.

Comment: USI designers should not create unnatural option labels just to ensure that the initial letter of each will be different. There must be some natural difference among option names, and a two- or three-letter code can probably be devised to emphasize that difference.

Reference: BB 2.3.6; Palms, 1979. (Note contrary views, favoring number codes, in BB 2.9.3; EG 2.2.7; PR 4.6.2.)

See also: 4.0-8.

-11 Consistent Coding of Menu Options

-11

If letter codes are used in menu selection, then insofar as possible those codes should be used consistently in designating options at different steps in a transaction sequence.

Example: The same action should not be given different names and hence different codes (F = FORWARD and N = NEXT); the same code should not be given to different actions (Q = QUIT and Q = QUEUE).

See also: 4.0-6.

-12 Distinctive Coding of Menu Options

-12

If menu options are included in a display intended also for data review and/or data entry, which is often a practical design approach, the labels for control entry should be located consistently in the display and should incorporate some consistent distinguishing feature to indicate their special function.

Example: All control options might be displayed beginning with a special symbol, such as a plus sign (+FORWARD, +BACK, etc.)

See also: 2.1.3-6, 4.0-11.

-13 Logical Ordering of Menu Options

-13

Displayed menu options should be listed in a logical order; if no logical structure is apparent, then options should be displayed in order of their expected frequency of use, with the most frequent listed first.

Example: (Good) Indicate vehicle type.
c = passenger Car
t = pick-up Truck
b = Bus

(Bad) Indicate vehicle type.
t = pick-up Truck
b = Bus
c = passenger Car

Comment: If the first menu option is always the most likely choice, then for some applications it may be useful for efficiency of sequence control if a null entry defaults to the first option. If that is done, it should be done consistently.

Reference: BB 2.9.4; EG 2.3.1; PR 4.6.6; Palme, 1979.

See also: 2.1.3-5, 2.3-5.

-14 Hierarchical Structure for Menus

-14

Displayed menu lists should be formatted to indicate the hierarchic structure of logically related groups of options, rather than as an undifferentiated string of alternatives.

Example: In vertical listing of options, subordinate categories might be indented.

Comment: Logical grouping of menu options will help users learn system capabilities.

Comment: When logical grouping requires a trade-off against expected frequency of use, USI designers should resolve that trade-off consistently throughout the menu structure.

Reference: EG 2.2.8, 2.3; Liebelt, McDonald, Stone and Karat, 1982.

See also: 4.4-3.

-15 Menu Options Ordered by Frequency of Use**-15**

If menu options are grouped in logical subunits, those groups should be displayed in order of their expected frequency of use.

Reference: PR 4.6.6.

-16 Labeling Grouped Options**-16**

If menu options are grouped in logical subunits, each group should be given a descriptive label that is distinctive in format from the option labels themselves.

Comment: Although this practice might sometimes seem to waste display space, it will help provide user guidance; moreover, careful selection of group labels may serve to reduce the number of words needed for individual option labels.

Reference: MS 5.15.2.10.

See also: 4.4-4.

-17 Tailored Menu Options

-17

A displayed menu should include only options appropriate at that particular step in a transaction sequence, and for the particular user.

Example: Displayed file directories should contain only those files actually available to the user.

Example: An UPDATE option should be offered only if the user has update rights for the particular data file being used.

Exception: Menu displays for a system still under development might indicate future options not yet implemented, but those options should be specially designated in some way.

Comment: A seeming exception might be a process control display in which current values of a number of variables must be monitored (i.e., must be displayed continuously), and where supplementary data (e.g., trend analysis) can be called out for some variables but not others. Here some means must be found to signal the user which variables can be selected and which not.

See also: 3.2-11, 4.4-1.

-18 Complete Display of Explicit Options

-18

Insofar as possible a displayed menu should include all options appropriate at that particular step in a transaction sequence.

Exception: A familiar set of general control options always available may be omitted from individual displays, and accessed as needed by a +OPTIONS entry.

See also: 4.4-1, and Section 3.2.

-19 Sequential Selection from Hierarchic Menus

-19

When menu selection must be made from a long list, and not all options can be displayed at once, a hierarchic sequence of menu selections should be provided rather than one long multi-page menu.

Exception: Where a long list is already structured for other purposes, such as a list of customers, a parts inventory, a file directory, etc., it might be reasonable to require the user to scan multiple display pages to find a particular item. Even in such cases, however, an imposed structure for sequential access may prove more efficient, as when a user can make preliminary letter choices to access a long alphabetic list.

Comment: Beginning users may prefer a menu permitting a single choice from all available options, when those can be displayed on one page. Experienced users, however, may perform faster with a sequence of choices from a hierarchy of separately displayed sub-menus.

Comment: A single menu that extends for more than one page will hinder learning and use. The USI designer can usually devise some means of logical segmentation to permit several sequential selections among few alternatives instead of a single difficult selection among many.

Reference: Dray, Ogden and Vestewig, 1981.

See also: 3.1.3-20, 4.4-4.

-20 Minimal Steps in Sequential Menu Selection**-20**

When the user must step through a sequence of menus to make a selection, the hierarchic structure should be designed, insofar as possible within the constraints of display space, to minimize the number of steps required.

Comment: This represents a trade-off against the need for logical grouping in hierarchic menus. The number of hierarchic levels should be minimized, but not at the expense of display crowding.

Comment: When space permits, it may be desirable to display further (lower) choices in the hierarchic structure, to give the user a deeper view of the structure and permit direct selection of specific lower-level options.

Reference: MS 5.15.2.2.a; Miller, 1981.

See also: 3.1.3-19.

-21 Easy Selection of Important Options**-21**

When hierarchic menus are used, they should be designed to permit the user immediate access to critical or frequently selected options.

Reference: MS 5.15.2.2.b.

-22 Display of Menu Structure**-22**

When hierarchic menus are used, the user should be given some displayed indication of current position in the menu structure.

Comment: One possible approach would be to recapitulate prior (higher) menu selections on the display. If routine display of path information seems to clutter menu formats, then a map of the menu structure might be provided at user request as an optional HELP display.

Reference: MS 5.15.2.2.c; Billingsley, 1982.

See also: 4.4-4.

-23 Consistent Format for Hierarchic Menus -23

When hierarchic menus are used, care should be taken to ensure consistent display formats at each level.

Reference: MS 5.15.2.2.d.

See also: 4.0-5.

-24 Return to Higher-Level Menus -24

When hierarchic menus are used, a single key action should permit the user to return to the next higher level.

Reference: BB 4.4.4.

-25 Consistent Display of Menu Options -25

Menus provided in different displays should be designed so that option lists are consistent in terminology and ordering.

Example: If +PRINT is the last option in one menu, the same print option should not be worded +COPY at the beginning of another menu.

See also: 4.0-6.

-26 Feedback for Menu Selection -26

When a control option has been selected and entered, if there is no immediately observable natural response some other form of acknowledgment should be displayed.

Comment: An explicit message might be provided. In some applications, however, it may suffice simply to highlight the selected option label (e.g., by brightening or inverse video) when that would provide an unambiguous computer acknowledgment.

Reference: MS 5.15.1.4.a.

See also: 1.1-6, 4.2-1, 4.2-10.

-27 By-Passing Menu Selection with Command Entry

-27

Experienced users should be provided means to by-pass a series of menu selections and make an equivalent command entry directly.

Reference: BB 6.7.

See also: 3.0-2, 4.4-26.

-28 Stacking Sequential Menu Selections

-28

When a user can anticipate menu selections before they are presented, means should be provided to enter several "stacked" selections at one time.

Comment: If necessary, stacked sequential entries might be separated by a special character, such as a slash, comma or semicolon. It would be preferable, however, if they were simply strung together without special punctuation.

Reference: BB 6.8.

-1 Function Keys

-1

Function keys should be considered for tasks requiring only a limited number of control entries, or in conjunction with other dialogue types as a ready means of accomplishing critical entries that must be made quickly without syntax error.

Reference: MS 5.15.3.7.

-2 Function Keys for Frequent Control Entries

-2

Function keys should be considered as a means of accomplishing frequently required control entries.

Example: ENTER, PRINT, PAGE FORWARD, PAGE BACK, OPTIONS, etc.

Comment: When generally used options are always implicitly available via function keys, they need not be included in displayed menus.

Reference: BB 4.4.

See also: 3.1.3-18, and Section 3.2.

-3 Function Keys for Interim Control Entries

-3

Function keys should be used as a means of permitting interim control entries, i.e., for control actions taken before the completion of a transaction.

Example: TAB, DITTO, DEFAULT, HELP, etc.

-4 Labeling Function Keys

-4

Function keys should be labeled informatively to designate the function they perform; labels should be sufficiently different from one another to prevent user confusion.

Example: Log-on should not be initiated by a key labeled PANIC. (This example may seem unlikely, but is cited from an actual USI design.)

Example: Two keys should not be labeled ON and DN.

Reference: BB 4.4.7; MS 5.15.2.10.c.

See also: 4.0-13.

-5 Single Label for Continuous Functions

-5

For a single-purpose function that is continuously available, just one label should be on the key.

-6 Labeling Multifunction Keys

-6

If a function key is used for more than one function, the user should always have some convenient means of knowing which function is currently available.

Comment: If a key is used for just two functions, depending upon defined operational mode, then alternate self-illuminated labels should be provided on the key to indicate which function is current. In these circumstances, it is preferable that only the currently available function is visible, so that the labels on a group of keys will show what can be done at any point rather than what has been done.

Comment: If the function of a key is specific to a particular step in the transaction sequence, then the current function should be indicated by an appropriate guidance message on the user's display.

Reference: MS 5.15.3.8.

See also: 4.4-10.

-7 Consistent Assignment of Function Keys

-7

If a function is assigned to a particular key for one task/transaction, that function should be assigned consistently to the same key in other transactions.

Reference: BB 4.4.7.

-8 Consistent Functions in Different Operational Modes

-8

When a function key performs different functions in different operational modes, those functions should be made as consistent as possible.

Example: A key labeled RESET should not be used to dump data in one mode, save data in another, and signal task completion in a third.

-9 Return to Base-Level Functions

-9

When the set of functions assigned to keys changes as a result of user selection (so-called multifunction keys), an easy means should be provided for the user to return to the initial, base-level functions.

Comment: In effect, multifunction keys can provide hierarchic levels of options much like menu selection dialogues, with the same need for rapid return to the highest-level menu.

Comment: For some applications, it may be desirable to automate the return to base-level assignment of multifunction keys, to occur immediately on completion of a transaction and/or by time-out following a period of user inaction. The optimum period for any automatic time-out would have to be determined empirically for each application.

Reference: Aretz and Kopala, 1981.

-10 Disabling Unneeded Function Keys

-10

Function keys (and other devices) not needed for current control entry should be temporarily disabled under computer control at any step in a transaction sequence; mechanical overlays manipulated by the user should not be used for this purpose.

Comment: If the user selects a function key that is invalid at a particular step in a transaction sequence, no action should result except display of an advisory message indicating what functions are available at that point.

Reference: MS 5.15.3.9.4.3; PR 4.12.4.5.

See also: 3.2-11, 6.5-7.

-11 Distinguish Active Function Keys

-11

When some function keys are active and some are not, the current subset of active keys should be indicated in some noticeable way to the user, perhaps by brighter illumination.

Comment: This practice will speed user selection of function keys.

Reference: Hollingsworth and Dray, 1981.

See also: 4.4-10.

-12 Distinctive Location for Function Keys

-12

Function keys should be grouped in distinctive locations on the keyboard to facilitate their learning and use; frequently used function keys should be placed in the most convenient locations.

Comment: It is preferable that frequently used keys not require double (control/shift) keying.

Reference: MS 5.15.4.6.1.d.

See also: 4.0-11.

-13 Function Key Location Compatible with Use

-13

The layout of function keys should be compatible with their importance; keys for emergency functions should have a prominent position and distinctive coding (e.g., size and/or color); keys with potentially disruptive consequences should be physically protected.

See also: 6.5-8.

-14 Single Activation for Function Keys

-14

Function keys should require only single activation to accomplish their function, and should not change function with repeated activation.

Example: Log-on should not be initiated by pressing a PANIC key twice.

See also: 4.0-2.

-15 Feedback for Function Key Activation

-15

When function key activation does not result in any immediately observable natural response, the user should be given some other form of computer acknowledgment.

Comment: Temporary illumination of the function key would suffice, if key illumination is not used to signal available options. Otherwise a displayed advisory message should be used.

Comment: As an interesting variation, user guidance prior to key activation might be provided, where partial depression of a double-contact function key would explain its use, either by voice output ("talking keyboard") or by visual display of a HELP message.

Reference: Geiser, Schumacher and Berger, 1982.

See also: 4.2-1.

-1 Command Language

-1

Command language dialogue should be considered for tasks involving a wide range of user control entries, where users may be highly trained in the interests of achieving efficient performance, and where computer response is expected to be relatively fast.

Comment: Command language should also be considered for data entry in arbitrary sequence.

-2 Command Entry Area on the Display

-2

When command language is used for control entry, an appropriate entry area should be provided in a consistent location on every display, preferably at the bottom.

Comment: Adjacent to the command entry area there should be a defined display window used for prompting control entry, for recapitulation of command sequences (with scrolling to permit extended review), and to mediate question-and-answer dialogue sequences (i.e., prompts and responses to prompts).

Reference: EG 2.3; MS 5.15.4.6.1.d; Granda, Teitelbaum and Dunlap, 1982.

See also: 2.5-8, 4.0-5.

-3 Familiar Wording for Commands

-3

The words chosen for a command language should reflect the user's point of view and not the programmer's, corresponding consistently with the user's operational language.

Reference: EG 4.1.1, 4.2.12, 4.2.13; MS 5.15.1.4.5.

See also: 4.0-16, 4.0-17.

-4 Abbreviation of Commands**-4**

Abbreviation of entered commands (i.e., entry of the first 1-3 letters) should be permitted to facilitate entry by experienced users.

Example: If a "P" uniquely identifies a print command (i.e., no other commands start with "P") then the user should be able to enter PRINT, or PR, or P, or any other truncation to initiate printing.

Comment: As a corollary, misspelling of command entries should also be tolerable, within the limits of computer recognition. The computer can interrogate the user as necessary to resolve ambiguous entries.

Comment: Variable abbreviation, i.e., keying only enough characters of a command to uniquely identify it, should probably not be used when the command set is changing. For the user, an abbreviation that works one day may not work the next. For the programmer, the addition of any new command may require software revision of recognition logic for other commands.

Reference: BB 6.4.3; Demers, 1981.

See also: 3.1.5-15.

-5 Consistent Wording of Commands**-5**

All words in a command language, and their abbreviations, should be consistently used and standardized in meaning from one transaction to another, and from one task to another.

Example: Do not use EDIT in one place, MODIFY in another, UPDATE in a third, all referring to the same kind of action.

Reference: EG 4.2.9, 4.2.13; MS 5.15.2.6; Demers, 1981.

See also: 4.0-6, 4.0-17.

-6 Distinctive Wording of Commands**-6**

Words in a command language should be chosen to be distinctive from one another, and to emphasize significant differences in function, in order to minimize user confusion.

Example: Do not label two commands DISPLAY and VIEW, when one permits editing displayed material and one does not.

Comment: In general, do not give different commands semantically similar names, such as SUM and COUNT.

Reference: BB 6.2.5; MS 5.15.2.10.c.

-7 User-Assigned Command Names**-7**

A command language should provide flexibility, permitting the user to assign personal names to files, frequently used command sequences, etc.

Comment: Frequently used commands should be made easy to accomplish. Where users will differ in the frequency of the commands they use, the designer should provide for flexibility in command naming.

-8 Functional Commands**-8**

A command language should be supported by whatever computer processing is necessary so that the user can manipulate data without concern for internal storage and retrieval mechanisms.

Example: The user should be able to request display of a file by name alone, without having to enter any further information such as file location in computer storage.

Comment: Where file names are not unique identifiers, the computer should be programmed to determine whatever further context is necessary for identification, automatically in relation to the current transaction sequence, or perhaps by asking the user to designate a "directory" defining a subset of files of current interest.

-9 Layered Command Language

-9

The features of a command language should be designed in groups (or "layers") for ease in learning and use.

Comment: The fundamental core, or bottom layer, of the language should be the easiest, allowing use of the system by people with little training and/or limited needs. Successive layers of the command language can then increase in complexity for users with greater skills.

Reference: Reisner, 1977.

See also: 4.4-26.

-10 User-Requested Prompts for Command Entry

-10

The user should be able to request prompts as necessary to determine required parameters in a command entry, or to determine available options for an appropriate next command entry.

Example: Keying a question mark in the command entry area would be a satisfactory method of requesting prompts, or else using an explicitly labeled HELP function key.

Comment: In some applications it may be desirable to let an inexperienced user simply choose a general "prompt mode" of operation, where any command entry produces automatic prompting of (required or optional) parameters and/or succeeding entry options.

Reference: MS 5.15.1.4.5; Demers, 1981.

See also: 4.4-7, 4.4-9.

-11 Command Stacking

-11

When command entries are prompted automatically, it should be possible for an experienced user to key a series of commands at one time ("command stacking") so as to shortcut the prompting sequence.

Reference: BB 6.8.

See also: 4.4-9, and Section 3.2.

-12 Minimal Command Punctuation -12

Insofar as possible, the user should be able to enter commands without punctuation.

-13 Standard Command Delimiter -13

If punctuation is needed, perhaps as a delimiter to distinguish optional parameters, or the separate entries in a stacked command, one standard symbol should be used consistently for that purpose, preferably the same symbol (slash) used to separate a series of data entries.

See also: 1.4-4, 3.2-17.

-14 Ignoring Blanks in Command Entry -14

Neither the user nor the computer program should have to distinguish between single and multiple blanks in a command entry.

See also: 1.0-25.

-15 Misspelled Commands -15

Where the set of potential command entries is well defined, the computer should be programmed to recognize common misspellings of commands, and to display inferred correct commands for user confirmation rather than requiring re-entry.

Comment: This practice should permit a sizable reduction in wasted keying without serious risk of misinterpretation. The necessary software logic is akin to that for recognizing command abbreviations.

Reference: BB 6.2.4; Gade, Fields, Maisano, Marshall and Alderman, 1981.

-16 Resolving Ambiguous Commands

-16

When command entries are subject to misinterpretation (as in the case of voice input), or when an interpreted command may have disruptive consequences, the user should be given an opportunity to review and confirm a displayed interpretation of the command before it is executed.

Comment: For beginning users, it might be desirable to permit review of interpreted commands for every transaction. Skilled users, however, should be able to suppress such routine review.

See also: 6.0-7, 6.5-11.

-17 Correcting Command Errors

-17

When a command entry is not recognized, the computer should initiate a clarification dialogue, rather than rejecting the command outright.

Comment: Poorly stated commands should not simply be rejected. Instead, the computer should be programmed to guide the user toward a proper formulation, preserving the faulty command for reference and modification, and not require the user to re-key the entire command just to change one part.

See also: 4.3-14.

-1 Query Language

-1

Query language dialogue should be considered as a specialized sub-category of general command language for tasks emphasizing unpredictable information retrieval (as in many analysis and planning tasks), with moderately trained users and fast computer response.

Comment: All recommendations for command language design would apply equally to query languages, with the addition of some more specific guidelines listed below.

Reference: Ehrenreich, 1981.

-2 Natural Organization of Data

-2

When the organization of the computer data base is reflected in the query language, that organization should match the data structure perceived by users to be natural.

Comment: The users' natural perception of data organization can be discovered through experimentation or by survey.

Reference: Durdling, Becker and Gould, 1977.

-3 Representation of Data Organization

-3

One single representation of the data organization should be established for use in query formulation, rather than multiple representations.

Comment: Beginning or infrequent users may be confused by different representational models.

See also: 4.4-14.

-4 Minimal Use of Quantifiers

-4

The need for quantificational terms in query formulation should be minimized.

Exception: "no" or "none".

Comment: People have difficulty in using quantifiers unambiguously. When quantifiers must be used, it may be desirable to have the user select the desired quantifier from a set of sample statements so worded as to maximize their distinctiveness.

-5 Minimal Use of Confusing Operators

-5

Use of operators subject to frequent semantic confusion, such as "or more" and "or less", should be minimized.

Example: A user should not have to convert "over 50 years old" into "51 or more".

-1 Natural Language

-1

Unconstrained natural language dialogue should not be considered for USI design at this time; natural language may find future use in applications where task requirements are broad ranging and poorly defined, where little user training can be provided, and where computer response will be fast.

Comment: Computer processing of natural language is now being developed on an experimental basis. For current applications where task requirements are well defined, other types of dialogue will prove more efficient.

Reference: Shneiderman, 1981.

-1 Graphic Interaction

-1

Graphic interaction should be considered as a supplement to other forms of man-machine dialogue where special task requirements exist; effective implementation of graphic capabilities will require very fast computer response.

-1 User Control in Transaction Selection

-1

The sequence of transaction selections should generally be dictated by the user's choices and not by internal computer processing constraints.

Example: A data entry clerk should be able to enter items in whatever order they are available, when order is variable, as in taking reservation data by telephone.

Comment: In some cases this means that the computer may have to store current entries until they become relevant to subsequent data processing.

Comment: When a logical sequence of transactions can be determined in advance, USI design might encourage and help a user to follow that sequence. Guidance may be desirable though constraint is not.

Reference: PR 4.6.7.

See also: 3.0-1, 3.0-5, 3.0-6, 4.0-2.

-2 General Menu of Control Options

-2

An initial OPTIONS menu should always be available for user selection, to serve as a "home base" or consistent starting point for control entries at the beginning of a transaction sequence.

Comment: Such a starting point is helpful even when all dialogue is user-initiated. This capability can be implemented as an OPTIONS function key, or as an explicit control option on every display, or as a generally available implicit option, or as a consistent default for a null control entry.

Reference: BB 4.1; PR 3.3.16.

See also: 4.4-2.

-3 OPTIONS Menu

-3

The general OPTIONS menu should show primary control entry options grouped, labeled and ordered in terms of their logical function, frequency and criticality of use, following the guidelines provided for menu selection.

See also: Section 3.1.3, 4.4-2, 4.4-3.

-4 Availability of Control Entries

-4

The user should be able to make at least some sequence control entries directly at any step in a transaction sequence (i.e., from any display page) without having to return to a general OPTIONS menu.

Comment: In particular, the user should not have to remember data or control codes given on one display for later entry on a different display.

See also: 4.4-1.

-5 Prompting Control Entries

-5

Information should be provided the user concerning control entries specifically appropriate at any step in a transaction sequence, either incorporated in the display or else available through a request for HELP.

Reference: MS 5.15.3.5.

See also: 4.4-1.

-6 Implicit Control Options

-6

Control options that are generally available at any step in a transaction sequence should be treated as implicit options, i.e., need not be included in a display of step-specific options; frequently used implicit options should be entered by function keys.

Comment: Experienced users may prefer to select implicit options by command entry.

See also: 3.1.4-2, 4.4-6.

- 7 Menu Selection by Pointing -7
- When selection among displayed menu options is to be accomplished by pointing, the cursor should be placed automatically on the first (most likely) option at initial display generation.
- See also: 3.1.3-13, 4.4-12.
- 8 Menu Selection by Keyed Entry -8
- When selection among displayed menu options is to be accomplished by keyed entry of a corresponding code, the cursor should be placed automatically in the command entry area at initial display generation.
- Reference: PR 4.7.1.
- See also: 4.4-12.
- 9 Codes for Menu Selection -9
- When displayed menu options can be selected by code entry, the code associated with each option should be included on the display in some consistent, identifiable manner.
- Example: In many applications an equal sign can be used for this purpose, as N = NEXT PAGE, P = PREV PAGE, etc.
- 10 Task-Oriented Wording for Options -10
- The wording of step-specific control options should reflect the current concerns and likely questions of the user at that step in a transaction sequence.
- Reference: MS 5.15.2.10.d.
- See also: 4.0-17.

-11 Only Active Options Offered -11

The user should be offered only control options that are actually available.

Comment: If certain options are not yet implemented, as during system development, or not available for any other reason, those should be annotated on the display.

See also: 3.1.3-17, 3.1.4-10, 6.5-7.

-12 Consistent Default Options -12

If a default for a null control entry is defined for any step in a transaction sequence, that default should be consistent in USI design, or else indicated in the display of step-specific control options.

Example: In menu selection, a null entry might always default to the first of the displayed options.

Reference: EG 4.2.4.

See also: 4.4-7.

-13 Consistent STEP Command -13

At any step in a defined transaction sequence, if there are no alternative step-specific control options, then a consistent command should be used to continue to the next step.

Example: NEXT, STEP or CONTINUE might be suitable names for this function.

Exception: If data entry is involved, then an explicit ENTER command should be used.

Reference: PR 4.11.

See also: 3.2-6, 4.4-5.

-14 Stacked Commands

-14

When control entry involves user-composed commands, the user should be permitted to key a sequence of codes for option selection as a single "stacked" command.

Example: In particular, the user should be able to enter stacked commands from the initial menu of general OPTIONS, so that an experienced user can make any specific control entry from the first step in a transaction sequence.

Example: Command stacking may be helpful when a user is being prompted to enter a series of parameter values, and knows in advance what several succeeding prompts will request and what values to enter.

Comment: Command stacking will permit a transition from simple step-by-step control entry by novice users, as in menu selection and question-and-answer dialogues, to the entry of extended command-language statements by experienced users; command stacking is especially helpful in time-shared systems where computer response to any user entry may be slow.

Reference: EG 6.2, 6.2.1; PR 2.6, 4.7.3; Palme, 1979.

See also: 3.1.5-11.

-15 Consistent Order in Command Stacking

-15

In command stacking, user entries should be in the same order as they would normally be made in a succession of separate command entry actions.

Reference: EG 6.2.1.

-16 Abbreviation in Command Stacking

-16

In command stacking, acceptable user entries should include command names or their abbreviations or defined codes; if stacked command entries are potentially ambiguous, the computer should display the interpreted command sequence for user confirmation or correction.

Reference: EG 6.2.1.

-17 Standard Delimiter in Command Stacking

-17

In command stacking, if some special symbol must be used to separate command entries, then one standard symbol should be adopted for that purpose, preferably the same symbol (slash) used as a delimiter for sequential data entries.

Reference: EG 6.2.1.

See also: 1.4-4, 3.1.5-13.

-18 Incomplete Stacked Commands

-18

If a stacked command results in only partial completion of a menu selection sequence (i.e., if further user selections must be made), then the appropriate next menu display should be presented to guide completion of control entry.

Reference: PR 4.7.3.

See also: 4.4-7.

-19 User Definition of Macro Commands

-19

Flexibility in transaction selection should be provided by permitting the user to assign a single name to a defined series of control entries, and then use this new "macro" for subsequent command entry.

Comment: In this way the user can make frequently required but complicated tasks easier to accomplish, when the USI designer has failed to anticipate the particular need.

Reference: Demers, 1981.

-1 User Interrupt

-1

Flexibility in control should be provided by permitting the user to interrupt, defer or abort a current transaction sequence, in consistently defined ways appropriate to specific task requirements.

Comment: For an experienced user, it may be desirable to permit multi-tasking of the computer, so that a transaction involving slow data processing can be accomplished as "background" to interim shorter transactions. In such a case, of course, the computer must be programmed to signal completion of the background transaction.

Comment: Provision of flexible interrupt capabilities for the user will generally require some sort of suspense file or other buffering in software design. Such capabilities are valuable, however, in permitting the user to correct mistaken entries, and in permitting the computer to require user confirmation of potentially destructive entries.

Reference: PR 3.3.16, 3.3.17.

-2 Interrupt Options

-2

If different kinds of interruption in sequence control are provided, each kind should be accomplished by a differently named control option.

Comment: As a negative example, it would not be good design practice to provide a single ESCAPE key which has different effects depending upon whether it is pushed once or twice; the user may be confused by such expedients, and uncertain about what action has been taken and its consequences.

-3 CANCEL Option

-3

If appropriate to sequence control, a CANCEL option should be provided, which will have the consistent effect of regenerating the current display without processing any interim changes made by the user.

Comment: In effect, interim entries would be erased.

See also: 1.4-2.

-4 BACKUP Option

-4

If appropriate to sequence control, a BACKUP option should be provided, which will have the consistent effect of returning to the display entered in the last previous transaction.

Comment: Such a BACKUP capability will generally prove feasible only in the software design of well-defined transaction sequences, but will prove helpful when it can be provided.

Comment: BACKUP might be designed to include cancellation of any interim entries made in a pending transaction. Alternatively, pending entries might be preserved without processing. USI design should be consistent in this regard.

Reference: MS 5.15.1.2.6.

See also: 1.4-2.

-5 RESTART Option

-5

If appropriate to sequence control, a RESTART option should be provided, which will have the consistent effect of returning to the first display in a defined transaction sequence, permitting the user to review a sequence of entries and make necessary changes.

Comment: As an extension of the BACKUP capability, RESTART is useful only in well-defined transaction sequences such as step-by-step data entry in a question-and-answer dialogue.

Comment: RESTART might be designed to include cancellation of any interim entries made in a pending transaction. Alternatively, pending entries might be preserved without processing. USI design should be consistent in this regard.

See also: 1.4-2.

-6 ABORT Option**-6**

If appropriate to sequence control, an ABORT option should be provided, which will have the consistent effect of canceling all entries in a defined transaction sequence; when data entries or changes will be nullified by an ABORT action, the user should be asked in an advisory message to CONFIRM the ABORT.

Comment: An ABORT action, combining the functions of RESTART and CANCEL, is again relevant only to well-defined transaction sequences, specifically those with a recognized beginning.

Reference: BB 1.8.

See also: 3.0-10, 4.2-11, 6.5-3.

-7 END Option**-7**

If appropriate to sequence control, an END option should be provided, which will have the consistent effect of concluding a repetitive transaction sequence and returning control to a general OPTIONS menu.

Example: In routine data entry, where the end of one transaction is designed to lead automatically to the beginning of the next transaction, the user needs some control option such as END to signal when a batch of transactions has been completed.

Comment: END can be implemented by whatever means are appropriate to the dialogue design, i.e., by menu selection, command entry, or function key.

Reference: EG 4.2.10.

See also: 3.0-15.

-1 Context Definition

-1

Sequence control software should be designed to maintain context for the user throughout the series of transactions comprising a task, recapitulating previous entries affecting present actions, and indicating currently available options where appropriate.

See also: 1.8-3, 3.0-17, 4.4-10.

-2 Record of Prior Entries

-2

In tasks where transaction sequences are variable, the user should be able to request a displayed list of prior entries if needed to determine present status.

Comment: Such a capability may not be needed for routine transactions if they are designed in such a way that each step identifies its predecessors explicitly, although even in those circumstances a user may be distracted and at least momentarily become confused.

Reference: EG 4.2.7.

See also: 4.4-18.

-3 Context Established by Prior Entries

-3

Insofar as possible, sequence control software should be designed to carry forward a representation of the user's knowledge base and current activities; the user should not have to re-enter previously entered data relevant to current control entries.

Example: If data have just been stored in a named file, then the user should be able to request a printout of that file without having to re-enter its name.

Exception: If transactions involving contextual interpretation would have destructive effects (e.g., data deletion), then the interpreted command should be displayed first for user confirmation.

Comment: The software logic supporting contextual interpretation of control entries need not be perfect in order to be helpful. When ambiguity results, it may still be easier for the user occasionally to review and correct an interpreted command than always to generate a complete command initially.

Reference: MS 5.15.2.9; PR 2.3.

-4 Display of Operational Mode

-4

When context for sequence control is established in terms of a defined operational mode, then some means should be provided to remind the user of the current mode and other pertinent information.

Example: If text is displayed in an editing mode, then a caption might indicate EDIT as well as the name of the displayed text; if an INSERT mode is selected for text editing, then some further displayed signal should be provided.

Reference: EG 4.2.1.

See also: 4.2-8, 4.4-10.

-5 Display of Operative Parameters

-5

The value of any control parameter(s) currently operative should be displayed for user reference.

Comment: This practice is helpful even when the user selects all parameters himself, since he may well forget them, particularly if his task activities are interrupted.

Comment: When there are a large number of currently operative control parameters, it may prove impractical to display them continuously. In such a case, it may suffice to list them on an auxiliary HELP display accessed by user request.

Reference: MS 5.15.3.5.b.

See also: 4.2-10, 4.4-10.

-6 Highlighting Selected Data

-6

When a user is performing an operation on some selected display item, that item should be highlighted.

Comment: This practice will help avoid error, if the user has misunderstood or perhaps forgotten which item was selected.

Reference: EG 2.1.1.

See also: 4.2-10.

-7 Consistent Context Display

-7

Whatever information is given the user to provide context for sequence control should be distinctive in location and format, and consistently displayed from one transaction to the next.

Reference: MS 5.15.3.6, 5.15.4.4.

See also: 4.0-5.

-1 Error Management

-1

The computer software should deal appropriately with all possible control entries, correct and incorrect, without inducing errors in sequence control.

Example: If the user selects a function key that is invalid at a particular step in a transaction sequence, no action should result except display of an advisory message indicating what functions are appropriate at that point.

Comment: For certain routine and easily recognized errors, such as trying to tab beyond the end of a line, a simple auditory signal ("beep") may be sufficient computer response.

Reference: PR 4.12.4.5.

See also: 3.1.4-10, 6.0-5.

-2 Command Editing

-2

The user should be able to edit an extended command during its composition, by backspacing and rekeying, before taking an explicit action to ENTER the command.

Comment: Users can often recognize errors in keyed entries prior to final entry.

Reference: EG 5.4.

See also: 1.4-2, 6.0-6, 6.3-10.

-3 Prompting Command Entry

-3

If an element of a command entry is not recognized, or logically inappropriate, sequence control software should prompt the user to correct that element without having to re-enter the entire command.

Example: A faulty command can be retained in the command entry area of the display, with the cursor automatically positioned at the incorrect item, plus an advisory message describing the problem.

Reference: BB 1.3; EG 4.2.2, 4.2.3; MS 5.15.1.2.1, 5.15.1.2.7.b.

See also: 4.3-14.

-4 Errors in Stacked Commands

-4

If an error is detected in a stacked series of command entries, USI design should be consistent in how this is handled, from one transaction sequence to another.

Comment: It may help the user if the commands are executed to the point of error, or it may not. In most applications, partial execution will probably prove desirable. The point is that a considered USI design decision should be made and then followed consistently.

Reference: BB 1.5; EG 5.6; PR 4.7.3.

-5 Partial Execution of Stacked Commands

-5

If only a portion of a stacked command can be executed, that problem should be indicated to the user with appropriate guidance to permit completion of the intended control entry.

Reference: Dwyer, 1981.

See also: 3.2-19, 4.4-7.

-6 Explicit Action for Command Entry -6

The ENTER action for command entry should be the same as that for data entry; direct selection of menu options should also require some explicit ENTER action.

Comment: When a common action is used for both data entry and command entry, there will be less likelihood of user confusion and error.

See also: 1.0-8, 1.1-4, 3.0-5.

-7 Explicit Action for Error Correction -7

When a user completes correction of an error, whether of a command entry or data entry, the user should be required to take an explicit action to re-enter the corrected material; the new ENTER action should generally be the same as the action that was used for the original entry.

Reference: PR 4.12.4.6.

See also: 6.0-3, 6.2-14.

-8 User Confirmation of Interpreted Commands -8

When a default value is included in command entry, it may be helpful to recapitulate the command in its fully interpreted form for user confirmation; if this practice is followed, it should be done consistently.

See also: 6.0-7.

-9 User Confirmation of Destructive Control Entries -9

When a control entry will cause any extensive change in stored data, procedures and/or system operation, and particularly if that change cannot be easily reversed, the user should be notified and required to confirm the action before it is implemented.

Reference: BB 1.10; EG 4.1.2, 4.2.8; MS 5.15.1.2.3, 5.15.1.2.7.c.

See also: 3.0-3, 4.3-16, 6.5-14, 6.5-18.

-10 User Warned of Potential Data Loss

-10

The prompt for CONFIRM action should be worded in such a way that potential data loss is clearly stated.

Example: (Good) CONFIRM DELETION OF ENTIRE AIRFIELD FILE

(Bad) CONFIRM DELETE ACTION

See also: 4.3-15, 6.5-17.

-11 Distinctive CONFIRM Action

-11

User confirmation of a control entry or data entry should be accomplished with an explicitly labeled CONFIRM function key, different from the ENTER key.

Comment: Confirmation should not be accomplished by pushing some other key twice.

Comment: Some USI designers recommend that in special cases confirmation should be made more difficult still, e.g., by keying the separate letters in C-O-N-F-I-R-M. A better approach might be to make any user action, including over-hasty confirmation, immediately reversible, a feature now available in some current USI designs.

See also: 3.1.4-4, 3.1.4-14, 6.5-19, 6.5-20.

-12 Preventing Data Loss at LOG-OFF

-12

When a user requests LOG-OFF, sequence control software should check pending transactions and, if data loss seems probable, should display an advisory message requesting confirmation.

Example: CURRENT DATA ENTRIES HAVE NOT BEEN FILED;
SAVE IF NEEDED, BEFORE CONFIRMING LOG-OFF.

Comment: The user may sometimes suppose that a job is done before taking necessary implementing actions.

See also: 4.3-15, 6.5-16.

-13 Immediate Error Correction

-13

When a data entry transaction has been completed and errors detected, sequence control logic should permit direct, immediate correction by the user.

Comment: It is helpful to correct data entry errors at the source, i.e., while the user still has the entry in mind and/or source documents at hand.

Comment: For transactions involving extended entry of multiple items, computer checking might be invoked by separate entry of each page (or section) of data.

Reference: EG 5.7; PR 2.5.

See also: 1.7-4, 6.3-11.

-14 Flexible BACKUP for Error Correction

-14

A user should be able to return easily to previous steps in a transaction sequence in order to correct an error or make any other desired change.

Reference: MS 5.15.1.2.6.

See also: 6.3-13, and Section 3.3.

-15 Changes to Displayed Data

-15

When considerations of data security do not prohibit, the user should be able to change any data that are currently displayed.

Comment: Users should not have to specify that changes will be made in advance of calling for a display. That practice may be simpler for the software designer, but is confusing for a user.

See also: 6.2-3.

-1 Alarm Definition**-1**

In many applications, particularly those involving monitoring and process control, the user (or some authorized supervisor) should be permitted to define conditions, in terms of variables and values, that will result in automatic generation of alarm messages.

Example: The nurse in charge of an intensive care monitoring station might need to specify for each patient warning signals when blood pressure ("variable") exceeds or falls below defined levels ("values").

Exception: Situations where alarm conditions must be pre-defined by functional, procedural, or perhaps even legal requirements, such as violation of aircraft separation in air traffic control.

See also: 4.1-9.

-2 Consistent Alarm Signals**-2**

Alarm signals and messages may take a variety of forms, but should be distinctive and consistent for each class of events.

Comment: The user might be permitted to define the nature of each alarm as well as its initiating event.

See also: 4.3-17.

-3 Alarm Acknowledgment**-3**

The user should be provided a simple, consistent means of acknowledging and turning off non-critical alarm signals.

Example: A function key labeled ALARM ACK would suffice for that purpose.

Reference: MS 5.15.4.6.1.a.

-4 Acknowledgment of Critical Alarms

-4

The user may be required to take more complicated actions in order to respond to critical alarms, and to acknowledge special alarms in special ways; but such special acknowledgment actions should be designed so that they will not inhibit or slow remedial user response to a critical initiating condition.

See also: 3.0-3.

-1 Flexible Design for Sequence Control

-1

When sequence control requirements may change, which is often the case, some means should be provided for the user (or an authorized supervisor) to make necessary changes to control functions.

Comment: Sequence control features that may need to be changed include those represented in these guidelines, namely, the types of dialogue that are provided, procedures for transaction selection and user interrupt, methods for context definition and error management, and alarm logic.

SECTION 4

USER GUIDANCE

The fundamental objectives of user guidance are to promote efficient system use (i.e., quick and accurate use of full capabilities), with minimal memory load on the user and hence minimal time required to learn system use, and with flexibility for supporting users of different skill levels.

A narrow view of user guidance has to do with preventing and correcting user errors. But minimizing user errors may require improvements in broad aspects of USI design -- in techniques for data display, in procedures for data entry and sequence control -- as well as provision of user guidance. Moreover, any general consideration of user guidance functions must include provision of status information, job aids, and routine feedback, as well as feedback for error correction.

Many USI design features contribute directly or indirectly to guide a user's interaction with an on-line computer system. The primary principle governing this aspect of USI design is to maintain consistency. Design consistency is emphasized in all published recommendations. With consistent USI design, the user can learn to apply computer tools more quickly, more accurately, and with more confidence.

Design consistency implies predictability of system response to user inputs. A fundamental rule is that some response be received. For every action (input) by the user there should be a noticeable reaction (output) from the machine processor. It is this feedback linking action to reaction that defines each discrete transaction and maintains user orientation in interaction with the system.

The importance of feedback information for guiding the user has been emphasized by Engel and Granda in their recommendations for improving USI design:

Feedback to user action covers keeping the user informed of where he is, what he has done, and whether it was successful. . . . In an interactive computing situation, immediate feedback by the system is important in establishing the user's confidence and satisfaction with the system. One of the more frustrating aspects of any interactive system is sitting at the terminal after entering something and waiting for a response. Questions arise such as, 'Is the system still going?', 'Is the terminal still

connected to the system?', 'Did the computer lose my input?', 'Is the system in a loop?'. A message that indicates that the system is still working on the problem or a signal that appears while the system is processing the user's input provides the user with the necessary assurance that everything is all right.

(1975, page 13)

Predictability of machine response is related to system response time. Timely response can be critical in maintaining user orientation to the task. If a response is received only after a long delay, the user's attention may have wandered. Indeed, the user may forget just which action the machine is responding to. Frequent user actions, generally those involving simple inputs such as function key or menu selections, should be acknowledged immediately. In transactions where output must be deferred pending the results of computer search and/or calculation, the expected delay should be indicated to the user in a quick interim message.

Some experts argue that consistency of system response time may be more important in preserving user orientation than the absolute value of the delay, even suggesting that designers should delay fast responses deliberately in order to make them more consistent with necessarily slow responses. Perhaps such a reduction in response time variability may be desirable. If system response time is always slow, a user can adapt to the situation and find something else useful to do while waiting. But surely a better solution is to make all responses uniformly fast, or, where that is not possible, to signal quickly the occasional slow response as suggested above. In that way, a slow response is made predictable even though it is not consistent with other responses.

As it happens, response time is probably a greater problem in the design of general-purpose, multi-user, time-shared systems than for most dedicated information system applications. Where a system is designed to accomplish defined tasks in a specified manner, data processing loads can usually be anticipated sufficiently well in USI design to provide adequately fast response time for all transactions.

Consistency is important in all aspects of USI design. For data display, formats should be consistent from one frame to another, including always a title at the top, labels to indicate page numbers in related displays, standard labeling of control options, standard positioning of guidance messages, etc. Messages indicating user error should be carefully worded to be both concise and informative, consistently worded from one message to the next, and consistently located in the display format. Even such a subtle

feature as cursor positioning should receive consistent treatment in USI design.

Data entry is guided by consistent treatment of entry fields on the display, including consistent wording of labels, consistent placement of labels with respect to entry fields, and consistent demarcation of the fields themselves. Underlining is frequently recommended for field delineation, displaying clearly to the user where each field is located and also its defined extent. Even more guidance could be provided by consistent use of different field marking to indicate different types of data entry.

For sequence control, consistent USI design is even more important. One design feature that can help guide the user is consistent provision of an OPTIONS display, a "home base" to which the user can return from any point in a transaction sequence in order to select a different transaction. As a general principle, the USI designer should not rely on a user to remember what prior actions have been taken, or even what action is currently being taken. Users may be distracted by competing job demands. For extended transaction sequences, the designer should arrange to display a cumulative record of control actions for user review. For control actions whose consequences are contingent on context, an indication of that context should always be displayed, even when context was defined initially by the user.

Although consistent USI design will provide much inherent guidance to the user, it is often desirable to include in computer-generated display outputs some explicit instructions or reminders to the user. Such instructions should be consistently located in display formatting, perhaps always at the bottom where they can be ignored by experienced users. For inexperienced users it may be desirable to provide supplementary guidance or job instruction, optionally available in response to user request, perhaps through selection of a HELP option. Such optional guidance will help adapt the USI to different user capabilities, supporting the novice user without hindering the expert.

The concern here is with on-line user instruction, as opposed to off-line system documentation. The need for on-line instruction has been emphasized by Brown, Burkleo, Mangelsdorf, Olsen and Williams in their USI guidelines developed as an in-house design standard at Lockheed:

Much of the information commonly provided in paper documentation, such as user manuals, should also be available online. A manual may not be available when it is needed. Some users may never receive the relevant

documentation. They may not know what documents are available, which ones are relevant, or how to procure them. Even users who possess the appropriate documentation will not necessarily have it with them when it is needed.

(1981, page 5-1)

And, one might add to this, even if users have the appropriate documentation on hand, they may not be able to find answers to their questions, especially when the documentation is bulky. Effective on-line instruction and other forms of user guidance can reduce the need for off-line training courses based on system documentation.

Certainly there is a strong trend in current system design toward on-line documentation for user instruction, as computer memory has grown cheaper, and as we have learned more about the techniques of on-line aiding, and as computer use has spread to a more general population of users with little understanding of machine function. The novelty of early pioneering efforts to program computers to teach their own use (Morrill, 1967; Morrill, Goodwin and Smith, 1968; Goodwin, 1974) has now become almost commonplace.

In considering guidelines for design of user guidance functions, we must recognize that user guidance is really a pervasive concern in USI design. Many of the guidelines proposed here for data entry, data display and sequence control functions have the implicit objective of making a system easier to learn and easier to understand during its use. From general recommendations for design consistency, through more detailed guidelines to help distinguish different aspects of data handling transactions, there is an underlying concern for guiding the user's interaction with the automated system. Thus almost any guideline for user guidance could be cross-referenced to a wide range of other design recommendations. Of the many possible cross references, a number of specific interest are cited in the listing that follows.

USER GUIDANCE

Objectives: Consistency of operational procedures
Efficient use of full system capabilities
Minimal memory load on user
Minimal learning time
Flexibility in supporting different users

USER GUIDANCE

General 4.0

-1 Standard Procedures

-1

User interface design should provide standard procedures for accomplishing different types of transactions, to facilitate user learning and efficient system operation.

Comment: Some designers may argue that for any one particular transaction a standard procedure seems inefficient. Perhaps the standard procedure requires one or two more keystrokes than some special procedure that might be devised. But every special feature of interface design will put a small added burden on user memory, and where special procedures are not remembered they may not be used properly. It is in the interest of overall operational efficiency that standard procedures are recommended.

Reference: BB 2.2.1, 6.1.4.

See also: 3.0-16, 3.0-18, 6.0-2, 6.5-6.

-2 Explicit User Actions

-2

User inputs to the computer should result from explicit actions, and not as (possibly unrecognized) side effects of other actions.

Comment: Explicit actions, even though they may require an extra keystroke or two, will help a user to learn procedures and to understand better what is happening in any transaction. In effect, requiring the user to take action to accomplish something can be regarded as a form of guidance.

Comment: An interface designer, with expert knowledge of the system and its internal workings, is sometimes tempted to provide the user with "smart shortcuts", where the computer will execute automatically some action that the user would surely need to take. Incorporating such smart shortcuts in interface design, though done with the intention of helping the user, will risk confusing any but the most expert users.

See also: 1.0-8, 1.7-3, 3.0-5, 3.1.4-14, 3.2-1, 6.0-3.

-3 Relevant Information Displayed

-3

Displays for any transaction should be tailored to the current information requirements of the user.

Comment: When this can be done successfully, so that only relevant data are displayed, the display itself provides implicit guidance, showing what data should be considered. Conversely, display of irrelevant data will tend to confuse the user.

See also: 2.0-1, 2.6-1, 2.6-2, 2.8-1.

-4 Initiating LOG-ON

-4

In applications where users must log on to the system, LOG-ON should be initiated as a separate procedure, accomplished before the user must select among any operational options.

Comment: Separate LOG-ON will focus user attention on the required input(s), without the distraction of having to anticipate other decisions, and will help reduce initial confusion, particularly for novice users.

Comment: Displaying only options that are immediately relevant will help guide users, just as displaying only currently relevant data.

Reference: BB 1.1.

See also: 4.0-3.

-5 Consistent Display Format

-5

Display formats should be designed with a consistent structure evident to the user, so that explicit user guidance information is always presented in the same places and in the same ways.

Example: Display titles should be centered at the top of the display, with display identification codes at the upper left corner. The bottom line of the display should be reserved for command entries, where needed, in which case the line just above it could be used for prompts and advisory messages.

Reference: EG 2.3; MS 5.15.2.2.d.

See also: 1.4-22, 1.4-23, 1.5-4, 2.2-2, 2.5-1, 2.5-5, 3.0-19, 3.1.3-6, 3.1.3-23, 3.1.5-2, 3.4-7, 4.4-8.

-6 Consistent Terminology

-6

Nomenclature for function keys, command names, etc., should be consistent for similar or identical functions in different transaction sequences.

Example: As a negative example, do not call the same function EDIT in one place, MODIFY in another, UPDATE in a third.

Comment: Consistency in interface design is the fundamental basis of effective user guidance.

Reference: EG 4.2.9.

See also: 3.0-21, 3.1.3-9, 3.1.3-11, 3.1.3-25, 3.1.5-5.

-7 Familiar Coding Conventions

-7

Codes and abbreviations for data entry/display should be assigned to conform with conventional usage and general user expectations.

Comment: This practice will aid user learning of codes, and reduce the likelihood of user error in both code generation (entry) and interpretation (display).

Comment: Direct contravention of familiar meanings, such as using an aircraft symbol to denote artillery and vice versa, will obviously lead to user misinterpretation of displayed data.

Reference: BB 7.7.1.

See also: 2.7-7, 2.7-28.

-8 Consistent Coding Conventions

-8

Symbols, and other codes as well, should be assigned to have consistent meanings from one display to another.

Comment: This practice will aid user learning of new codes, so that they will gain familiarity. Where codes have special meanings, those should be defined in the display.

Reference: BB 7.6.2.

See also: 2.7-10, 2.7-14, 2.7-28, 3.1.3-10, 4.4-17.

-9 Display of Guidance Information

-9

In general, the display of information for user guidance should follow recommendations for the design of data displays.

Comment: Some of the specific guidelines for data display are restated for convenient reference in this section, as particularly appropriate for display of user guidance. Many other applicable data display guidelines are cited by cross reference.

See also: Section 2.

-10 Consistent Format for User Guidance

-10

Different types of user guidance should each be formatted consistently in computer-generated displays.

Comment: Categories of user guidance to be considered here include display titles, labeling of data entry fields, prompts for data/command entry, error messages, alarms, status and other advisory messages, as well as on-line instructional material.

Comment: Consistent allocation of particular areas of a display for user guidance may be sufficient. Certain types of guidance, however, such as alarms and error messages, may require auxiliary coding to help attract user attention.

Reference: BB 2.1.2, 6.1.2; EG 2.3; PR 4.5.3.

See also: 1.4-13, 2.5-8, 4.0-5, 4.0-11, 4.3-16.

-11 Distinctive Format for Guidance Material

-11

Displays should be formatted so that user guidance material can be readily distinguished from displayed data.

Comment: Consistent location of user guidance on the display will usually suffice, but other formatting conventions may help distinguish particular categories of user guidance, such as labels, prompts, etc., as recommended in other guidelines.

Reference: BB 2.8.5, 6.1.1; EG 2.1, 2.3, 3.2.3.

See also: 1.4-14, 1.4-15, 1.5-3, 2.1.2-5, 2.5-8, 3.1.3-12, 3.1.4-12.

-12 Distinctive Cursor**-12**

A cursor used for pointing should be designed to be readily distinguished from other items on the display.

Comment: A cursor is the most immediate and continuously available form of user guidance, and should be designed to catch the eye.

See also: 1.1-1.

-13 Control Labeling**-13**

Data/command entry keys and other controls should be clearly labeled to indicate their function.

See also: 1.0-9, 3.1.4-4.

-14 Data Labeling**-14**

All displayed data items, fields and groups should be clearly labeled.

Comment: The user should not have to rely on contextual cues to identify displayed data. Individual field labels can be omitted only where display format and labeling of grouped data clearly identify subordinate items, as in row/column labeling of tabular data.

Reference: MS 5.15.4.9.1.

See also: 1.4-5, 1.4-16, 1.4-18 thru -20, 1.5-2, 2.1.2-1.

-15 Highlighting**-15**

Techniques for highlighting critical items in data display should also be considered for adding emphasis to the display of critical user guidance information.

Reference: EG 2.1.

See also: 2.7-1.

-16 Familiar Terminology

-16

Labels, prompts and user guidance messages should be worded clearly, using terminology familiar to the users.

Example: (Good) Data requires special access code; call Data Base Admin, X 9999.

(Bad) IMS/VS DBMS private data; see DBSA, 0/99-99.

Comment: User testing and iterative design will often be needed to eliminate difficult words, abbreviations and acronyms that are not generally familiar to all users.

Reference: BB 3.7.1, 3.7.4; EG 3.4.5, 4.2.12; PR 2.4.

See also: 2.0-3, 2.1.1-18, 2.1.1-20, 3.1.5-3.

-17 Task-Oriented Terminology

-17

Labels, prompts and user guidance messages should be task-oriented, incorporating special terms and technical jargon related to the users' tasks.

Comment: Jargon terms may be helpful, if they represent the jargon of the user and not of the designer or programmer. The rule here should be to know the users and adapt interface design to their vocabulary instead of forcing them to learn yours.

Reference: BB 3.7.2; EG 4.2.13; PR 2.4.

See also: 1.4-21, 2.1.1-18, 2.1.1-19, 3.1.5-3, 3.1.5-5, 3.2-10.

-18 Affirmative Statements**-18**

User guidance messages should be worded as affirmative statements, rather than negative statements.

Example: (Good) Clear the screen before entering data.

(Bad) Do not enter data before clearing the screen.

Comment: Affirmative statements are easier to understand. Tell the user what to do, rather than what to avoid.

Reference: BB 3.8.3.

See also: 2.1.1-10.

-19 Active Voice**-19**

User guidance should be worded in active rather than passive voice.

Example: (Good) Clear the screen by pressing RESET.

(Bad) The screen is cleared by pressing RESET.

Comment: Sentences in active voice are easier to understand.

Editorial Comment: Some writers of guidelines may not follow their own advice in this regard.

Reference: BB 3.8.5.

See also: 2.1.1-11.

-20 Temporal Sequence

-20

When user guidance describes a sequence of steps, the wording should follow that temporal sequence.

Example: (Good) Enter LOG-ON sequence before running programs.

(Bad) Before running programs, enter LOG-ON sequence.

Reference: BB 3.8.6.

See also: 2.1.1-12.

-21 Consistent Grammatical Structure

-21

User guidance should be worded consistently in terms of grammatical construction.

Example: (Good)

(Bad)

Options:

s = Select data

e = Erase display

w = Write file

Options:

s = Select data

d = Display erasure function

w = Write file

Comment: Even minor inconsistencies can distract a user, and delay comprehension as the user wonders momentarily whether some apparent difference represents a real difference.

Reference: HB 3.8.4.

See also: 2.1.2-4, 3.1.3-7.

-22 Flexible User Guidance

-22

When techniques adopted for user guidance (display of option lists, command prompting, etc.) may slow an experienced user, provision should be made for alternative paths/modes to by-pass standard guidance procedures.

Comment: Multiple paths, such as command entry to by-pass a menu, or use of abbreviated rather than complete commands, can speed the performance of an experienced user. The interface designer, however, should take care that such shortcuts supplement rather than supplant the standard, fully guided procedures provided for novice users.

Reference: BB 4.5.

See also: 3.0-2, 4.4-26, 4.4-27.

-1 Status Information

-1

Some indication of system status should be available to a user at all times.

Comment: In some applications, system status may be continuously displayed. Status display can be explicit (e.g., by message), or can be implicit (e.g., by a displayed clock whose regular time change offers assurance that the computer link is still operating). Alternatively, system status information might be provided only on user request, following a general or specific query.

Comment: Status information is particularly needed, of course, when system operation is unreliable for any reason. Under those conditions, if status information is not provided by design, users will often devise their own repertoire of harmless but time wasting test inputs to check system performance.

Reference: BB 4.3; MS 5.15.1.4, 5.15.1.4.b.

-2 Automatic LOG-ON Display

-2

In applications where users must log on to the system, a LOG-ON display with appropriate guidance should appear automatically at a user's terminal.

Comment: An automatic LOG-ON display will signal the operational availability of a terminal, as well as prompting the user to make necessary initial inputs. The user should not have to take any special action to obtain the LOG-ON display itself, other than turning on a terminal.

Reference: BB 1.1; KG 4.2.6.

-3 LOG-ON Delay

-3

When users must log on to a system, if LOG-ON is denied then an advisory message should be displayed to tell a user what the system status is and when the system will become available.

Example: System is down for maintenance until 9:30 AM.

Comment: Avoid "as soon as possible" messages. Make an estimate of system availability, and update the estimate if that becomes necessary.

Reference: BB 4.3.3.

-4 Keyboard Lock

-4

If at any time the keyboard is locked, or the terminal is otherwise disabled, that condition should be signaled by disappearance of the cursor from the display and (especially if infrequent) by some more specific indicator such as an auditory signal.

See also: 3.0-10.

-5 Current Users

-5

In applications where task performance requires data exchange and/or interaction with other users, status information should be available concerning other current users of the system.

See also: 5.3-2.

-6 Current Load

-6

In applications where task performance is affected by operational load (e.g., number of on-line users), status information should be available indicating current load and/or current system performance.

Comment: Such load information is primarily helpful, of course, when system use is optional, i.e., when a user can choose to defer work until low-load periods. But load status information may help in any case by establishing realistic user expectations for system performance.

-7 External Status**-7**

In applications where task performance requires data exchange and/or interaction with other systems, relevant status information for external systems should be available to the user.

See also: 5.3-2.

-8 Date-Time Signals**-8**

In applications where task performance requires or implies the need to assess currency of information, date-time signals should be available to users as an annotation on displays.

Comment: Date-time status might be displayed continuously or periodically, as on displays that are automatically updated, or by user request, depending on the application.

Comment: In some applications, request for date-time display can provide an innocuous means for a user to check on general system response.

-9 Alarm Status**-9**

In applications where alarm signals are established on the basis of logic defined by users, status information should be available concerning the current status of alarm settings, in terms dimensions/variables covered and values/categories established as critical.

Comment: Alarm status information will be particularly helpful in monitoring situations where responsibility may be shifted from one user to another.

See also: 3.6-1.

-1 Consistent Feedback

-1

Every input by a user should consistently produce some perceptible response output from the computer.

Comment: Keyed entries should appear immediately on the display. Function key activation or command entries should be acknowledged either by evident performance of the requested action, or else by an advisory message indicating an action in process or accomplished. Unrecognized inputs should be acknowledged by an error message.

Comment: Absence of system response is not an effective means of signaling acceptable entry. At best, a dialogue without feedback will be disconcerting to the user, as when we talk to an unresponsive human listener. At worst, the user may suspect system failure, with consequent disruption and/or termination of the interaction sequence.

Reference: BB 1.2, 4.3.2; EG 3.3.2, 4.2.5, 6.3.7;
MS 5.15.3.2, 5.15.3.4.1.

See also: 1.0-2, 3.0-12 thru -15, 3.1.3-26, 3.1.4-15.

-2 Rapid Feedback

-2

Computer response to user entries should be rapid, with consistent timing as appropriate to different types of transactions.

See also: 1.1-5, 2.4-1, 3.0-8, 3.1-2.

-3 Feedback on Processing Status

-3

When computer processing of a user entry is delayed, the user should be informed in an advisory message when processing is completed, with appropriate guidance for further user actions.

Comment: For long delays, interim feedback on processing status before completion may be reassuring to a user.

Reference: MS 5.15.1.4.c.

See also: 3.0-11.

-4 Display Identification

-4

Each display page should have a unique identification prominently displayed in a consistent location at the top.

Comment: A displayed title may suffice, although a shorter identification code may be helpful for some purposes. The objective is to help the user recognize a display when it appears, to learn interactive sequences stepping from one display to another, and (in some system applications) to request a particular display directly. Display identification will also help both users and interface designers to refer to individual displays in discussion and documentation.

Comment: In applications involving menu selection, it may prove helpful to code each display with the concatenated string of option selections (letter codes) used to reach that display. This practice is particularly useful in situations where a user can learn to by-pass the menu selection sequence by entering option string codes as a single command to request a familiar data display.

Reference: BB 2.2.3; PR 4.5.2.

See also: 2.2-1, 2.2-2, 2.5-7.

-5 Incomplete (Multi-Page) Displays**-5**

When lists or data tables extend beyond the capacity of a single display frame, the user should be informed that the display is not complete.

Example: Incomplete lists might be annotated at the bottom as CONTINUED ON NEXT PAGE.

Example: For extended data tables, the display title should be annotated PAGE ____ OF ____.

Example: For scrolled material, displays should be annotated with the current and concluding locations, LINE ____ OF ____.

Comment: As a complementary recommendation, it may also be desirable to conclude completed lists with the annotation END OF LIST, unless the list is so short that it obviously does not fill available display space.

Reference: BB 2.9.6; EG 3.4.1; PR 4.5.5.

See also: 2.8-5, 2.8-6.

-6 Feedback for Printout Requests**-6**

When user requests for printed output will be handled by a remote printer, the user should receive an advisory message confirming that a printout request has been processed.

Reference: EG 4.2.14.

-7 Transaction Status

-7

Some indication of transaction status should be provided to the user automatically whenever the complete computer response to a user entry will be delayed.

Comment: After making an entry to the computer, the user needs feedback to know whether that entry is being processed properly. Delays in computer response longer than a few seconds can be disturbing to the user, especially for a transaction that is usually processed immediately. In such a case some intermediate feedback should be provided, perhaps as an advisory message that processing has been initiated, and ideally with an estimate of how long it will take to complete.

Reference: BB 4.3.1; EG 4.2.5; MS 5.15.1.4.b.

See also: 3.0-12.

-8 Feedback for Mode Selection

-8

When a user (or computer) action establishes a change in operational mode, which will affect subsequent user actions, some continuing indication of mode selection should be displayed to the user.

Example: Selection of a DELETE mode in text editing.

Comment: This practice is particularly helpful when the mode selected is one seldom used.

Comment: Display of mode selection will help prevent unintended data loss when the mode is potentially destructive (e.g., DELETE). For destructive modes, it may help if the mode indication is implemented as some sort of distinctive change in the appearance of the cursor, since the cursor is the display feature most surely seen by a user.

See also: 3.4-4, 4.4-10, 6.0-4, 6.5-13.

-9 Feedback for Option Selection

-9

When previously selected options are still operative, they should be automatically recapitulated on the display, or else be displayable on user request.

Comment: Such an "audit trail" of option selections will help a novice user learn transaction sequences, and may help any user deal with complex transactions.

Reference: EG 3.4.

See also: 4.4-10, 4.4-18.

-10 Feedback for Item Selection

-10

When a user selects any displayed item in order to perform some operation on it, that item should be highlighted on the display.

Comment: This practice will provide a routine natural feedback that item selection has been accomplished, and will provide a continuing reminder to the user of just what selection has been made.

Comment: For a selection among displayed options, the selected option might be brightened, or the non-selected options might be dimmed.

Reference: EG 2.1.1, 3.1, 3.1.1; MS 5.15.1.4.a.

See also: 1.1-5, 3.1.3-26, 3.4-6.

-11 Feedback for User Interrupt

-11

Following user interrupt of data processing, an advisory message should be displayed assuring the user that the system has returned to its previous status.

Reference: BB 1.8.

See also: 3.3-6.

-1 Informative Error Messages

-1

When the computer detects an entry error, an error message should be displayed to the user stating what is wrong and what can be done about it.

Example: (Good) Unrecognized code format; enter two letters, then three digits.

(Bad) Invalid input.

Comment: The user should never have to search through reference information to translate error messages.

Reference: BB 1.4.2; EG 3.3.1; MS 5.15.1.2.5, 5.15.1.4.d; PR 4.12.1.

-2 Specific Error Messages

-2

Error messages should be worded as specifically as possible, based on computer analysis of data handling transactions.

Example: (Good) No record for Loan 6342; check number.

(Bad) No record for inquiry.

Reference: MS 5.15.1.2.5; PR 4.12.5.1.

-3 Task-Oriented Error Messages

-3

The wording of error messages should be appropriate to a user's task and level of knowledge.

Example: (Good) Unrecognized contract number; check file and enter a current number.

(Bad) Entry blocked. Status Flag 4.

Comment: Error messages that can be understood only by experienced interface designers and programmers may have no value for ordinary users.

Reference: BB 1.4.5, 6.1.5; EG 3.3.7; MS 5.15.1.2.5.

See also: 2.1.1-18, 4.0-17.

-4 Error Message Indicates Correct Entries

-4

When a data entry or (more often) a control entry must be made from a small set of alternatives, those correct alternatives should be indicated in the error message displayed in response to a wrong entry.

See also: 3.2-5, 4.4-1.

-5 Brief Error Messages

-5

Error messages should be brief, consistent with being informative.

Example: (Good) Entry must be a number.

(Bad) Alphabetic entries are not acceptable because this entry will be processed automatically.

Comment: Often a user will recognize that an error has been made, and the message will serve merely as a confirming reminder. In such instances, short error messages will be scanned and recognized more quickly.

Comment: For a user who is truly puzzled, and who needs more information than a short error message can provide, auxiliary HELP can be provided either on-line or by reference to system documentation.

Comment: Some authorities recommend starting each error message with an identifying code, to facilitate rapid recognition of error messages. That practice might help experienced users, who would come to recognize the codes, and could be efficient in referring users to correspondingly coded parts of system documentation if on-line explanation (HELP) is not available.

Reference: BB 1.4.1, 1.4.4, 3.8.1; EG 3.1.3, 3.3, 3.3.7; PR 2.2, 4.1.2.2.

See also: 2.1.1-8, 2.1.1-9, 4.4-19.

-6 Neutral Wording for Error Messages

-6

Error messages should be stated in neutral wording, without implications of blame to the user, without personalization of the computer, and without attempts at humor.

Example: (Good) Entry must be a number.

(Bad) Illegal entry.

(Bad) I need some digits.

(Bad) Don't be dumber, use a number.

Comment: Error messages should reflect a consistent view that the computer is a tool, with certain limitations that a user must take into account in order to make the tool work properly. If error messages reflect an attitude that the computer (or its programmer) imposes rules, or establishes "legality", the user may feel resentful. If error messages reflect personalization of the computer, as if it were a friendly colleague, a naive user may be misled to expect human abilities the machine does not actually possess. If error messages are worded humorously, any joke will surely wear thin with repetition, and come to seem an intrusion on a user's concern with efficient task performance.

Comment: The same considerations apply for the wording of computer-generated prompts and other instructional material.

Reference: BB 1.4.3; EG 3.3.8, 5.3; PR 2.2.

-7 Layered Error Messages

-7

Following the output of simple error messages, the user should have the option of requesting more detailed explanation for errors.

Comment: A more complete discussion of each error could be made available on-line, perhaps at several levels (or "layers") of increasing detail, supplemented by reference to off-line system documentation if necessary. Successively deeper levels of explanation could then be provided in response to repeated user requests for HELP.

Reference: BB 1.6, 5.3; EG 3.3.

See also: 4.4-23.

-8 Multiple Error Messages

-8

When multiple errors are detected in a combined user entry, some indication should be given to the user, even though complete messages for all errors cannot be displayed together.

Example: * DATE should be numeric. [+ 2 other errors]

Comment: Cursor placement should be in the data field referred to by the displayed error message, with other error fields highlighted in some way (e.g., by reverse video).

Reference: PR 4.12.3.

See also: 4.3-17.

-9 Repeated Errors

-9

When an entry error is repeated, some noticeable change in the displayed error message should be provided.

Example: Some designers go to the extent of providing two versions of each error message, alternating on the display in response to repeated errors.

Comment: If an error message is repeated identically, so that displayed feedback seems unchanged, the user may be uncertain whether the computer has processed the revised entry. A simple expedient might be to display the same verbal message but with changing annotation, perhaps marked with one asterisk or two.

-10 Non-Disruptive Error Messages

-10

Error messages should be output after a user's entry has been completed.

Example: An error message should not be generated as wrong data are keyed, but only after an explicit ENTER action has been taken.

Comment: In general, the display of error messages should be timed so as to minimize disruption of the user's thought process and task performance.

Reference: EG 7.1.

See also: 1.7-5.

-11 Response Time for Error Messages

-11

An error message should be displayed approximately two seconds after the user entry in which the error is detected.

Comment: Longer delays in error feedback may cause user uncertainty and/or confusion. Longer delays will cause frustration when the user already knows an error was made, which is often the case.

Comment: Shorter delays in error feedback can pose problems of a different sort. An error message following immediately upon a user entry can be disconcerting, as the user needs a few moments to shift gears. Immediate error feedback can also be irritating. User expectations are conditioned by human dialogues, where an immediate contradiction is considered rude, and where a polite listener will at least pretend to think for a few moments before saying that you are wrong.

Reference: EG Table 2; Miller, 1968.

-12 Documenting Error Messages

-12

System documentation should include, as a supplement to on-line guidance, a listing and explanation of all error messages.

Comment: Documentation of error messages will facilitate review of that aspect of user interface design, since it is difficult to generate all possible error messages by actually making errors in on-line transactions.

Comment: Documentation of error messages will permit users to reference particular messages for fuller explanation, and will permit user review of all messages as a means of understanding data processing requirements and limitations.

Comment: Developing good error messages may require iterative review by designers, users, and human factors specialists.

Reference: BB 1.12, 2.2.4.

-13 Cursor Placement Following Error

-13

When an error has been detected in a data/command entry, in addition to the display of an error message the cursor should be automatically positioned at the point of error (data field or command word).

Comment: Display of the cursor at a non-routine position will help emphasize that an error has occurred, and help direct the user's attention to the faulty entry.

Reference: PR 4.12.1.

See also: 4.4-12.

-14 Selective Error Correction

-14

Following error detection, users should be prompted to re-enter only the portion of a data/command entry that is not correct.

Comment: The user should not have to re-key an entire command string or data set just to correct one wrong item.

Reference: EG 4.2.3.

See also: 3.1.5-17, 3.5-3, 6.0-6, 6.3-12.

-15 Cautionary Messages

-15

When a user entry can be recognized as doubtful, in terms of defined data/command validation logic, a cautionary message should be displayed asking the user to confirm that entry.

Example: Blood pH of 6.6 is outside the normal range;
confirm or change entry.

Comment: Feedback to the user can be worded to deal with a range of intermediate categories between a seemingly correct entry and an outright error.

See also: 3.5-10, 3.5-12, 4.3-16.

-16 User Confirmation of Destructive Entries

-16

A user should be asked to take explicit action to confirm potentially destructive data/command entries before they are accepted by the computer for execution.

Comment: A requirement to take an explicit CONFIRM action will direct user attention to questionable entries and help the user avoid the consequences of thoughtless errors.

Reference: BB 1.10; EG 4.2.8.

See also: 3.0-3, 3.5-9, 4.3-15, 6.0-8, 6.3-20, 6.5-14, 6.5-18.

-17 Alarm Coding

-17

For conditions requiring (or implying the need for) special user attention, distinctive coding should be used for alarms (or warning messages).

Example: Alarm messages might be marked with a blinking symbol and/or displayed in red, and be accompanied by an auditory signal. Warnings and error messages might be marked with a different special symbol and/or displayed in yellow.

Comment: This practice will help ensure appropriate attention, even when the user is busy at routine tasks.

Reference: BB 7.7.2, 7.7.3; EG 2.1.3; MS 5.15.2.10.b, 5.15.4.6.1.a.

See also: 2.7-11, 2.7-28, 2.7-30, 2.7-34, 3.0-3, 3.6-2, 6.0-8, 6.5-17.

-1 Display of Guidance Information

-1

At any point in a transaction sequence specific user guidance information should be available for display.

Comment: Do not require a user to remember information not currently displayed. The user should not have to remember what actions are available, or what action to take next. Human memory is unreliable, and without guidance the user can be expected to make errors.

Reference: BB 4.3.4; EG 3.4.4; MS 5.15.2.8.

See also: 2.0-2, 2.6-1, 2.8-1, 3.1.3-17, 3.1.3-18, 3.2-4, 3.2-5.

-2 General Menu of Control Options

-2

A general menu of control options should always be available for user selection, to serve as a "home base" or consistent starting point at the beginning of a transaction sequence.

See also: 3.2-2, 3.2-3.

-3 Logical Menu Structure

-3

Menu options should be grouped logically to aid user learning and selection among displayed alternatives.

See also: 3.1.3-14, 3.2-3.

-4 Hierarchic Menus

-4

When hierarchic menus must be used, they should be organized and labeled to guide the user within the hierarchic structure.

Comment: Users will learn menus more quickly if a map of the menu structure is provided as HELP.

Reference: Billingsley, 1982.

See also: 3.1.3-16, 3.1.3-19, 3.1.3-22.

-5 Guidance for Sequence Control

-5

At any point in a transaction sequence there should be displayed guidance telling the user how to continue.

Example: (Good) Data base status is current through March 1983. Press STEP key to continue.

(Bad) Data base status is current through March 1983.

Reference: BB 4.2; EG 3.1.2; PR 2.2.

See also: 3.0-6, 3.1.3-8, 3.2-13.

-6 Implicit Options

-6

Control options that are generally available at any step in a transaction sequence should be treated as implicit options, i.e., need not be included in a display of step-specific options.

Comment: The user may be expected to remember continuously available options, once they have been learned, without their specific inclusion in a display of guidance information. Perhaps the best design expedient is to implement implicit options on appropriately labeled function keys, which will aid user learning and provide a continuing reminder of their availability.

See also: 3.2-6.

-7 Prompting User Entries

-7

The computer should be programmed to provide prompting, i.e., to display advisory messages to guide users in entering required data and/or command parameters.

Comment: Prompting in advance of data/command entry will help reduce errors, particularly for inexperienced users.

Comment: If a default value has been defined for null entry, that value should be included in the prompting information.

Reference: EG 4.2.2, 4.2.4; MS 5.15.1.2.7, 5.15.3.5; PR 4.9.2.

See also: 1.7-7, 3.1.5-10, 3.2-12, 3.2-18, 3.5-5.

-8 Display Location for Prompting -8

Prompts for command entry should be displayed next to the command entry area, at the bottom of the display.

See also: 4.0-5.

-9 Prompting by User Request -9

When users vary in experience, which is often the case, prompting should be an optional guidance feature that can be selected by novice users but can be omitted by experienced users.

Comment: Flexibility in prompting can also be provided by multi-level HELP options, so that additional guidance information can be obtained if the standard prompt is not adequate.

See also: 3.1.5-10, 3.1.5-11, 4.4-23.

-10 Displayed Context -10

When the results of a user entry are contingent upon context established by previous entries, some indication of that context should be displayed to the user.

Example: Selection of operational modes.

Example: If the user is editing a data file, both the file name and an indication of EDIT mode should be displayed.

Reference: EG 4.2.1; MS 5.15.1.2.7.a, 5.15.3.5.b.

See also: 2.9-6, 2.9-7, 3.0-17, 3.1.4-6, 3.1.4-11, 3.4-1, 3.4-4, 3.4-5, 4.2-8.

-11 Implicit Prompts for Data Entry

-11

Implicit cues for data entry should be provided by consistent and distinctive formatting of data fields.

Example: A colon could be used consistently to indicate that an entry can be made, followed by an underscored data field to indicate item size, such as

Enter part code:

or perhaps just simply

PART CODE:

Comment: Consistent use of implicit prompting cues can sometimes provide sufficient guidance to eliminate the need for more explicit advisory messages.

Reference: BB 6.5; EG 6.3.1.

See also: 1.4-8 thru -10, 1.4-17.

-12 Automatic Cursor Positioning

-12

Following computer generation of display output, the cursor should automatically be positioned on the display in a location consistent with the type of transaction.

Example: For data entry displays, the cursor should be placed initially at the first data field, or else at the first wrong entry if an error has been detected. In other displays, the cursor should be placed at a consistent HOME position, or at the first control option for menu selection, or else in a general command entry area, depending upon the type of display.

Comment: Consistent cursor positioning will provide an implicit cue for user guidance.

Reference: EG 4.2.3; PR 3.3.3.

See also: 1.1-6, 1.1-19, 1.4-25, 3.2-7, 3.2-8, 4.3-13.

-13 On-Line System Guidance

-13

Reference material should be available for on-line display to the user describing system capabilities and procedures.

Comment: Many systems are not utilized effectively, by experienced users as well as by novices, because users do not fully understand system capabilities. On-line access to a description of system structure, components and options will aid user understanding.

Comment: On-line guidance can supplement or in some instances substitute for off-line training. An investment in designing user aids may be repaid by reduced costs of formal training as well as by improved operational performance.

Reference: BB 5.1, 5.2.

See also: 4.4-14, 4.4-15.

-14 Data Index

-14

In applications where the user has selective access to stored data, the computer should provide an on-line data index to help guide user selection.

Comment: The data index should indicate file names and structure as needed to access different categories of data.

See also: 3.1.6-3, 4.4-13.

-15 Command Index

-15

In applications where a user may employ command entry, the computer should provide an on-line command index to help guide user selection and composition of commands.

Comment: Such a command index may help the user to phrase a particular command, but will be more generally useful as a reference for discovering related commands and learning the overall command language.

See also: 4.4-13.

-16 Dictionary of Abbreviations

-16

A complete dictionary of abbreviations used for data entry, data display, and command entry, should be available for on-line user reference and in system documentation.

Comment: In applications where users can create their own abbreviations, as in the naming of command "macros", it will be helpful to provide aids for users to create their own individual on-line dictionaries.

Reference: BB 3.1.3.

See also: 2.1.1-25.

-17 Definition of Display Codes

-17

When codes are assigned special meaning in a display, a definition should be provided at the bottom of the display.

Comment: This practice will aid user assimilation of information, especially for display codes that are not already familiar.

Reference: BB 7.6.1.

See also: 2.7-8.

-18 Transaction Records

-18

In system applications where it is warranted, the user should be able to request a displayed record of past transactions in order to review prior actions.

Reference: EG 4.2.7.

See also: 3.4-2, 4.5-3.

-19 Optional HELP

-19

In addition to explicit aids (labels, prompts, advisory messages) and implicit aids (cueing) provided in user interface design, there should also be a capability for a user to request further on-line guidance by a request for HELP.

Comment: It is difficult for an interface designer to anticipate the degree of prompting that may be required to guide all users. Moreover, even when prompting needs are known, it may be difficult to fit all needed guidance information on a working display. A supplementary HELP display can be provided to deal with such situations.

Reference: BB 5.3; MS 5.15.1.4.5; PR 3.3.15.

-20 Standard HELP Request

-20

The HELP request should be initiated consistently by some simple, distinctive user action.

Example: HELP could be requested by an appropriately labeled function key, or perhaps by keying a question mark.

Comment: The user should be able to request HELP at any point in a transaction sequence. It will help more if the procedure is always the same, whether the user wants an explanation of a particular data entry, or displayed data, or command option.

Reference: BB 5.3.

-21 Task-Oriented HELP

-21

Computer response to HELP request should be tailored to task context and the current transaction.

Example: If a data entry error has just been made, HELP should display information concerning correct entry requirements for that particular data item.

Example: If an error in command entry has just been made, HELP should display information concerning that command, its function, its proper structure and wording, required and optional parameters, etc.

-22 Defining HELP Requests

-22

When a request for HELP is ambiguous in context, the computer should initiate a dialogue in which the user can specify what data, message or command requires explanation.

Reference: BB 5.3.

-23 Multi-Level HELP

-23

When an initial HELP display provides only summary information, more detailed explanations should be available in response to repeated user requests for HELP.

Comment: It is necessarily a matter of judgment just what information should be provided in response to a HELP request. Designing the HELP function to provide different levels of increasing detail permits users to exercise some judgment themselves as to just how much information they want.

Reference: BB 1.6, 5.3.

See also: 4.3-7.

-24 Browsing HELP

-24

Novice users should be able to browse on-line HELP displays, just like a printed manual, to gain familiarity with system functions and operating procedures.

Reference: Cohill and Williges, 1982.

-25 On-Line Training**-25**

For many system applications, an on-line training capability should be provided to introduce new users to system capabilities and to permit simulated "hands on" experience in data handling tasks.

Comment: On-line simulation, using the same hardware, user interface software and data processing logic as for the real job, can prove an efficient means of user training. Care must be taken, however, to separate and distinguish simulated from actual system operation.

Reference: BB 5.4.

See also: 6.2-2, 6.3-4, 6.5-4.

-26 Flexible Training**-26**

On-line training capabilities should be adapted to the needs of different users.

Example: Instruction on keyboard use and lightpen selection of menu options might be provided for novice users, while a tutorial on command language might be provided at several levels for more experienced users.

Example: In systems supporting different user jobs, on-line instruction might describe the procedures for each different data handling task.

See also: 3.0-4, 3.1-1, 3.1.3-27, 3.1.5-9, 4.0-22.

-27 Adaptive Training**-27**

In applications where complex user skills must be developed, computer-mediated training should adapt automatically to current user skill levels.

Example: On-line tracking and other skilled control tasks.

Comment: Adaptive training will require some means for computer assessment of appropriate components of user performance.

See also: 4.0-22, 4.5-1.

-1 User Performance Measurement

-1

In applications where skilled user performance is critical to system operation, provision should be made for automatic computer recording and assessment of appropriate skills.

See also: 4.4-27.

-2 User Awareness

-2

Users should be informed concerning any records kept of individual performance.

Comment: Informing the user concerning the nature and purpose of performance records is required by ethical principle, and in some situations may be required by law.

-3 Transaction Records

-3

In applications where it is warranted, the computer should be programmed to maintain records of user transactions.

Comment: Transaction recording could be made optional, under control of a system manager.

Comment: Record keeping might include duration, sequencing and frequency of different transactions.

Comment: Transaction records will aid task analysis, particularly in developing systems where data handling requirements are not yet fully defined.

Comment: Buffered store of current transaction sequences may be required for user guidance.

See also: 4.4-18.

-4 Data Access Records**-4**

In applications where it is warranted, the computer should be programmed to maintain records of data access, i.e., which data files, categories, items have been called out for display.

Comment: Records of data use may help software designers improve file structure, reduce data access time, and manage multiple use of shared data files.

Comment: Data access records may also be required for purposes of data protection/security.

See also: 6.2-7.

-5 Program Use Records**-5**

In applications where it is warranted, the computer should be programmed to maintain records of use for different portions of operational software.

Comment: In many cases "program calls" can be derived from transaction records rather than measured directly. Records of software use may not affect user interface design directly, but can help detect and correct programming inefficiencies and improve system response, particularly during early stages of system development.

-6 Error Records**-6**

The system should provide a capability for recording user errors.

Comment: Error recording might be done continuously, or by periodic sampling, under the control of system supervisors.

Comment: Error records can be used to indicate supplemental instruction needed by different users, if individual user errors are identified. In that case, users should be informed that such records will be kept.

Comment: Error records can be used to indicate whether particular transactions are giving trouble to many users, in which case design improvements to the user interface may be needed, including changes to user guidance.

Reference: BB 1.11.

See also: 4.5-2, 4.6-1.

-7 HELP Records**-7**

The system should provide a capability for recording user requests for HELP.

Exception: There is probably no need to record user browsing of HELP messages, if such a capability is provided.

Comment: HELP records can be used to detect deficiencies in user interface design in early system development, and can be used to improve user guidance in later system operation.

Comment: In effect, user requests for HELP might be regarded as a possible symptom of poor interface design. If HELP requests are frequent for a particular transaction, then some design improvement may be needed, in procedures, or prompting for user guidance, or both.

See also: 4.4-19, 4.4-24.

-1 Flexible Design for User Guidance

-1

When user guidance requirements may change, which is often the case, some means should be provided for the user (or an authorized supervisor) to make necessary changes to user guidance functions.

Comment: User guidance functions that may need to be changed include those represented in these guidelines, namely, changes in status information, routine and error feedback, job ads and user records.

SECTION 5

DATA TRANSMISSION

Preceding sections of this report have dealt with fundamental functions in using on-line information systems -- putting data into a computer, getting data from a computer, controlling the sequence of input-output transactions, with guidance throughout the process. What other functions can a computer serve? One area that demands attention is the use of computers for communication, i.e., to mediate the transmission of data from one person to another.

In considering data transmission functions, we must adopt a broad perspective. The data that are transmitted via computer may include words and pictures as well as numbers. And the general process of data transmission may cover a wide range of applications.

In some applications, computer-mediated data transmission may be a discrete, task-defined activity, as when a system used for planning/scheduling is later used to generate and transmit the orders to implement a plan. In other applications, data transmission may be a continuing, intermittent activity, as when air traffic controllers use their computer facilities to exchange information (and to handover responsibility) while monitoring flight operations.

An even broader reliance on computer-based message handling can be seen in systems whose explicit, primary purpose is to support communication. In such applications, computer-mediated data transmission is now sometimes called "electronic mail". In conjunction with other new technology, the current development of electronic mail has led to forecasts of a "wired society" in which we rely increasingly on computers for communication (Martin, 1978).

Effective communication can be of critical importance in applications where coordinated data processing by groups of people is required. This will be true whether communication is mediated by computer or by other means. Computer-based message handling offers a potential means of improving communication efficiency, but careful design of the user interface will be needed to realize that potential.

In the interests of efficiency, much data transmission among computers is designed to be automatic, representing a programmed message exchange between one computer and another, with no direct user involvement. When a user does not participate in data transmission, then there is of course no need to include data

transmission functions in user interface design. It is only when users must undertake data transmission transactions that USI design guidelines will be needed.

For the users of computer systems, data transmission implies an extra dimension of complexity. A user not only must keep track of transactions with the computer, but must exchange data with other people as well. Data exchange may be with other users of the same system, and/or with users of other systems. Data exchange may be immediate, with other currently active users; or data exchange may be deferred until other users come on-line, and so extend over a period of time.

Computer users will need extra information to control data transmission, perhaps including status information about other systems, and the communication links with other systems. Users will need feedback when sending or receiving data. Users may need special computer assistance in composing, storing and retrieving messages, as well as in actual data transmission.

The general objectives of user interface design in other functional areas will be equally valid for data transmission functions. Procedures for data transmission should be consistent in themselves, and should be compatible with procedures for data entry and display. Interface design should minimize effort and memory load on the user, and permit flexibility in user control of data transmission.

Recent studies of computer-based message handling have been chiefly concerned with determining the functional capabilities required in communication system applications (cf., Goodwin, 1980). There is already evidence that the practical use of data transmission functions can be limited by deficiencies in the user interface (Goodwin, 1982), but no accepted guidelines for USI design are presently available. Some tentative guidelines are proposed in the following pages, but with little reference to supporting data or prior recommendations.

DATA TRANSMISSION

Objectives: Consistency of data transmission
Minimal user actions
Minimal memory load on user
Compatibility with other data handling
Flexibility for user control of data transmission

DATA TRANSMISSION

General 5.0

-1 Consistent Procedures -1

Procedures for data transmission, i.e., for sending data or for receiving data, should be consistent from one transaction to another.

-2 Minimal User Actions -2

Design of data transmission procedures should minimize user actions required.

Example: Automatic formatting of messages derived from data already stored in the computer.

Example: Automatic reformatting of stored data for transmission, where format change is required.

Example: Automatic queuing of outgoing messages pending completion of transmission, and incoming messages pending review and disposition.

-3 Minimal Memory Load on User -3

Data transmission procedures should be designed to minimize memory load on the user.

Example: Automatic provision of standard header information, distribution lists, etc.

Example: Automatic record keeping, message logging, status displays, etc.

-4 Compatible Procedures for Message Sending -4

Procedures for sending data, i.e., for composing messages, should be compatible with procedures for general data entry.

-5 Compatible Procedures for Message Receipt -5

Procedures for receiving data, i.e., for review of incoming messages, should be compatible with procedures for general data display.

-6 Flexible User Control -6

Flexible user control of data transmission should be provided, so that the user can decide what data should be transmitted, when, and where.

Exception: In monitoring and control applications where data processing and transmission must be event-driven.

-7 Explicit User Action -7

Data transmission for both sending and receiving should be accomplished by explicit user action.

Comment: Automatic message generation and receipt will be helpful in many applications, but in such cases the user should participate by establishing, reviewing and/or changing the computer logic controlling automatic data transmission.

See also: 1.0-8, 3.0-5, 4.0-2, 6.0-3.

-1 Formatted Text

-1

When transmitted text must be formatted in a particular way, format control should be automatic, with no extra attention required from the user.

Example: Defined message formats filled automatically from stored data.

Example: Automatic header/paging formatting in document transmission.

Comment: A user should not have to transpose data for transmission in a form different than that used originally for data entry.

See also: 5.0-2, 5.4-3.

-2 Unformatted Text

-2

The user should be able to compose and transmit messages as unformatted text.

Comment: Arbitrarily created text messages (sometimes called "chatter") will let users deal flexibly with a variety of communication needs not anticipated by system designers.

-3 Data Forms

-3

In transmission of data forms, the user should be able to enter, review and change data in an organized display with field labeling, rather than as an unlabeled string of items.

Comment: User composition and review of unlabeled data strings, especially those requiring delimiters to mark items, will be prone to error. If such data strings are needed, they should be generated automatically from data entered in a form-filling dialogue.

Comment: Transmission of data from one computer to another will often be more economical if field labels and other display formatting features are omitted. In such cases, a format code should be included with the message, so that forms filled by the sender can be re-created in a display useful to the receiver.

Comment: The same arguments apply to transmission of tabular or graphic data, which the user should be able to handle in customary formats, regardless of what the computer-imposed format is for actual transmission.

See also: 5.0-4, 5.0-5.

-4 Message Highlighting

-4

When it will help message handling, transmitted data should be annotated with appropriate highlighting to emphasize alarm/alert conditions, priority indicators, and other significant second-order information.

Comment: Second-order information, i.e., data about data, will often aid processing and interpretation of messages. Such annotation can be automatic (e.g., a computer-generated date/time stamp to indicate currency) and/or added by the sender or receiver (e.g., attention arrows).

See also: Section 2.7.

-1 Source Selection When Sending Messages

-1

When sending data, a user should generally be able to choose whether to transmit directly from computer-stored data files, or from data displays.

Comment: Automatic data transmission will usually be from files. User-initiated transmission might be from displays, where the user can review and annotate messages before sending them.

-2 Source Selection When Receiving Messages

-2

For receiving data, a user should generally be able to specify from what sources data are needed, and/or will be accepted.

Comment: Computer-mediated message handling offers the potential for screening out the electronic equivalent of "junk mail" or "crank calls". A user might be selective in specifying the people or organizations from which messages will be received.

Comment: Source specification may be in terms of data files, or other users, or external sources. Standard sources may be specified as a matter of routine procedure, with special sources designated as needed for particular transactions.

-1 Selecting Destination When Sending Messages

-1

When sending data, a user should generally be able to specify the destinations where data are sent.

Exception: Routing by message content, in bus communication systems.

Comment: Specification of destination may be in terms of data files, or other users, or external destinations, including remote printers. Standard destinations may be specified as a matter of routine procedure, with special destinations designated as needed for particular transactions.

See also: 5.0-6.

-2 Status Information

-2

When sending data, a user should have access to status information concerning what other system users are currently on-line, and the availability of communication with external systems.

Comment: Such status information may influence the user's choice of destinations for data transmission.

See also: 4.1-5, 4.1-7.

-3 Message Printing

-3

The user should be able to request transmission of displayed data to a local printer for making hard-copy records.

Exception: Where security constraints make printed records inadvisable.

Comment: Printing may be regarded as a special case of data transmission, where no other users are necessarily involved.

Comment: User requirements for printed records are often unpredictable to system designers, and so a general printing capability should be provided.

Reference: EG 4.2.14; MS 5.15.4.8.

See also: 2.4-2, 2.4-3, 6.2-6, 6.4-5.

-4 Selecting Destination When Receiving Messages

-4

For receiving data, a user should generally be able to choose the medium of message receipt, i.e., which device will be the local destination.

Comment: Data might be received directly into computer files, or might be routed to an electronic display for quick review, or to a local printer for hard-copy reference purposes.

Comment: Device destination might be specified differently for different types of messages, or for messages received from different sources.

Comment: When transmitted data are received on a display, care should be taken to queue incoming messages, so that they will not interfere with other data processing.

See also: 5.0-6, 5.6-4.

-1 Functional Terminology

-1

The terms used in controlling data transmission, for data specification, message routing and initiation, etc., should be related to the user's job.

Example: A user should be able to address messages to other people or agencies by name, without concern for computer addresses, communication network structure and routing.

Comment: In general, a user should not have to learn the technical details of communication protocols, codes for computer "hand shaking", data format conversion, etc., but should be able to rely on computer tools to handle those aspects of data transmission automatically.

See also: 5.0-3.

-2 Flexible Data Specification

-2

Users should have flexible means for specifying data to be transmitted.

Comment: For sending, the user may wish to specify data by display name or file name, either all or a designated part, or by defined data category.

Comment: For receiving, the user may wish to accept data from specified sources, and/or by defined data categories.

See also: 5.0-6.

-3 Automatic Message Formatting

-3

When data must be transmitted in a particular format, as in data forms or formatted text, computer aids should generate the necessary format automatically.

Comment: It is not sufficient merely to apply computer checking to validate formats generated by the user.

See also: 5.0-2.

-4 Automatic Message Routing

-4

When data are transmitted to standard recipients, computer aids should generate the distribution lists and necessary address headers.

Comment: Users might sometimes wish to modify standard distributions, or the distribution for any particular message. Appropriate review/change procedures should be provided.

See also: 5.0-3.

-5 Automatic Message Initiation

-5

When standard messages must be transmitted following data change, computer aids should be provided to initiate such transmission.

Example: Many operations monitoring tasks.

Comment: Automatic transmission of routine messages will reduce the workload on the user, and help ensure timely reporting. Users might sometimes wish to modify the logic for automatic message initiation, and appropriate review/change procedures should be provided.

-6 User Review of Transmitted Data

-6

When computer aids are provided for selection, routing and initiation of data transmission, users should have the option of reviewing and changing automated message handling logic, in general and for selected messages prior to transmission.

Comment: In applications where message review is critical (perhaps for purposes of security), users might be required to confirm data release for transmission.

See also: 5.4-4, 5.4-5, 6.4-2.

-1 Feedback for Data Transmission -1

Feedback for data transmission should be provided as necessary to assure effective user participation in message handling.

Comment: Specific requirements will vary with the application, but some feedback should be provided.

-2 Feedback for Message Sent -2

Feedback for messages sent should be available as required to advise users of initiation of message transmission, confirmation of message receipt at destination, and/or communication failure.

Comment: In some applications, users may require notification only of exceptional circumstances, as in the event of transmission failure after repeated attempts.

See also: 5.6-2.

-3 Notification of Messages Received -3

Notification of messages received should be available as required to advise users of the type, source and priority of incoming data transmissions.

Comment: In some applications, user may require notification only of urgent messages, and rely on periodic review to deal with routine messages.

See also: 5.6-5.

-4 User Specification of Feedback -4

Users should be able to specify what routine feedback for data transmission should be provided automatically, and also to request specific feedback for particular messages.

Comment: Users may wish to specify minimal feedback, or perhaps none at all, as the automatic notification of routine data transmissions. On the other hand, users may wish to request more specific feedback for transmission of critical messages, as an electronic version of registered mail.

-1 Queuing for Data Transmission

-1

Automatic message queuing should be employed to reduce the need for user involvement in the routine mechanics of data transmission.

Comment: Specific requirements will vary with the application, but some queuing should be provided.

-2 Queuing Messages Sent

-2

Outgoing messages should be queued at user request, to be released by later action, and queued automatically in the event of transmission failure.

Comment: A user may wish to defer data transmission until a batch of related messages has been prepared, or perhaps until a specified date/time for release.

Comment: In the event of transmission failure, automatic queuing and retransmission of outgoing messages will reduce load on the user. If transmission fails in repeated attempts, however, then user intervention may be required, and some notification of that problem should be given to the user.

See also: 5.5-2.

-3 Queuing Messages Received

-3

Unless otherwise specified by a user, incoming messages should be routed automatically to a queue pending subsequent review and disposition by the user.

Comment: Some computer buffering of received data transmissions will be required in any case to deal with near simultaneous receipt of multiple messages. This guideline recommends that the buffer queue for incoming transmissions be enlarged as necessary to permit indefinite retention of messages. Any queue will have limits, of course, and the user should be warned before those limits are exceeded.

See also: 5.0-6, 5.0-7.

-4 Non-Disruptive Message Receipt

-4

Message receipt should be accomplished without interfering with a user's ongoing task.

Comment: As a negative example, an incoming message should not preempt a user's display, but might be signaled by an advisory notice in a portion of the display reserved for that purpose.

Comment: Review and disposition of received messages, like other transactions, should generally require explicit actions by the user. When an incoming message implies an urgent need for user attention, the user should be advised of that urgency.

Reference: EG 7.1.

See also: 5.0-7, 5.3-4, 5.6-5, 6.4-4.

-5 Priority Notification

-5

User notification of queued messages should include clear indication of message priority, and/or other information indicating urgency of user action.

Comment: This practice is required, if incoming messages are queued so as not to disrupt current user tasks. It is important that a user be advised when an interruption may in fact be necessary.

See also: 5.5-3, 5.6-4.

-6 User Review of Messages Received

-6

Convenient means should be provided for user review of received messages in queue, without necessarily requiring any disposition action (i.e., without removal from the queue).

Exception: In some applications, user review of critical messages may be accompanied by a requirement for further disposition actions.

Comment: Rapid review of queued messages will permit a user to exercise judgment as to which require immediate attention, and/or which can be dealt with easily. Other messages may be left in the queue for more leisurely disposition later.

-1 Record Keeping

-1

When a log of data transmissions is required, maintenance of that log should be automatic based on user specification of message types and record formats.

Comment: The objective here is to minimize routine "housekeeping" chores for the user. Once a user has specified the appropriate logging format, i.e., what elements of each message should be recorded, computer aids should generate the log automatically, either for all messages, or for specified categories of messages, or for particular messages identified by the user.

Comment: The same kind of aids should be available for maintaining a journal of data transmissions, in applications where a full copy of each message is required.

See also: 5.0-2, 5.0-3.

-1 Flexible Design for Data Transmission

-1

When data transmission requirements may change, which is often the case, some means should be provided for the user (or an authorized supervisor) to make necessary changes to data transmission functions.

Comment: Data transmission functions that may need to be changed include those represented in these guidelines, namely, changes in the types, sources and destinations of transmitted data, transmission control, feedback, queuing and record keeping.

SECTION 6

DATA PROTECTION

With increasing use of computer-based information systems, there has been increasing concern for the protection of computer-processed data. Data protection is closely allied with other functional areas. The design of data entry, data display, sequence control and data transmission functions can potentially affect the security of the data being processed. In many applications, however, questions of data protection require explicit consideration in their own right.

Data protection must deal with two general problems. First, there is the need to protect data from unauthorized access and tampering. This is the problem of data security. Second, there is the need to protect data from errors by authorized system users, in effect to protect users from their mistakes. This is the problem of error prevention.

Design techniques to achieve data security and to prevent user errors are necessarily different, but they must solve the same dilemma. How can the user interface be designed to make correct, legitimate transactions easy to accomplish, while making mistaken or unauthorized transactions difficult? In each system application, a balance must be struck between these fundamentally conflicting design objectives.

Concern for data security will assume different forms in different system applications. Individual users may be concerned with personal privacy, and wish to limit access to private data files. Corporate organizations may seek to protect data related to proprietary interests. Military agencies may be responsible for safeguarding data critical to national security.

The mechanisms for achieving security will vary accordingly. Special passwords might be required to access private files. Special log-on procedures might be required to assure positive identification of authorized users, with records kept of file access and data changes. Special confirmation codes might be required to validate critical commands.

At the extreme, measures instituted to protect data security may be so stringent that they handicap normal system operations. Imagine a system in which security measures are designed so that every command must be accompanied by a continuously changing validation code which a user has to remember. Imagine further that

when the user makes a code error, which can easily happen under stress, the command sequence is interrupted to re-initiate a user identification procedure. In such a system, there seems little doubt that security measures could reduce operational effectiveness.

In recent years, computer security measures have concentrated increasingly on automatic means for data protection, implemented by physical protection of computing equipment and by tamper-proof software. Automation of security makes good sense. If data security can be assured by such means, there will be less need to rely on fallible human procedures. And, of course, user interface design will be that much easier.

It seems probable, however, that absolute data security can never be attained in any operational information system. There will always be some reliance on human judgment, as for example in the review and release of data transmissions, which will leave systems in some degree vulnerable to human error. Thus a continuing concern in user interface design must be to reduce the likelihood of errors, and to mitigate the consequences of those errors that do occur.

Like data security, error protection is a relative matter. An interface designer cannot reasonably expect to prevent all errors, but frequent user errors may indicate a need for design improvement. Data protection functions must be designed 1) to minimize the entry of wrong data into a system; 2) to minimize mistakes that make wrong changes to stored data; and 3) to minimize the loss of stored data. In considering these objectives, prevention of catastrophic data loss is clearly vital for effective system operation, but all three aspects of data protection are important.

Data entry and change, of course, are transactions frequently performed by system users. Careful interface design can help prevent many errors in those transactions, by providing automatic data validation and reversible sequence control, as described in previous sections of these design guidelines. But the designer needs a good deal of ingenuity in applying guidelines within the context of each data handling job.

The use of job context for computer validation of user inputs is best illustrated by example. Here is a discussion of error prevention in data entry that was published in a local newspaper, suggesting ways to reduce billing errors:

. . . designers of applications systems have resorted to a number of strategies to minimize ill effects and keep the errors from escaping into the world at large. The commonest of these is the process known as 'verification,' which in

its simplest form, means instructing the system to respond to input with the figurative question, 'Do you really mean that?'

That is, when a data[-entry] operator enters an amount, or name, or serial number -- the system draws attention to the just-typed item (by causing it to flash on-screen, for example). The operator then is supposed to take a good hard look at the item and press a verification key if the data is correct.

Better yet, what you need is for the system to do some checking on its own . . . to a certain extent, it can use internal evidence (and the percentages) to perform its own verification.

Here's a simple strategy that, though currently used to some extent, will some day become universal, one hopes. It's good because it relies on an understanding of human habits.

Let's take billing again. Most times, when you pay a bill, you pay either the minimum amount due or the full balance. Suppose we instruct the machine to compare the operators entry of 'amount paid' with the minimum due and with the full balance for that particular account. If the entry is equal to one or the other, pass it on through. It's very likely correct. If there's a discrepancy, discontinue entry and signal the operator to check the amount.

And it does so optimally: the right ones pass through with minimum slow down, the potential wrong ones get the attention.

(Bertoni, 1982)

Once data are correctly entered into a computer, the emphasis shifts to prevention of unwanted changes to the data, including the extreme form of change represented by data loss. Stored data must be protected from vicissitudes of computer operation and also from system users. Advances in computer technology, with less volatile memory, automatic archiving to back up data stores, and redundant processing facilities, have significantly reduced the hazard of data loss resulting from machine failure. What remains is to reduce the hazard of human failure.

In the interface design guidelines presented here, the primary concern is for the general users of computer systems. It must be

noted, however, that data protection requires consideration of other aspects of system design and operation. In particular, the expert operators who maintain and run the computer system must assume a large responsibility for data protection.

Consider an example. In one computer center, an operator must enter a command '\$u' to update an archive tape by writing a new file at the end of the current record, while the command '\$o' will overwrite the new file at the beginning of the tape so that all previous records are lost. A difference of one keystroke could obliterate the records of years of previous work. Has that ever happened? Yes, it has. If an error can happen, then it probably will happen.

In that respect, expert computer operators are just like the rest of us. When tired, hurried or distracted, they can make mistakes. And not all computer operators are experts. Some are still learning their jobs, and so may be even more error prone. Careful design and supervision of operating procedures is needed to minimize data processing errors.

If data loss from machine failure and data loss from faulty system operation are minimized through careful design, then the most serious threat to data protection is the system user. This is especially true in applications where the user must participate directly in establishing and maintaining stored data files. Means must be found to protect files from inadvertent erasure.

It is clear that some difficult design trade-offs may be required. As an example, consider the guideline recommendation that a user not be allowed to change or delete data without first displaying the data. In a file deletion transaction it would usually be impractical to force a user to review the entire file. One might imagine displaying the first page of a file nominated for deletion, and requiring the user to CONFIRM the DELETE action. But even that would be disruptive in many circumstances.

As a fall back position, we might recommend that when a file has been nominated for deletion enough descriptive information should be displayed about that file so that a reasonably attentive user can determine whether that file should be deleted. The issue is how to ensure that the user knows what he/she is doing.

When a user selects a file for deletion, at least as much information should be provided as when a user selects a file for display and editing. Thus, if an on-line index of displayable files contains a line of information about each one, perhaps including name, description, size and currency (date last changed), then such information would also be appropriate in prompting the CONFIRM action for file deletion:

CONFIRM DELETION OF THE FOLLOWING FILE:

USIplan 5-year plan for USI effort 3 pages 11-25

Any required confirmation procedure, of course, will tend to slow file deletion, in accord with the general guideline that destructive actions should be made difficult. Where is the trade-off when destructive actions are also frequent? What about the user who wishes to scan a file index and delete a series of files? Must each separate DELETE action be confirmed? Unless DELETE actions are easily reversible, the answer for most users is that an explicit confirmation probably should be required for each file deletion. When a user must undertake a series of file deletions, the repetitive nature of the task may increase the risk of inadvertent deletion, and so increase the need for CONFIRM protection.

It should be recognized that explicit DELETE commands are not the only actions that can result in accidental file erasure. In some systems, it is possible to over-write a stored file with whatever data are currently in temporary, on-line, "working" storage. Used properly, this capability permits desired editing and replacement of files. A user might call out a file, make changes to it, and then re-store it under its own name.

Used improperly, this capability risks file deletion. A user might call out and edit one file, but then absent-mindedly store it with the name of another file, thus over-writing whatever data had been previously stored in that other file. Such a hazard requires just as much protection as an outright DELETE action, or perhaps even more since the danger is more subtle. In effect, an explicit CONFIRM action should be required whenever a user attempts to store a data file under the name of any other file already stored in the system. The prompt for confirmation might read something like this:

CONFIRM OVER-WRITING THE CURRENT FILE OF THIS NAME:

SCG sequence control guidelines 45 pages 10-08

It is interesting that many systems do not require this kind of selective confirmation. One well known system requires user confirmation of every over-write action, even in the common case where an edited file is being stored by the same name to replace its previous version. Thus the CONFIRM action itself becomes routine, and no longer provides any significant protection to a forgetful user. Another system avoids the problem by the rigid expedient of allowing a user to store an edited file only under its last previous name, which is safe but sometimes inconvenient.

To some extent a wary user can protect himself/herself by careful selection of file names, trying to ensure that any file name is descriptive of file content, and also distinctive in comparison with the names of other files. In practice, that goal is hard to achieve. Users often work with groups of files dealing with different aspects of a common topic. For such files, if names are descriptive they will tend to be similar rather than distinctive. If file names are made longer in order to become more distinctive, then that may reduce efficiency in the general use of file names for storage and retrieval.

In systems where there is no effective on-line protection against inadvertent file deletion (or replacement), either a user must be exceedingly careful, or else the system must provide effective off-line procedures to recover from archive records an earlier version of an accidentally erased file. Neither alternative is entirely satisfactory. Even a careful user will make mistakes. And archives will not protect a user from loss of current work.

A better solution can be provided by on-line computer aids to make user actions reversible. In effect, systems should be programmed to permit users who notice unintended deletions to retrieve lost files by taking an UNDO action. Some current systems provide such an UNDO capability, permitting users to change their minds and to correct their more serious mistakes.

There must be, of course, some practical time limit to the reversibility of data processing. A user might be able to UNDO the last previous deletion, or perhaps even all deletions made during the current working session, but there seems no feasible way to make it easy to undo a particular deletion made days ago and now regretted. Moreover, reversibility will not help a user who does not notice that a mistake has been made. So even where an UNDO capability is available, other aspects of the user interface must still be carefully designed.

The guidelines proposed in the following pages illustrate the range of topics to be considered in this area, and the general need for many kinds of data protection. These guidelines draw heavily from material already presented in previous sections, as indicated by the extensive cross referencing. In this new context of data protection, some previous guidelines have been slightly reworded, others preserved intact. They are repeated here for the convenience of a designer who must review all material pertinent to data protection functions.

DATA PROTECTION

Objectives: Effective data security
Minimal entry of wrong data
Minimal loss of needed data
Minimal interference with data handling tasks

DATA PROTECTION **General** **6.0**

-1 Automatic Security Measures -1

Data security should be protected by automatic measures whenever possible, relying on computer capabilities rather than on fallible human procedures.

-2 Consistent Procedures -2

User interface design should provide consistent procedures for data transactions, including data entry and error correction, data change and deletion.

Comment: Consistent procedures will reduce the likelihood of user confusion and error, and are especially important for any transaction that risks data loss.

Reference: BB 2.2.1, 6.1.4.

See also: 4.0-1, 5.0-1, 6.5-6.

-3 Explicit User Actions

-3

Inputs to the computer, including data entries and control entries, should require explicit user actions.

Exception: An exception can be made for repetitive tasks, as when correct entry of one data set in a form-filling dialogue might automatically result in display of the next (empty) form, without specific user request.

Exception: Automatic cross-file updating.

Comment: In effect, a computer should not initiate data changes unless requested (and possibly confirmed) by a user. Interface designers are sometimes tempted to contrive "smart shortcuts" in which one user action may automatically produce several other associated data changes, perhaps saving the user a few keystrokes. Since such shortcuts cannot generally be made standard procedures, they will tend to confuse novice users, and so may pose a potential threat to data protection.

See also: 1.0-8, 1.1-4, 3.0-5, 3.1.3-4, 3.5-7, 4.0-2, 5.0-7.

-4 Feedback for Mode Selection

-4

When the result of user actions will be contingent upon prior selection among differently defined operational modes, that mode selection should be continuously indicated to the user, particularly when user inputs in that mode might result in unintended data loss.

Example: DELETE mode when editing displayed data.

Comment: A user cannot be relied upon to remember prior actions. Any action whose results are contingent upon previous actions represents a potential threat to data protection.

See also: 4.2-8, 6.5-13.

-5 Error Management

-5

User interface design should deal appropriately with all possible control entries, correct and incorrect, without introducing unwanted data change.

Comment: The interface designer must try to anticipate every possible user action, including random keying and perhaps even malicious experimentation. The user interface must be "bullet-proofed" so that an unacceptable entry at any point will produce no more significant computer response than an error message.

Reference: PR 4.12.4.5.

See also: 3.5-1.

-6 Editing Entries

-6

For both data entry and control entry, the user should be able to edit composed material before initial entry and also before any required re-entry.

Comment: This capability will permit a user to correct many entry errors before computer processing. When errors are made, the user will be able to fix them without having to regenerate correct items and risk introducing further errors.

Reference: EG 5.4.

See also: 1.4-2, 3.5-2, 4.3-14.

-7 Resolving Ambiguous Entries

-7

For both data entry and control entry, the user should be required to resolve any detected ambiguity requiring computer interpretation.

Example: Resolving ambiguous abbreviation by selecting among displayed alternatives.

See also: 1.0-19, 3.1.5-16, 3.5-8, 6.5-11.

-8 Warning the User

-8

The user should be warned of potential threats to data security by appropriate messages and/or alarm signals.

Reference: BB 7.7.2, 7.7.3; EG 2.1.3; MS 5.15.2.10.b,
5.15.4.6.1.a.

See also: 4.3-16, 4.3-17, 6.5-17.

-1 User Identification

-1

Users should be identified at LOG-ON in a process as simple as possible consistent with protecting data security.

Comment: Authentication of user identity is generally not enhanced by requiring a user to enter routine data such as terminal, telephone, office or project numbers. In most organizations, those data can readily be obtained by other people. If verification of those data is needed, the user should be asked to review and confirm currently stored values in a supplementary procedure following LOG-ON.

See also: 1.8-1.

-2 Authenticating User Identity

-2

The LOG-ON process should include prompted entry of passwords and/or whatever other data are required to confirm user identity and to authorize appropriate data access/change privileges.

Reference: EG 4.2.11.

-3 User Choice of Passwords

-3

When passwords are required, a user should be able to choose and/or to modify his/her own password.

Comment: A password chosen by the user will generally be easier for that individual to remember. User choice is especially helpful when passwords must be periodically changed.

-4 Covert Entry of Passwords

-4

When a private password must be entered by a user, that entry should be covert, i.e., should not be displayed.

Comment: This represents an exception to the general recommendation that all entries should be displayed.

See also: 1.0-2.

-5 Preserving User Identity

-5

In general, once a user's identity has been authenticated, whatever data access/change privileges are authorized for that user should continue throughout a work session.

Exception: In special instances a user's data access/change privileges might reasonably change as a result of succeeding transactions.

Exception: A user might reasonably be required to repeat procedures for authentication of identity following some specified period of inactivity at the work station.

Comment: If a previously identified user later is required to take separate actions to authenticate data handling transactions, such as access to particular files or issuance of particular commands, the efficiency of system operations may be degraded. Where continuous verification of user identity seems required for data protection, perhaps some automatic means of identification can be devised for that purpose.

See also: 6.2-1, 6.3-1.

-1 Data Access

-1

In general, user authorization for data access should be established at initial LOG-ON, and should not require special procedures for further authentication in connection with particular data display requests.

See also: 6.1-5.

-2 Displayed Security Classification

-2

When displayed data are classified for security purposes, an indication of security classification should be included prominently in each display.

Comment: This practice will serve to remind users of the need to protect classified data, both in access to the display itself and in any further dissemination of displayed data.

Comment: In applications where either real or simulated data can be displayed, a clear indication of simulated data should be included as part of the classification label.

Comment: Where a display includes several partitioned "windows" of data from different sources, it may be necessary to label security classification separately for each window. Under those conditions, some form of auxiliary coding (e.g., color coding) may help the user distinguish among different classes of data.

See also: 6.5-4.

-3 Changes to Displayed Data

-3

In general, users authorized to access data for display should also be authorized to make changes to displayed data; where data changes are not authorized, that should be indicated on the display.

Comment: If users can generally make additions and/or corrections to displayed data, then any exception to that practice in the interests of data protection will risk confusing the user.

See also: 2.0-5, 3.5-15.

-4 Protection of Displayed Data -4

When protection of displayed data is essential, maintain computer control over the display and do not permit the user to change controlled items.

Comment: Never assume compliance with instructions by the user, who may attempt unwanted changes by mistake, or for curiosity, or to subvert the system.

Reference: EG 3.4.8.

See also: 2.0-6.

-5 Display Format Protection -5

Some portions of data displays, such as field labels and other formatting features, should routinely be protected from accidental change by users.

See also: 1.1-22, 1.4-7.

-6 Printouts -6

Insofar as possible within constraints of data security, a user should be allowed to generate printed copies of displayed data.

Comment: User requirements for printed data are often unpredictable, and printing restrictions may handicap task performance. Rather than restrict printing, appropriate procedures should be established for restricting further distribution of data printouts.

Reference: BB 1.7; EG 4.2.14; MS 5.15.4.8.

See also: 2.4-2, 5.3-3, 6.4-5.

-7 Data Access Records

-7

When records of data access are required in the interests of monitoring data protection measures, those records should be kept automatically by the computer without any explicit action by the user.

Comment: Even cooperative, well-intentioned users can forget to keep manual logs of data access, and will resent the time and effort required to keep such logs. Subversive users, of course, cannot be expected to provide accurate records.

See also: 4.5-4.

-1 Authorization for Data Entry/Change

-1

In general, user authorization for data entry/change should be established at initial LOG-ON, and should not require special procedures for further authorization in connection with particular data entry/change transactions.

See also: 6.1-5.

-2 Data Entry/Change Transaction Records

-2

In system applications where it is warranted for purposes of data protection, the user (and/or an authorized supervisor) should be able to request a record of data entry/change transactions.

Comment: Transaction records might, of course, be maintained for purposes of user guidance as well as for data protection, as recommended elsewhere.

See also: 3.4-2, 4.4-18, 4.5-3.

-3 Protection from Data Change

-3

When protection from data entry/change is essential, maintain computer control over the data and do not permit a user to change controlled items.

Comment: Never assume compliance with instructions by the user, who may attempt unwanted changes by mistake, or for curiosity, or to subvert the system.

Comment: Similar considerations also dictate routine protection of display formatting features, as recommended elsewhere.

See also: 1.1-22, 1.4-7, 2.0-6, 6.2-5.

-4 Protect Real from Simulated Data

-4

When simulated data are processed and stored, as for on-line user training, care should be taken that changes to simulated data are processed separately and do not affect real data.

See also: 4.4-25, 6.2-2.

-5 Accuracy of Data Entry/Change

-5

When accuracy of data entry/change is more important than speed, display formats and user guidance should emphasize the accuracy requirement.

Comment: Slow but correct initial entry of data will take less time than a hasty entry that must later be corrected.

Reference: EG 7.2.

-6 Simple Data Entry/Change Procedures

-6

To help ensure data accuracy, data entry/change procedures should be simplified, insofar as possible, following general guidelines for data entry, so that the user can enter short rather than long items, need not enter leading zeros, need not count blanks, etc.

See also: 1.0-11, 1.0-24, 1.0-25, 3.1.5-14.

-7 Default Values

-7

Currently operative default values for data entry should be displayed for user confirmation prior to entry and processing by the computer.

See also: 1.7-7, 1.7-8.

-8 Explicit Action for Data Entry/Change

-8

Data entry/change should result only from an explicit ENTER action by the user, and not as a possibly unrecognized side effect of other actions.

Example: A dual-activation lightpen that permits pointing at an item without data entry/change unless some further explicit action is taken.

Exception: Automatic generation of routine, redundant or derived data.

Comment: Explicit actions will help direct user attention to data entry/change, and reduce the likelihood of thoughtless errors.

See also: 1.0-8, 1.1-4, 3.0-5, 3.1.3-4, 3.5-6, 4.0-2, 6.0-3.

-9 Single Entry of Related Data

-9

Entry of logically related items, as in a form-filling dialogue, should be accomplished by a single, explicit action at the end of the sequence, rather than by separate entry of each item.

Comment: This practice permits user review and possible data correction prior to entry. It will also permit efficient cross validation of related data items by the computer.

See also: 1.4-1, 6.3-18.

-10 Data Correction Before Entry

-10

A user should be able to correct keyed data items or commands prior to taking an explicit ENTER action.

Comment: Easy correction prior to entry will avoid the need for computer processing of user-detected errors.

Comment: Error correction procedures should permit back-spacing with a nondestructive cursor to the point of error, re-keying the corrected item, followed by an ENTER action without any further cursor positioning.

Reference: EG 5.4; MS 5.15.1.2.4.

See also: 1.4-2, 3.5-2, 6.0-6.

-11 Immediate Error Correction

-11

When an error in data entry/change has been detected, the user should be permitted to make an immediate correction.

Comment: Immediate corrections will be made more easily and accurately, when the data (e.g., source documents) are still available to the user.

Reference: EG 5.7; MS 5.15.1.2.6.

See also: 1.7-4, 3.5-13.

-12 Selective Error Correction

-12

Following error detection, users should be required to re-enter only that portion of a data entry that was not correct.

Comment: If the user must re-enter an entire data set to correct one wrong item, then there will be the risk of new errors in previously correct items.

Reference: EG 4.2.3, 5.4; MS 5.15.1.2.7.b.

See also: 4.3-14, 6.0-6.

-13 Flexible BACKUP for Error Correction

-13

A user should be able to return easily to previous steps in a transaction sequence in order to correct an error or make any other desired change.

Reference: MS 5.15.1.2.6.

See also: 3.5-14.

-14 Explicit Action for Error Correction

-14

The user should be required to take an explicit ENTER action to request processing of corrections to wrong data.

Reference: PR 4.12.6.

See also: 3.5-7, 6.0-3.

-15 Data Verification by User Review

-15

When verification of prior data entries is required, that should be accomplished by user review and confirmation, rather than by requiring re-entry of the data.

Comment: For routine verification, data review by the user will be quicker than re-entry, with less risk of introducing new errors.

Comment: For special verification, as when computer processing has detected doubtful and/or discrepant data entries, the user should be alerted with an appropriate advisory message.

See also: 1.8-3, and Section 4.3.

-16 Automatic Generation of Data Entry/Change

-16

When routine/redundant data are available in the computer, those data should be accessed or derived automatically for user review, rather than requiring entry by the user.

Comment: This represents an exception, in the interests of improved data accuracy, to the general recommendation that data entry/change should occur only as a result of explicit user actions. Automatic data generation by the computer, where it can be based on derived values or cross-file updating, will be faster and more accurate, with risk of error being introduced in re-entry by the user. In effect, having a computer do automatically what the user may do poorly is here regarded as a form of data protection.

See also: 1.8-1, 1.8-2, 1.8-4, 1.8-5.

-17 Automatic Validation of Data Changes

-17

Software for automatic validation of data entry/change should be incorporated to check insofar as possible for erroneous/doubtful values.

Comment: Do not rely on the user always to make correct entries. When validity of data entries can be checked automatically, such computer aids will help improve accuracy of data entry.

Reference: MS 5.15.1.2.2; PR 4.12.4.

See also: 1.7-1.

-18 Automatic Cross Validation

-18

Whenever possible, automatic cross validation of data entries and changes should be provided to ensure that each data item is logically consistent with other related data items.

Comment: Such cross checking is one of the potential advantages of on-line data handling, helping the user detect logical errors.

Reference: MS 5.15.1.2.2; PR 4.12.5.

See also: 1.4-1, 1.7-1, 6.3-9.

-19 Displayed Feedback for Changing Data

-19

When a user requests change (or deletion) of a stored data item that is not currently being displayed, then both the old and new values should be displayed so that the user can confirm or nullify the change before the transaction is completed.

Comment: This practice will tend to prevent inadvertent change, and is particularly useful in protecting delete actions.

See also: 1.0-10, 6.5-10.

-20 User Confirmation

-20

A user should be asked to take explicit action to confirm doubtful and/or potentially destructive data change commands before they are accepted by the computer for execution.

Comment: A requirement to take an explicit CONFIRM action will direct user attention to questionable data changes, and help the user avoid the consequences of thoughtless errors.

See also: 4.3-15, 4.3-16, 6.0-8, 6.5-18.

-1 Automatic Protection of Transmitted Data

-1

In general, whatever measures are adopted to protect data during transmission, e.g., encryption, parity checks, buffering until acknowledgment of receipt, etc., should be applied by the computer automatically, without the need for any explicit action by a user.

Comment: Users are fallible, and cannot be relied upon to participate quickly and accurately in the mechanisms of data transmission, whereas this is the sort of thing that computers can do well. A user might be asked to supply an encryption key, but the computer should handle any actual encryption process.

See also: 6.0-1.

-2 User Review of Transmitted Data

-2

When human judgment may be required to determine whether to release data for transmission, convenient means should be provided the user (and/or an authorized supervisor) to review outgoing messages and confirm their release before transmission.

Comment: Sometimes message release may require coordination among several reviewers in the interests of data protection.

See also: 5.4-6.

-3 Data Protection While Sending

-3

When data are being sent, a copy should be retained by the originator of the transmission until correct receipt is confirmed, and possibly longer.

Comment: The primary objective here is to prevent irretrievable data loss during transmission. For most system applications, however, the originator of a message will probably want to retain a copy in any case, and will prefer that any subsequent deletion be handled as a separate transaction, distinct from data transmission.

-4 Data Protection While Receiving

-4

When data are being received, incoming messages should be queued as necessary to ensure that they will not disrupt or destroy any ongoing data transactions by a user.

Comment: In general, incoming data should not replace existing data until reviewed by the user. Exceptions must be made, however, in applications where automatic updating of current situation data is required for effective user monitoring, as in air traffic control systems. The difference there is that data updating is the primary purpose of the system, and the user knows what is going on.

See also: 5.6-4.

-5 Message Printout

-5

Insofar as possible within constraints of data security, a user should be allowed to generate printed copies of transmitted data, including messages sent and received.

Comment: As noted elsewhere, user requirements for printed data may be unpredictable, and restrictions on data printout may hinder task performance. Rather than restrict printout, appropriate procedures should be established for restricting further distribution of printed messages.

See also: 2.4-2, 5.3-3, 6.2-6.

-1 Data Protection from Computer Failure

-1

Information systems should be designed to minimize data loss from computer failure.

Example: Depending upon the criticality of the application, different protective measures may be justified, including periodic automatic archiving of data files, maintenance of transaction logs for reconstruction of recent data changes, or even provision of parallel computing facilities.

Comment: An automatic capability is needed because users cannot be relied upon to remember to take necessary protective measures.

Comment: Though not strictly a feature of user interface design, reliable data handling by the computer will do much to maintain user confidence in the system. Conversely, data loss resulting from computer failure will destroy user confidence, and reduce user acceptance where system use is optional.

-2 Data Protection from Other Users

-2

User transactions and data should routinely be protected from actions by other users.

Comment: When one user's actions can be interrupted by another user, as in defined emergency situations, that interruption should be temporary and nondestructive. The first user should subsequently be able to resume operation at the point of interruption without data loss.

Reference: MS 5.15.2.5.

See also: 3.0-20.

-3 Data Protection from User Interrupt

-3

User interrupts of data processing should be accomplished without incorrect modification of stored data.

Reference: BB 1.8.

See also: 3.3-6.

-4 Segregating Simulated Data -4

When simulated data and system functions are provided for on-line user training, care should be taken to protect real data and to distinguish simulated from actual system operation.

Reference: BB 5.4.

See also: 4.4-25, 6.2-2.

-5 Ease/Difficulty of User Control -5

The ease of sequence control by the user should match desired ends; frequent or urgent actions should be easy to take, whereas potentially destructive actions should be made sufficiently difficult to require explicit user attention.

See also: 3.0-3.

-6 Standard Procedures -6

User interface design should provide standard procedures for accomplishing different types of transactions, to facilitate user learning and efficient system operation.

Comment: Standard procedures will help the user build consistent operational habits, reduce confusion, and reduce the likelihood of user errors.

See also: 4.0-1, 6.0-2.

-7 Disabling Unneeded Function Keys -7

Function keys (and other devices) not needed for current control entry should be temporarily disabled by the computer, especially those with potentially destructive effects.

See also: 3.1.4-10, 3.2-11.

-8 Protected Keys

-8

Function keys (and other devices) whose activation may result in data loss should be located and/or physically protected in such a way as to reduce the likelihood of accidental activation.

See also: 3.1.4-13.

-9 Data Entry Independent of Cursor Placement

-9

An ENTER action for multiple data items should result in entry of all items, regardless of where the cursor is placed on the display.

Comment: A user may choose to move the cursor back in order to correct earlier data items, and cannot be relied upon to move the cursor forward again. The computer should ignore cursor placement in such cases.

See also: 1.1-23.

-10 Displayed Feedback for Changing Data

-10

When a user requests change (or deletion) of a stored data item that is not currently being displayed, then both the old and new values should be displayed so that the user can confirm or nullify the change before the transaction is completed.

Comment: This practice will tend to prevent inadvertent change, including changes resulting in loss of needed data. User attempts at selective data change without displayed feedback will be error prone.

Comment: For delete actions involving significant amounts of data, such as entire files, display of all the data will probably not be feasible. In such instances, the user should be clearly warned of potential data loss and required to confirm that destructive action.

See also: 1.0-10, 6.3-19, 6.5-17, 6.5-18.

-11 Interpreted Commands

-11

When an ambiguous command entry must be interpreted by the computer, and especially when the interpreted command may threaten data loss, the user should be given an opportunity to review and confirm a displayed interpretation of the command before it is executed.

See also: 3.1.5-16, 6.0-7.

-12 Destructive Defaults

-12

Commands that might result in data loss should not be provided as automatic defaults for command entry.

Example: When requesting a printout of filed data, one option may be to delete that file after printing; the default value for that option should automatically be set to NO whenever printing options are presented to a user for selection.

-13 Destructive Operational Modes

-13

Operational modes that might result in data loss should not be established automatically by the computer, but only if explicitly selected by a user.

Example: DELETE mode in text editing.

Comment: In most applications, it may be better not to provide any destructive modes. Rather than providing a DELETE mode, require that DELETE be a discrete action subject to confirmation by the user. User interface design must determine the proper balance here between data protection and operational efficiency.

See also: 4.2-8, 6.0-4.

-14 Data Protection from User Error**-14**

No single user error should cause significant change or loss of stored data; users should be required to take extra actions to implement any destructive command, such as deleting a data file.

Example: Enter next command: D

If deleted, all data in this file will be lost.
Enter YES to confirm deletion: _____

Reference: BB 1.10; EG 4.2.8; MS 5.15.1.2.3, 5.15.1.2.7.c.

See also: 3.5-9, 4.3-16, 6.0-8, 6.3-20, 6.5-18.

-15 Distinctive File Names**-15**

When data files may be deleted by name, care should be taken to ensure that the names of different files are distinctive.

Comment: If this guideline is not followed, it is easy for the user to make an error in file storage, by specifying an unintended overwriting of one file with data from a similarly named other file.

Comment: When two or more files are similarly named, the distinctive feature should be near the beginning of the names rather than at the end; in particular, no file name should simply be a truncated version of another.

Comment: In many applications, file naming is a user option, and distinctive naming can be achieved only as a matter of good operational procedure. In such circumstances, perhaps the only feasible aid in the programmed user interface might be a computer-generated advisory message when a proposed new file name is similar (e.g., identical in the first 5 letters) to the name of an existing file.

-16 Preventing Data Loss at LOG-OFF

-16

When a user requests LOG-OFF, the computer should check pending transactions and, if data loss seems probable, should display an appropriate advisory message.

Example: CURRENT DATA ENTRIES HAVE NOT BEEN FILED;
SAVE IF NEEDED BEFORE CONFIRMING LOG-OFF.

Comment: The user may sometimes suppose that a job is done before taking necessary implementing action.

See also: 3.5-12, 4.3-14.

-17 User Warned of Potential Data Loss

-17

For conditions requiring (or implying the need for) special user attention to protect against data loss, an explicit alarm and/or warning message should be provided.

Reference: BB 7.7.2, 7.7.3.

See also: 3.5-10, 4.3-14, 4.3-17, 6.0-8, 6.5-14.

-18 User Confirmation of Destructive Actions

-18

A user should be required to take explicit action to confirm potentially destructive entries before they are accepted by the computer for execution.

See also: 3.5-9, 4.3-16, 6.3-20, 6.5-14.

-19 Distinctive CONFIRM Action

-19

User confirmation of an entry should be accomplished with a distinctively labeled CONFIRM function key.

See also: 3.5-11.

-20 Reversible Control Actions: UNDO

-20

Insofar as practical, the computer should maintain a record of data changes resulting from current transactions, and provide a capability for a user to UNDO any recognized error that caused an unintended data loss.

Comment: Some version of such an UNDO capability is now often provided in interface design. UNDO represents one more level of data protection, when warning messages and confirmation procedures fail to prevent error, but can only help the user who notices that an error has been made.

-1 Flexible Design for Data Protection

-1

When data protection requirements may change, some means should be provided for the user (or an authorized supervisor) to make necessary changes to data protection functions.

Comment: Data protection functions that may need to be changed include those represented in these guidelines, namely, changes in protective measures regulating user identification, data access, data entry/change, data transmission, and methods of loss prevention.

-2 Protection from Design Change

-2

User interface design should be protected from changes that might impair functions supporting data entry, data display, sequence control, user guidance, data transmission and data protection.

Comment: A trade-off is required between design flexibility, to permit needed improvements to the user interface, and design control, to protect current functions from undesirable changes. Some form of continuing configuration management should be instituted to evaluate changes to user interface design, just as for any other critical system interface.

REFERENCES

Anyone involved in compilation of USI design guidelines must begin and end by acknowledging the significant contributions of other people. This undertaking is truly a collaborative effort, as each new student of the field builds upon the work of others. This is a good thing. All USI design guidelines are based in some degree on judgment, and the collective judgment of multiple contributors may well prove sounder than the views of just one person.

Most of the guidelines presented in this report were not invented here, but were built on contributions by other people. Where the idea for a guideline came from a particular source, or is supported by other published recommendations, appropriate reference annotation has been included for that guideline.

Reference annotation is probably of interest more to the scholar, a student of guidelines, than to the designer, who is a user of guidelines. Such annotation offers credit, where credit is due. More importantly, cited references permit anyone questioning a particular guideline to explore its antecedents, perhaps to gain a better understanding of what is intended.

References to specific articles and papers have been cited in conventional form by author and date. A detailed listing of those references is presented in the pages that follow. Four reference sources contributed generally to the guidelines presented here. Those sources are referenced so frequently here that they have been cited in abbreviated form, simply by initials:

BB = Brown, Burkleo, Mangelsdorf, Olsen and Williams, 1981

EG = Engel and Granda, 1975

MS = MIL-STD-1472C, 1981

PR = Pew and Rollins, 1975

The 1975 IBM report by Engel and Granda (EG) was the first widely recognized compilation of USI design guidelines. That report has provided inspiration and has served as a seminal reference for others working in this field. It is still in demand, and has been reprinted to permit its continued distribution. That report has been cited here 187 times, for some 152 guidelines.

The 1975 BBN report by Pew and Rollins (PR) represents an admirable attempt to establish USI design guidelines for one

particular (proposed) system application. Its recommendations, however, can readily be generalized for broader application. That report has been cited here 84 times, for 74 guidelines.

The 1981 report by Lin Brown and his colleagues at Lockheed (BB) is a good example of USI design guidelines developed for in-house use, but which are available for public reference. That report has been cited here 182 times, for 150 guidelines. That report has also influenced the formatting of guidelines adopted here, including the use of short titles. The Lockheed guidelines have been undergoing further revision, and an expanded update of the 1981 version should be published in the near future.

MIL-STD-1472C (MS), the current US military standard for human engineering in system design, has been cited here 125 times, for 111 guidelines. It must be recognized, of course, that guidelines do not necessarily carry the same weight as design standards, usually being cited as guidance for system acquisition rather than being imposed contractually. And it must be acknowledged that the current standard offers only rudimentary coverage for USI software design. Nonetheless, it may be useful to note what correspondence there is between the current standard and the guidelines proposed here. The standard itself will probably be expanded in the future to provide improved coverage in this area.

It should be emphasized that citation of references does not necessarily mean that their authors would agree with the wording of guidelines presented here. In some instances, an idea has been borrowed intact, with the wording of a previously published guideline preserved. In many more cases, however, ideas have been modified and reworded here, sometimes drastically, perhaps beyond the intent of their original authors.

Revision of guidelines published elsewhere has not been undertaken capriciously, but only after careful consideration. And still further revision is surely needed. During the past several years, readers of previous reports in this series have been asked to review and criticize earlier versions of the guidelines proposed here. Improvements have been made to successive versions of the guidelines as a result of recommendations received from thoughtful critics:

Sara R. Abbott	Union Carbide Corporation
Christopher J. Arbak	Systems Research Laboratories, Inc.
J. David Beattie	Ontario Hydro
C. Marlin Brown	Lockheed Missiles and Space Company
Kent B. Davis	Litton Data Command Systems
Richard M. Kane	Essex Corporation

Lorraine F. Normore Ohio State University Human Performance Center
Steven P. Rogers Anacapa Sciences, Inc.
Eric M. Schaffer Human Performance Associates
John C. Thomas IBM Corporation

Critical review of USI design guidelines must continue. No guideline proposed here is worded so perfectly that it cannot be improved. And a need persists for more illustrative examples, more completely noted exceptions, more explanatory comment, and more references.

Perhaps you can help in this work by proposing new guidelines, or by suggesting improvements to those published here, or by citing examples, exceptions, and references. If so, please copy the change sheet included at the back of this report, and use it to record your recommendations.

You should record your comments not only where you believe a guideline is wrong, but also if a guideline does not seem clearly stated. If the wording of a guideline is not clear to you, then it will probably confuse other people who try to use it in the future.

- Albert, A. E. The effect of graphic input devices on performance in a cursor positioning task. In Proceedings of the 26th Annual Meeting. Santa Monica, California: Human Factors Society, 1982, 54-58.
- Aretz, A. J. and Kopala, C. J. Automatic return in multifunction control logic. In Proceedings of the 25th Annual Meeting. Santa Monica, California: Human Factors Society, 1981, 254-256.
- Bertoni, P. Of slipped disks Boston Globe, 22 June 1982.
- Billingsley, P. A. Navigation through hierarchical menu structures: Does it help to have a map? In Proceedings of the 26th Annual Meeting. Santa Monica, California: Human Factors Society, 1982, 103-107.
- Brown, C. M., Burkleo, H. V., Mangelsdorf, J. E., Olsen, R. A., and Williams, A. R., Jr. Human Factors Engineering Standards for Information Processing Systems. Sunnyvale, California: Lockheed Missiles and Space Company, Inc., 15 June 1981.
- Bury, K. F., Boyle, J. M., Evey, R. J., and Neal, A. S. Windowing versus scrolling on a visual display terminal. Human Factors, 1982, 24(4), 385-394.
- Butterbaugh, L. C. Evaluation of alternative alphanumeric keying logics. Human Factors, 1982, 24(5), 521-533.
- Campbell, A. J., Marchetti, F. M., and Mewhort, D. J. K. Reading speed and text production: A note on right-justification techniques. Ergonomics, 1981, 24(8), 633-640.
- Cohill, A. M. and Williges, R. C. Computer-augmented retrieval of HELP information for novice users. In Proceedings of the 26th Annual Meeting. Santa Monica, California: Human Factors Society, 1982, 79-82.
- Demers, R. A. System design for usability. Communications of the ACM, 1981, 24(8), 494-501.
- Dray, S. M., Ogden, W. G., and Vestewig, R. E. Measuring performance with a menu-selection human-computer interface. In Proceedings of the 25th Annual Meeting. Santa Monica, California: Human Factors Society, 1981, 746-748.
- Durding, B. M., Becker, C. A., and Gould, J. D. Data organization. Human Factors, 1977, 19(1), 1-14.

- Dwyer, B. A user-friendly algorithm. Communications of the ACM, 1981, 24(9), 556-561.
- Ehrenreich, S. L. Query languages: Design recommendations derived from the human factors literature. Human Factors, 1981, 23(6), 709-725.
- Elkerton, J., Williges, R. C., Pittman, J. A., and Roach, J. Strategies of interactive file search. In Proceedings of the 26th Annual Meeting. Santa Monica, California: Human Factors Society, 1982, 83-86.
- Engel, S. E. and Granda, R. E. Guidelines for Man/Display Interfaces, Technical Report TR 00.2720. Poughkeepsie, New York: IBM, December 1975.
- Gade, P. A., Fields, A. F., Maisano, R. E., Marshall, C. F., and Alderman, I. N. Data entry performance as a function of method and instructional strategy. Human Factors, 1981, 23(2), 199-210.
- Galitz, W. O. Human Factors in Office Automation. Atlanta, Georgia: Life Office Management Association, Inc., 1980.
- Geiser, G., Schumacher, W., and Berger, L. Talking keyboard for user guidance in multifunction systems. In Proceedings of the 26th Annual Meeting. Santa Monica, California: Human Factors Society, 1982, 436-439.
- Goodwin, N. C. INTRO -- In Which a Smart Terminal Teaches Its Own Use, ESD-TR-74-374, Electronic Systems Division, AFSC, Hanscom AFB, MA 01731, March 1974. (NTIS No. in process)
- Goodwin, N. C. A user-oriented evaluation of computer-aided message handling. In Proceedings of the 24th Annual Meeting. Santa Monica, California: Human Factors Society, 1980, 585-589.
- Goodwin, N. C. Effect of interface design on usability of message handling systems. In Proceedings of the 26th Annual Meeting. Santa Monica, California: Human Factors Society, 1982, 69-73.
- Granda, R. E., Teitelbaum, R. C., and Dunlap, G. L. The effect of VDT command line location on data entry behavior. In Proceedings of the 26th Annual Meeting. Santa Monica, California: Human Factors Society, 1982, 621-624.

- Gregory, M. and Poulton, E. C. Even versus uneven right-hand margins and the rate of comprehension in reading. Ergonomics, 1970, 13, 427-434.
- Hamill, B. W. Experimental document design: Guidebook organization and index formats. In Proceedings of the 24th Annual Meeting. Santa Monica, California: Human Factors Society, 1980, 480-483.
- Hanson, R. H., Payne, D. G., Shiveley, R. J., and Kantowitz, B. H. Process control simulation research in monitoring analog and digital displays. In Proceedings of the 25th Annual Meeting. Santa Monica, California: Human Factors Society, 1981, 154-158.
- Hollingsworth, S. R. and Dray, S. M. Implications of post-stimulus cueing of response options for the design of function keyboards. In Proceedings of the 25th Annual Meeting. Santa Monica, California: Human Factors Society, 1981, 263-265.
- Liebelt, L. S., McDonald, J. E., Stone, J. D., and Karat, J. The effect of organization on learning menu access. In Proceedings of the 26th Annual Meeting. Santa Monica, California: Human Factors Society, 1982, 546-550.
- Martin, J. Design of Man-Computer Dialogues. Englewood Cliffs, New Jersey: Prentice-Hall, 1973.
- Martin, J. The Wired Society. Englewood Cliffs, New Jersey: Prentice-Hall, 1978.
- MIL-H-48655B. Military Specification: Human Engineering Requirements for Military Systems, Equipment and Facilities. Washington: Department of Defense, 31 January 1979.
- MIL-STD-1472B. Military Standard: Human Engineering Design Criteria for Military Systems, Equipment and Facilities. Washington: Department of Defense, 31 December 1974.
- MIL-STD-1472C. Military Standard: Human Engineering Design Criteria for Military Systems, Equipment and Facilities. Washington: Department of Defense, 2 May 1981.
- Miller, D. P. The depth/breadth tradeoff in hierarchical computer menus. In Proceedings of the 25th Annual Meeting. Santa Monica, California: Human Factors Society, 1981, 296-300.

M-P REFERENCES

- Miller, R. B. Response time in user-system conversational transactions. In Proceedings of the AFIPS Fall Joint Computer Conference, 1968, 33, 267-277.
- Morrill, C. S. Computer-aided instruction as part of a management information system. Human Factors, 1967, 9(3), 251-256.
- Morrill, C. S. and Davies, B. L. Target tracking and acquisition in three dimensions using a two-dimensional display surface. Journal of Applied Psychology, 1961, 45, 214-221.
- Morrill, C. S., Goodwin, N. C., and Smith, S. L. User input mode and computer-aided instruction. Human Factors, 1968, 10(3), 225-232.
- Moses, F. L. and Ehrenreich, S. L. Abbreviations for automated systems. In Proceedings of the 25th Annual Meeting. Santa Monica, California: Human Factors Society, 1981, 132-135.
- NASA (National Aeronautics and Space Administration). Spacelab Experiment Computer Application Software (ECAS) Display Design and Command Usage Guidelines, Report MSFC-PROC-711. George C. Marshall Space Flight Center, Alabama, January 1979.
- Neal, A. S. and Emmons, W. H. Operator corrections during text entry with word processing systems. In Proceedings of the 26th Annual Meeting. Santa Monica, California: Human Factors Society, 1982, 625-628.
- Noyes, L. The positioning of type on maps: The effect of surrounding material on word recognition. Human Factors, 1980, 22(3), 353-360.
- Palme, J. A human-computer interface for non-computer specialists. Software -- Practice and Experience, 1979, 9, 741-747.
- Parrish, R. N., Gates, J. L., Munger, S. J., Grimmer, P. R., and Smith, L. T. Development of Design Guidelines and Criteria for User/Operator Transactions with Battlefield Automated Systems, Phase II Final Report: Volume II, Prototype Design Handbook for Combat and Materiel Developers, Report WF-80-AE-00. Alexandria, Virginia: US Army Research Institute for the Behavioral and Social Sciences, February 1982.
- Parsons, H. M. The scope of human factors in computer-based data processing systems. Human Factors, 1970, 12(2), 165-175.

- Penniman, W. D. Past chairman's message. SIG Newsletter No. UOI-10. Washington: American Society for Information Science, May 1979.
- Pew, R. W. and Rollins, A. M. Dialog Specification Procedures, Report 3129 (revised). Cambridge, Massachusetts: Bolt Beranek and Newman, 1975.
- Ramsey, H. R. and Atwood, M. E. Human Factors in Computer Systems: A Review of the Literature, Technical Report SAI-79-111-DEN. Englewood, Colorado: Science Applications, Inc., September 1979. (NTIS No. AD A075 679)
- Ramsey, H. R. and Atwood, M. E. Man-computer interface design guidance: State of the art. In Proceedings of the 24th Annual Meeting. Santa Monica, California: Human Factors Society, 1980, 85-89.
- Ramsey, H. R., Atwood, M. E. and Kirshbaum, P. J. A Critically Annotated Bibliography of the Literature of Human Factors in Computer Systems, Technical Report SAI-78-070-DEN. Englewood, Colorado: Science Applications, Inc., May 1978. (NTIS No. AD A058 081)
- Reisner, P. Use of psychological experimentation as an aid to development of a query language. IEEE Transactions on Software Engineering, 1977, SE-3, 218-229.
- Rogers, J. G. and Armstrong, R. Use of human engineering standards in design. Human Factors, 19(1), 15-23, 1977.
- Rogers, J. G. and Pegden, C. D. Formatting and organization of a human engineering standard. Human Factors, 19(1), 55-61, 1977.
- Savage, R. E., Habinek, J. K., and Blackstad, N. J. An experimental evaluation of input field and cursor combinations. In Proceedings of the 26th Annual Meeting. Santa Monica, California: Human Factors Society, 1982, 629-633.
- Seibel, R. Data entry devices and procedures. In Van Cott, H. P. and Kinkade, R. G. (Eds.) Human Engineering Guide to Equipment Design. Washington: U. S. Government Printing Office, 1972, 311-344.
- Shneiderman, B. Software Psychology: Human Factors in Computer and Information Systems. Cambridge, Massachusetts: Winthrop Publishers, Inc., 1980.

S REFERENCES

- Shneiderman, B. A note on human factors issues of natural language interaction with database systems. Information Systems, 1981, 6(2), 125-129.
- Sidorsky, R. C. and Parrish, R. N. Guidelines and criteria for human-computer interface design of battlefield automated systems. In Proceedings of the 24th Annual Meeting. Santa Monica, California: Human Factors Society, 1980, 98-102.
- Smith, S. L. Angular estimation. Journal of Applied Psychology, 1962, 46, 240-246. (a)
- Smith, S. L. Color coding and visual search. Journal of Experimental Psychology, 1962, 64(5), 434-440. (b)
- Smith, S. L. Color coding and visual separability in information displays. Journal of Applied Psychology, 1963, 47(6), 358-364. (a)
- Smith, S. L. Man-computer information transfer. In Howard, J. H. (Ed.) Electronic Information Display Systems, 284-299. Washington: Spartan Books, 1963.
- Smith, S. L. Requirements definition and design guidelines for the man-machine interface in C3 system acquisition, Technical Report ESD-TR-80-122. Bedford, Massachusetts: USAF Electronic Systems Division, June 1980. (NTIS No. AD A087 258)
- Smith, S. L. Man-Machine Interface (MMI) Requirements Definition and Design Guidelines: A Progress Report, Technical Report ESD-TR-81-113. Bedford, Massachusetts: USAF Electronic Systems Division, February 1981. (NTIS No. AD A096 705) (a)
- Smith, S. L. Design guidelines for the user-system interface of on-line computer systems: A status report. In Proceedings of the 25th Annual Meeting. Santa Monica, California: Human Factors Society, 1981, 509-512. (b)
- Smith, S. L. "User-system interface". Human Factors Society Bulletin, 1982, 25(3), 1. (a)
- Smith, S. L. User-System Interface Design for Computer-Based Information Systems, Technical Report ESD-TR-82-132. Bedford, Massachusetts: USAF Electronic Systems Division, April 1982. (NTIS No. AD A115 853) (b)

- Smith, S. L. and Aucella, A. F. Numbering formats for hierarchic lists. In Proceedings of the 26th Annual Meeting. Santa Monica, California: Human Factors Society, 1982, 538-541.
- Smith, S. L., Farquhar, B. B., and Thomas, D. W. Color coding in formatted displays. Journal of Applied Psychology, 1965, 49(6), 393-398.
- Smith, S. L. and Goodwin, N. C. Computer-generated speech and man-computer interaction. Human Factors, 1970, 12(2), 215-223.
- Smith, S. L. and Goodwin, N. C. Alphabetic data entry via the Touch-Tone pad: A comment. Human Factors, 1971, 13(2), 189-190. (a)
- Smith, S. L. and Goodwin, N. C. Blink coding for information display. Human Factors, 1971, 13(3), 283-290. (b)
- Smith, S. L. and Goodwin, N. C. Another look at blinking displays. Human Factors, 1972, 14(4), 345-347.
- Smith, S. L. and Thomas, D. W. Color versus shape coding in information displays. Journal of Applied Psychology, 1964, 48(3), 137-146.
- Tullis, T. S. An evaluation of alphanumeric, graphic, and color information displays. Human Factors, 1981, 23(5), 541-550.
- Whalley, P. C. and Fleming, R. W. An experiment with a simple recorder of reading behaviour. Programmed Learning and Educational Technology, 1975, 12(2), 120-124.
- Woodson, W. E. Human Factors Design Handbook. New York: McGraw-Hill, 1981.
- Wright, P. and Reid, F. Written information: Some alternatives to prose for expressing the outcomes of complex contingencies. Journal of Applied Psychology, 1973, 57(2), 160-166.

ABORT - A capability that cancels all user entries in a defined transaction sequence.

Background Fields - See "Data Field Labels".

BACKUP - A capability that returns the user to the first display in a defined transaction sequence.

CANCEL - A capability that regenerates (or re-initializes) the current display without processing any entries or changes made by the user.

Category - A grouping of data values along a dimension for operational purposes. For example, an air traffic controller might be instructed to implement the same procedures for all aircraft with speeds in the category of 600 to 800 knots. See also "Value".

Command Language - A type of dialogue in which the user formulates control entries with minimal prompting by the computer.

Control Entry - User input for controlling a transaction sequence, such as function key activation, menu selection, command entry, etc.

Cursor - A marker on the display screen that indicates the "current" position for attention, which may designate a displayed item. The cursor may be positioned under computer control or moved by the user.

Data - The raw materials from which a user extracts information.

Data Base - The collection of data that is stored in the central computer for relatively long periods of time.

Data Display - Output of data from a computer to its users. Generally, this phrase denotes visual output, but it may be qualified to indicate a different modality, such as an "auditory display".

Data Entry - User input of data for computer processing.

Data Field - An area of the display screen reserved for user entry of a data item.

Data Field Label - An area of the display screen that serves as a prompt for entering a data item. It usually cannot be changed by a user.

Data Item - A set of characters of fixed or variable length that forms a single unit of data. Examples of a data item might be a person's last name or zip code. Sometimes a data item may contain only a single character. Data items may be entered by the user or may be supplied by the computer.

Data Protection - Functional capabilities that guard against unauthorized data access and tampering, and also against user errors.

Data Transmission - Computer-mediated communication that transfers data from one user to another. The data transmitted may include numbers, words and pictures.

Data Validation - Functional capabilities that check data entry items for correct content or format.

Default Value - A predetermined, frequently used, value for a data or control entry, intended to reduce required user action.

Dialogue - A structured series of interchanges between a user and a computer terminal. Dialogues can be computer-initiated, e.g., question and answer, or user-initiated, e.g., command language.

Dimension - A type of data that may contain different values at different times. For example, dimensions for an aircraft include its heading, speed and altitude. See also "Variable".

Display - See "Data Display".

ENTER - An explicit user action that effects computer processing of user entries. For example, after typing a series of numbers, a user might press a specially marked ENTER key that will add them to a data base, subject to data validation.

Entry - See "Data Entry".

Field - See "Data Field".

File - A collection of data, treated as a single unit, that is stored in the computer.

Form Filling - A type of dialogue in which the computer displays forms containing labeled fields for data entry by a user.

- Function** - A computer-supported capability provided to users as an aid for task performance. Examples of functions are position designation or direction designation.
- Hard Copy** - A printed paper display of data processed by the computer.
- HELP** - A capability that displays information upon user request for on-line guidance. HELP may inform a user generally about system capabilities, or may provide more specific guidance in data handling transactions.
- Information** - Organized data that users need to successfully perform their tasks. Information serves as an answer to a user's question about data. It is used here to refer to the effective processing and arrangement of data.
- Information System** - A computer-supported, task-oriented tool designed to help perform defined data handling tasks.
- Interface** - See "User-System Interface".
- Interaction** - See "Transaction".
- Input** - See "Control Entry" and "Data Entry".
- Job** - A set of responsibilities and activities that are defined for each system user.
- Menu Selection** - A type of dialogue in which the user selects one item out of a list of displayed alternatives.
- Message** - Data that are transmitted from one computer user to another as a discrete transaction.
- Natural Language** - A type of dialogue in which users formulate control entries in their natural language, e.g., English, Spanish, French.
- Operator** - See "User".
- Output** - See "Data Display".
- Page** - The data appearing at one time on a single display screen.
- Protected Field** - See "Data Field Label".

Query Language - A type of dialogue in which users formulate control entries for extracting specified data from a data base.

Question and Answer - A type of dialogue in which the computer displays questions, one at a time, to the user.

Screen - See "Page".

Sequence Control - Logic and means by which user actions and computer responses are linked to become coherent transactions.

Suspense File - A temporary collection of data saved by the computer for later use.

System - See "Information System".

Task - A series of transactions that comprises part of a user's defined job.

Terminal - An input/output device used to enter and display data. Data are usually entered via a keyboard, and are usually displayed via a video screen ("soft copy") or a printer ("hard copy").

Transaction - An action by the user followed by a response from the computer. Transaction is used here to represent the smallest functional "molecule" of user-system interaction.

User - Any person who uses an information system in performing his/her job.

User Guidance - Computer prompts and feedback that aid users in performing their tasks. Examples include data field labels, alarm or alert signals, error messages, and HELP messages.

User-System Interface - All aspects of information system design that affect a user's participation in data handling transactions.

Value - Specific data for a particular dimension or variable. For example, values for an aircraft's speed might be 800 knots during one observation and 500 knots during another. See also "Category".

Variable - See "Dimension".

Work Station - The general physical environment in which the user works. It includes such things as computer terminals, source documents, desks, chairs, and lighting.

A

abbreviations
 commands 3.15-4
 3.2-16
 data display — 2.1.1-22/-26
 data entry 1.0-14/-20
acknowledgment
 - see feedback
active voice 2.1.1-11
 4.0-19
affirmative statements 2.1.1-10
 4.0-18
alarms 3.6
 4.1.9
 4.3-17
 6.0-8
 6.5-17
alignment
 - see justification
assistance, on-line
 - see user guidance
auditory
 coding 2.7-34/-35
 displays 1.4-9
 signals — 1.4-12
automatic 2.9-1
 cursor placement 1.1-11
 1.1-19
 1.1-22
 1.4-7
 1.4-25
 3.2-7/-8
 4.3-12
 4.4-12
 6.2-5
 — 5.1-1
 — 5.4-3
 — 1.4-11
 — 1.5-5/-6
 — 4.5-2
 formatting — 5.1-1
 — 5.4-3
 justification — 1.4-11
 — 1.5-5/-6
awareness, user — 4.5-2

B

BAUD rate
 - see display rate

C

category 2.7-2
 2.7-13
 2.7-23
 2.7-27
 2.7-35
coding
 - see display coding
color coding
 - see display coding
command language 3.1.3-9
 3.1.7
command stacking
 - see stacking
communication
 - see data transmission
compatibility 1.4-23/-24
 3.0-3/-4
 3.1-1
 3.1.4-13
 3.1.6-2
 6.5-5
 — 2.0-6
computer control — 2.0-6
concise
 - see information
consistent, consistency
 - see format
 grammar
 grouping
 user actions
 wording
context definition 2.8-3/-6
 3.0-17
 3.1.3-22
 3.4
 4.2-8/-9
 4.4-10
 6.0-4
 — 4.4-5
continuation — 4.4-5
 - also see page
contractions 2.1.1-21
control, computer
 - see computer control
control, user
 - see user control

- conventional usage
 - see wording
 (familiar/meaningful)
- cues
 - see prompts
- cursor
 multiple 1.1-16/-18
 step-size 1.1-9/-10
 windowing 2.8-8
 - also see automatic
 position designation
- D
- data access
 data protection 6.2
 - also see data display
- data change
 data protection 6.3
 - also see data entry
- data dimension
 - see unit of measurement
- data display 2.0
- data entry 1.0
- data field 1.4-8/-10
- data field labels
 - see labels
- data forms
 data display 2.1.2
 data entry 1.4
- data items 1.0-11/-13
 2.1.2-11/-12
 2.6-3
- data protection 6.0
- data selection
 data display 2.2
- data transmission 5.0
 data protection 6.4
- data type 2.1
 5.1
- data validation
 data entry 1.7
 data protection 6.3-17/-18
 - also see user confirmation
- decimal numbers 1.5-6
 2.1.3-3
- dedicated functions 1.4-2
- ABORT 3.3-6
- BACKUP 3.3-4
 6.3-13
- CANCEL 3.3-3
- CONFIRM 1.0-10
 3.5-11
 6.5-19
- DITTO 1.5-7
- END 3.3-7
- ENTER 1.0-8/-9
 1.1-23
 3.5-6
 6.5-9
- RESTART 3.3-5
- STEP 3.2-13
- default 1.7-6/-9
 3.2-12
 6.3-7
 6.5-12
- deferral, user 1.7-2/-3
- definitions
 codes 2.7-8
 2.7-27
 4.4-17
- delay, system 4.1-3
 4.2-3
 4.2-7
 - also see response time
- delimiter 1.4-3/-4
 3.1.5-13
 3.2-17
- destination
 data transmission 5.3
- dialogue type 3.1
- dictionary
 abbreviations 2.1.1-25
 4.4-16
- dimensions 2.7-15/-16
- direction designation 1.2
- disabled
 function keys 3.1.4-10
 6.5-7
- keyboard lockout 3.0-10
 4.1-4

- display coding** 2.7
 alphabetic 2.7-3/-5
 auditory 2.7-34/-35
 blink 2.7-30/-33
 brightness 2.7-21/-23
 color 2.7-23/-29
 consistent 2.7-14
 3.1.3-11
 4.0-8
distinctive — 3.1.3-12
 4.3-17
familiar/meaningful — 2.7-6/-7
 2.7-10
 2.7-28
 4.0-7
labels — 1.4-5
line 2.7-15/-18
menus 3.1.3-5
 3.2-8/-9
shape — 2.7-13/-14
size 2.7-19/-20
symbols 2.7-11/-12
 2.7-31
display coverage — 2.8
display density 2.6
display frame
 - see page
display freeze 2.9-5/-8
display generation 2.4
display partitioning 2.5
display rate 2.1.1-9
 2.9-2/-4
display screen
 - see page
display suppression 2.10
display update — 2.9
display windows 2.5-3/-4
 2.5-6
distinctive
 - see format
 wording
documentation 4.3-12
 4.4-13
- E**
editing 1.0-5
 1.4-2
 3.5-2
 4.3-14
 6.0-6
 6.3-10
error correcting — 3.1.5-17
 3.5-7
 3.5-13/-14
 6.3-11/-12
 6.3-14
 6.5-20
error editing — 3.5-2
error feedback 4.3
 - also see feedback
error management 3.5
error records 4.5-6
 - also see record keeping
 user records
- F**
familiar, familiarity
 - see coding
 wording
feedback 1.0-2
 1.1-5
 3.0-11/-15
 3.1.3-26
 3.1.4-15
 3.6-3/-4
 4.1-1
 5.5
 6.3-19
 6.5-10
 - also see error feedback
 routine feedback
field
 - see data field

F-H INDEX

flexibility	1.4-2	G	
	3.0-1	grammar	
	3.3-1	consistent	2.1.2-4
	4.0-22		4.0-21
	4.4-25	- also see text	
	5.0-6	graphic data	1.6
	5.2-1/-2	graphic interaction	3.1.7
	5.3-1	grouping	
	5.3-4	consistent	2.3-1
	5.4-2		3.1.3-25
	6.3-13		3.2-15
	6.5-10	logical	1.4-24
- also see interrupt			2.1.1-14
multi-layered			2.1.3-5
user control			2.3-2/-6
user defined			2.7-5
user skills			3.0-18
stacking			3.1.3-13/-16
form filling	3.1.2		3.1.6-2
format			4.4-3/-4
consistent	1.4-13	memory aids	1.0-13
	1.5-4		2.0-7
	2.1.1-1		2.1.1-16/-17
	2.1.2-8		
	2.1.2-10/-11		
	2.2-2		
	2.5-1/-8	H	
	2.7-8	hardcopy	
	2.7-18	- see printout	
	3.1.3-7	HELP	4.4-19/-24
	3.1.3-23		4.5-7
	3.1.5-2	hierarchical	2.1.3-8
	4.0-5	- also see menu selection	
	4.0-10	highlighting	2.1.2-7
	4.4-8		2.7-1
distinctive	1.5-3		3.4-6
	2.1.2-5		4.0-15
	2.1.3-6		4.2-10
	2.5-3		4.3-17
	2.5-5		5.1-4
	3.0-19		5.6-5
	3.1.4-12	labels	1.4-22
	3.4-7		2.1.2-7
	4.0-11		
	4.4-17	HOME	
lists	2.1.1-13	cursor location	1.1-6
	2.1.1-15	menu	3.2-2
tables	2.1.3-8/-11		4.4-2
function keys	3.1.4		

I		L	
index	2.1.3-10	labels	4.0-14
	4.4-14/-15	data field	1.4-5/-7
information	2.6-1		1.4-13/-21
	2.8-1		2.1.2-1/-9
	3.1.3-18		5.1-3
	4.4-1	display page	2.2-1/-2
minimize display	2.0-1		2.5-2
	2.6-2		2.5-7
	3.1.3-17		2.9-6
	3.2-11		4.2-4
	4.0-3		6.2-2/-3
	4.3-5	function keys	3.1.4-4/-6
interrupt	3.3		4.0-13
		menus	3.1.3-16
			4.4-4
		tables	1.5-1/-4
			2.1.3-1/-2
			2.1.3-6
J		layout	
job aids	4.4	- see format	
justify, justification		length	
alphabetic	2.1.3-4	abbreviation	1.0-16
decimal point	1.5-6	code	2.7-9
	2.1.3-3	data item	1.0-11/-12
left	1.5-4		2.1.2-12
	1.5-6	lists	
	2.1.1-3	- see text	
	2.1.3-4	location	
numeric	1.5-6	- see format	
	2.1.3-3	logical	3.0-18
right	2.1.3-3	- also see grouping	
	2.1.3-8	LOG OFF	3.5.-12
			6.5-16
		LOG ON	4.0-4
			4.1-2/-3
			6.1-1/-2
			6.2-1
			6.3-1
K		loss prevention	
keyboard actions		data protection	6.5
- see user actions			
keyboard lockout			
- see disabled			

M

macro command 3.2-19
 meaningful
 - see wording
 memory aids 2.0-7
 5.0-3
 - also see grouping
 menu selection 3.1.3
 by-pass — 3.1.3-27
 hierarchical 3.1.3-14
 3.1.3-19/-24
 4.4-4
 - also see coding labels
 mode
 - see operational mode
 multi-layered 3.1.5-9
 3.3.5-9
 4.3-7
 4.4-23
 multiple cursors
 - see cursor

N

natural language — 3.1.7
 numbering 2.1.3-7
 2.8-6

O

operational mode 3.4-4
 4.2-8
 6.0-4
 6.5-13
 options, implicit — 4.4-6
 - also see user control
 options, listing
 - see menu selection
 ordering
 - see grouping
 other data processing
 data entry 1.8

P

page 2.2-1
 2.2-5
 2.8-1/-6
 4.2-4/-5
 - also see display partitioning labels
 paging 2.8-2/-4
 pointing
 direct 3.1.3-3/-4
 3.2-7
 - also see position designation
 position designation 1.1
 - also see cursor
 printout 2.4-2/-3
 4.2-6
 5.3-3
 6.2-6
 6.4-5
 prompts
 implicit 1.4-8/-10
 1.4-17/-18
 4.4-11
 system — 3.5-3
 4.3-14
 4.4-7
 4.4-22
 user requested — 3.1.5-10
 3.2-5
 4.4-9
 - also see labels
 proximity 1.0-1
 2.1.5-1
 punctuation
 command entry 3.1.5-12
 labels — 1.4-17
 text display — 2.1.1-4
 2.1.1-6
 2.1.1-21
 2.1.1-26

- Q**
- query language 3.1.6
 - question and answer 3.1.1
 - queuing
 - data transmission 5.7
- R**
- reading rate
 - 2.2.2-9
 - 2.9-2/-4
 - record keeping
 - data transmission 5.7
 - also see error records
 - user records
 - response time
 - 1.0-7
 - 1.1-5
 - 1.1-7
 - 2.4-1
 - 3.0-8/-12
 - 3.1.2
 - 4.2-2/-3
 - 4.2-2/-7
 - 4.3-11
 - routine feedback — 4.2
- S**
- screen
 - see page
 - scrolling
 - 2.8-2/-4
 - 2.8-7
 - 2.8-10/-11
 - self-paced
 - see user paced
 - sentences
 - see text (prose)
 - sequence
 - see grouping
 - sequence control 3.0
 - shape coding
 - see coding
 - simultaneous users
 - 3.0-20
 - 4.1-5
 - 6.5-2
- source**
- data transmission — 5.2
- spacing**
- 1.0-25
 - 2.1.2-6
 - 2.1.3-9/-11
 - 2.7-12
 - 3.1.5-14
- stacking**
- command
 - 3.1.5-11
 - 3.2-14/-18
 - 3.5-4/-5
 - menus 3.1.3-28
- standard procedures**
- see user actions (consistent)
- standards**
- see consistent
- statements**
- see text (prose)
- status information** — 4.1
- status, system**
- 2.9-6
 - 5.3-2
 - also see status information
- symbols**
- see coding
- system interrogation**
- see user confirmation
- T**
- tabbing
 - 1.1-21
 - 1.4-12
 - tables
 - 2.8-4
 - also see tabular data
 - tabular data — 1.5
 - task oriented
 - 3.0-18
 - 4.4-21
 - also see wording
 - (familiar/meaningful)
 - terms, terminology
 - see wording
 - text
 - data display 2.1.1
 - data entry — 1.3
 - lists — 2.1.1-13/-17
 - 2.8-3
 - prose — 2.1.1-2/-12

T-U INDEX

training	4.4-25/-27	user actions (cont.)	
transaction selection	3.2	minimizing	1.0-3
transmission control			1.0-22/-23
data transmission	5.4		1.1-20
truncation			1.5-7
- see abbreviation			3.0-2
			3.1.3-2
			3.1.3-20
			3.1.3-24
			3.1.4-14
			5.0-2
U		- also see abbreviation	
unit of measurement	1.4-19	user confirmation	1.0-10
	1.4-21		1.0-19
	2.1.2-9		1.1-23
	2.1.3-2		1.7-8
user actions			3.1.5-15/-16
consistent	3.0-16		3.5-3
	3.2-13		3.5-8/-9
	3.6-2		4.3-15/-16
	4.0-1		4.4-22
	4.4-20		6.0-7
	5.0-1		6.3-7
	6.0-2		6.3-15
	6.5-6		6.3-20
easy	— 1.1-14		6.4-2
	3.1.3-21		6.5-11
	3.1.4-9		6.5-18
	3.1.5-8	user control	— 1.7-5
	3.6-2		2.1.4-1
	5.6-1		2.9-1
	6.3-6		2.9-5
explicit	— 1.0-8		2.10-1/-2
	1.1-14		3.0-6
	1.7-3		3.2-1
	3.0-5		3.5-15
	3.5-6/-7		5.1-2
	3.5-10/-11		5.4-6
	4.0-2		6.2-3
	4.3-16	experience	— 2.0-3
	5.0-7		2.1.2-7
	6.0-3		4.4-9
	6.3-8/-10	user defined	— 1.7-6
	6.3-14		3.1.5-7
	6.3-20		3.2-19
	6.5-13		3.6-1
	6.5-19		5.5-4
			6.1-3

user error		wording	
- see error		concise	2.1.1-9
user feedback		consistent	1.4-6
- see feedback			2.1.1-19/-20
user guidance	4.0		2.8-11
user identification			3.1.3-25
data protection	6.1		3.1.4-7
user paced	1.0-6		3.1.5-5
	3.0-7		4.0-6
- also see user control		distinctive	1.4-14
user records	3.4-2		2.1.2-3
	4.4-18		3.1.4-4
	4.5		3.1.5-6
	6.2-7		3.3-2
	6.3-2		6.5-15
user requested	5.6-2	familiar/meaningful	1.4-16
- also see prompts			1.5-2
user skills	2.0-4		2.1.1-18
	3.0-4		3.1.4-8
	3.1-1		3.1.5-3
	4.0-22		3.2-10
			4.0-16/-17
			4.4-3
V			5.4-1
vocabulary		neutral	4.3-6
- see wording		temporal sequence	2.1.1-12
voice			4.0-20
- see auditory			
W		Z	
windowing	2.8-7/-9	zero	1.0-24
	2.8-11		

USI Design Guidelines -- Changes/Additions

Code number, if referring to a guideline in this report: _____

Proposed wording for guideline: _____

Example: _____

Exception: _____

Comment _____

References to previously published recommendations or supporting data:

Please mail to:
Sidney L. Smith
The MITRE Corporation
Bedford, MA 01730 USA

Your name: _____

Date: _____