

Chapter 19

Graphical User Interfaces

Aaron Marcus
 Aaron Marcus and Associates
 Emeryville, California, USA

19.1 Introduction	424	19.7 Design Guidelines	432
19.2 Applications and GUIs	424	19.7.1 Design Characteristics.....	432
19.3 Windowing Systems	425	19.7.2 Design Guidelines	432
19.3.1 Macintosh.....	425	19.7.3 Order and Chaos.....	433
19.3.2 NeXTStep.....	425	19.7.4 Consistency	433
19.3.3 Open LOOK.....	425	19.7.5 External Consistency: Leverage	
19.3.4 OSF/Motif	429	Known Design Techniques	433
19.3.5 Microsoft Windows and OS/2		19.7.6 GUI Screen Layout.....	433
Presentation Manager	429	19.7.7 Visual Relationships	433
19.3.6 Custom GUIs.....	429	19.7.8 Navigability.....	433
19.4 Windowing System Architectures	430	19.7.9 Economy.....	434
19.4.1 Kernel-Based Architecture.....	430	19.7.10 Balanced Communication	435
19.4.2 Client-Server Based Architecture	430	19.7.11 Layout	435
19.5 Window Management	430	19.7.12 Legibility	435
19.5.1 Tiled Windows.....	430	19.7.13 Backgrounds.....	435
19.5.2 Overlapping Windows	430	19.7.14 Readability	435
19.5.3 Cascading Windows.....	431	19.7.15 Typography	435
19.6 Windowing System Components	431	19.7.16 Symbolism.....	436
19.6.1 Windows	431	19.7.17 Multiple Views.....	436
19.6.2 Menus.....	431	19.7.18 Advantages of Color	436
19.6.3 Controls.....	431	19.7.19 Color Similarity.....	436
19.6.4 Dialogue Boxes	431	19.7.20 Color Consistency	436
19.6.5 Modeless Dialogues	432	19.7.21 Color Economy: Redundancy	436
19.6.6 Modal Dialogues	432	19.7.22 Color Economy: Enhancement.....	437
19.6.7 Control Panels.....	432	19.7.23 Color Economy: Sequencing.....	437
19.6.8 Query Boxes.....	432	19.7.24 Color Emphasis	437
19.6.9 Message Boxes.....	432	19.7.25 Color Emphasis: Hierarchy	437
19.6.10 Mouse and Keyboard Interface	432	19.7.26 Color Emphasis: Viewing Differences ..	437
		19.7.27 Color Communication: Central and	
		Peripheral Colors	437
		19.7.28 Color Communication: Combinations ...	437
		19.7.29 Color Communication: Area	437
		19.7.30 Color Communication: High Chroma	
		and Spectrally Extreme Colors	438
		19.7.31 Color Communication: Chroma	
		and Value	438
		19.7.32 Color Communication: Combinations to	
		Avoid.....	438
		19.7.33 Color Communication: For	
		Dark-Viewing.....	438
		19.7.34 Color Communication: For	
		Light-Viewing	438

19.7.35 Color Communication: Interactions.....	438
19.7.36 Color Symbolism: Using Color Codes.....	438
19.7.37 Color Symbolism: Connotations.....	438
19.7.38 Keyboard Short-Cuts.....	438
19.8 Rules of Thumb	438
19.9 Conclusions	439
19.10 Acknowledgments.....	439
19.11 Bibliography (Including Cited References)	440

19.1 Introduction

Human-computer communication and interactivity in previous decades was a limited exchange of alphanumeric characters. Today, advanced graphical user interfaces (GUIs sophisticated interactive displays that enable novice, intermediate, and expert users to work more productively. Many of these GUIs use windows, icons, menus, and pointing devices (WIMPs) with two-dimensional or slightly three-dimensional (beveled edges and overlapping planes) to achieve their communication and interactivity goals. In particular, many classical GUIs for typical office applications simulate a “desktop” environment in the display.

To be successful, the GUI must accurately and efficiently relate to the tasks, workflow, objectives, education, personality, and culture of the user. The GUI specifically must provide the following design components in a functional and aesthetic form (also referred to as performance- and preference-oriented form):

Metaphors (essential concepts communicated through terms and images)

Mental model (organization of data, functions, tasks, and roles)

Navigation of the mental model (menus, icons, dialogue boxes, and windows)

Appearance (visual, auditory, and verbal properties of controls and ornamental background)

Interaction (behavior of interactive screen controls and physical input and output display devices).

Users can appreciate and take advantage of quality in each of these components. Good organization of contents, economical means to express each compo-

nent, effective use of visual elements, and efficient interaction all lead to friendlier and more usable systems.

This chapter discusses various aspects of GUI environments focusing primarily on windowing systems, user-interface elements, and window management. Many tips and suggestions are provided at the end of the chapter that will aid in the understanding, evaluation, and development of GUI applications. A thorough discussion cross-comparing GUIs and their components is available in (Marcus, Smilovich, and Thompson, 1995). A thorough discussion of GUI design tips is available in (Marcus, 1992) and (Marcus, 1995)

19.2 Applications and GUIs

Many applications present designers with a special challenge: large collections of functions that act upon very large amounts of data in complex ways. To present their data and functions, many programs have converted to one or more standard commercial GUI paradigms such as Windows, Macintosh, Motif, etc., many of which are discussed individually in this chapter.

Many multimedia applications and interactive content available on the Internet and other private online services use non-standard GUIs. Whether for standard or customized GUIs, GUI designers should be especially sensitive to how clearly the menu hierarchy is organized and to how clearly dialogue boxes are labeled and laid out, because much of the user’s mental work takes place in examining and interacting with these GUI components. For many of these GUI-based products, the use of icons to represent objects, structures, or processes has become a significant feature. GUI designers also should examine how clearly icons are designed and labeled.

Current GUI building tools enable developers to construct applications more quickly than ever before. However, the tools do not ensure that the applications are automatically well-designed. To provide some background, this chapter discusses some of the standard GUI paradigms and principles of good GUI design.

The GUI windowing system is similar to an operating system. Instead of file systems or central processing-unit cycles, however, the windowing system manages resources such as screen space and input devices. In GUIs, the windowing system acts as a front end to the operating system by shielding developers, and users, from the abstract and often confusing syntax and vocabulary of a keyboard-oriented command language.

19.3 Windowing Systems

Each of the windowing systems discussed briefly below has unique features and a place in the history of GUIs. Figure 1 through Figure 6 shows applications operating within some of these GUI paradigms.

19.3.1 Macintosh

The Apple Macintosh appeared in 1984 as the first mass-marketed computer to feature a high-resolution, bit-mapped graphic display and a direct-manipulation interaction style. Its windowing system is built on top of a proprietary library of operating system and user interface tool kit routines in the Macintosh read-only memory (ROM).

The classic Macintosh GUI was a single-tasking system with a high level of responsiveness and a very simple model of the designer's tasks. Current versions permit multiple applications to be opened and operated. Apple Computer has succeeded in creating widespread acceptance among third-party software developers for their standard GUI components. As a result, knowledge about familiar Macintosh applications can ease the task of learning new ones.

The Macintosh was the first computer system with a GUI to gain widespread market acceptance and experience significant commercial success. Its popularity, particularly in non-technical market segments traditionally unreceptive toward computing, can be attributed in large part to Apple's commitment to the creation of a consistent and user-supportive user interface. Because of its historical precedence and market penetration, the Macintosh established the standard of interaction by which GUIs are judged. The degree of responsiveness to the actions of the designer demonstrates the quality of interaction that is possible when the windowing system is integrated tightly with a particular hardware and software environment.

19.3.2 NeXTStep

The NeXTStep GUI provides a windowing system and graphical desktop environment originally intended for the NeXT Computer, which began shipping in 1988, but which ceased production in 1993. Nevertheless, the NeXTStep GUI has survived and is being made available on several types of workstations. The four component modules of the NeXTStep GUI are its Window Server, Workspace Manager, Application Kit, and Interface Builder.

NeXTStep was the first in a series of products to adopt a simulated three-dimensional appearance for its standard components. The Window Server uses Display PostScript to create high-quality grayscale screen displays providing graphics that can be output on any PostScript compatible printer. The Application Kit provides a standard set of object-oriented components that can be customized by application developers. The Interface Builder is an end-user oriented tool that allows operators to link these objects to system and application level functions with no additional programming. With this tool, standard user interface components can be used to automate the designer's tasks.

Like the Macintosh user interface, NeXTStep is oriented toward the needs of the non-technical designer. A straightforward mental model (i.e., an organization of data and functions), a simple set of controls, and a well-developed collection of software tools shield the designer from the complexity of the operating system and increase the suitability of the system, especially for the initially targeted market (students and scholars in higher education). Although the sophisticated UNIX-based operating system makes some degree of complexity inevitable, the design of the NeXTStep user interface makes the system accessible even for completely UNIX-naive users.

The NeXTStep GUI is notable for, among other things, using detachable, pop-up sub-menus under the primary menu that originally appear hanging from the top of the screen and "march" to the right in successively deeper layers. For ease of reference, each sub-menu repeats at the top of the sub-menu command list, the term that called it.

19.3.3 Open LOOK

The Open LOOK GUI was developed jointly by Sun Microsystems and AT&T as the standard operating environment for UNIX System V.4. Open LOOK exists as a layer on top of a base windowing system that provides the imaging model (management of how graphical parts are displayed) and network communication services. Versions of Open LOOK have been implemented on top of both the X Window System, the base-level set of windowing functions developed by a consortium of computer companies and MIT, and Sun's Network-extensible Window System (NeWS). In 1994, further Open LOOK development was discontinued, and many applications providers are converting to other GUIs; however, Open LOOK applications continue to exist.



Figure 1. Typical applications operating in the Apple Macintosh® GUI. Single or multiple windows can be moved and resized. Toolboxes and dialogue boxes can also be placed where needed. Pointer signs automatically change as required by the state of the system.

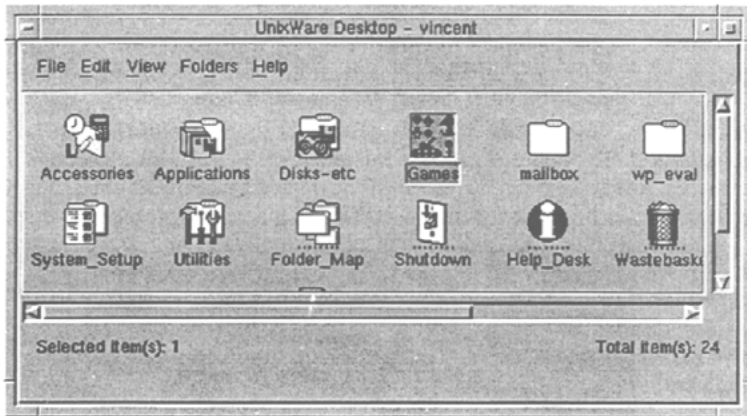


Figure 2. A typical application using a Motif-based GUI. On the screen is a typical window and typical icons. By selecting an icon, the user can launch a background task. If the user interface is consistent across applications as well as across hardware platforms, switching among platforms is easy, and training is simplified. (Figure courtesy of Addison-Wesley)

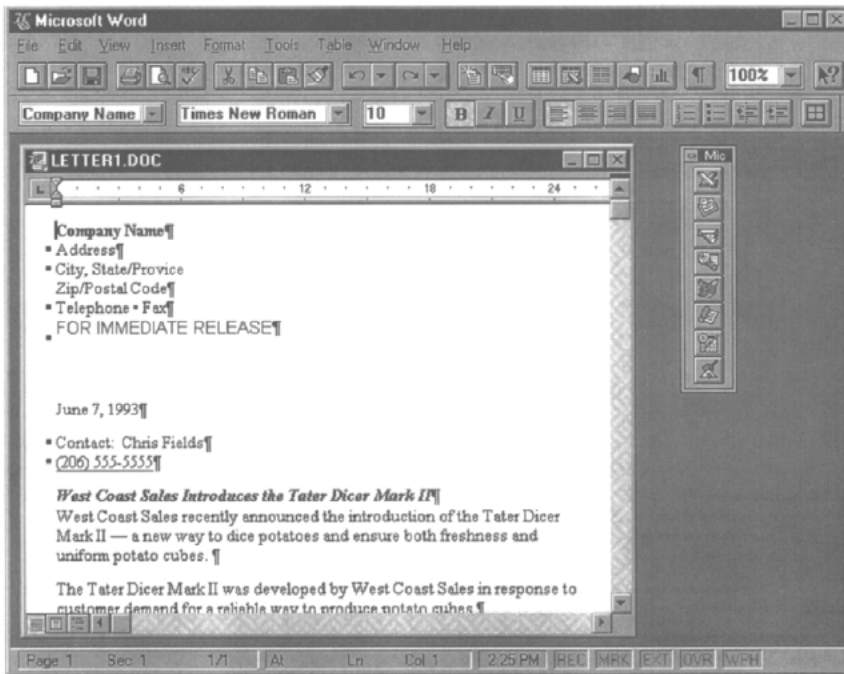


Figure 3. An application compliant with Microsoft Windows 95™. Typical features include task bar, dialogue-box controls with visual feedback, and object linking and embedding.

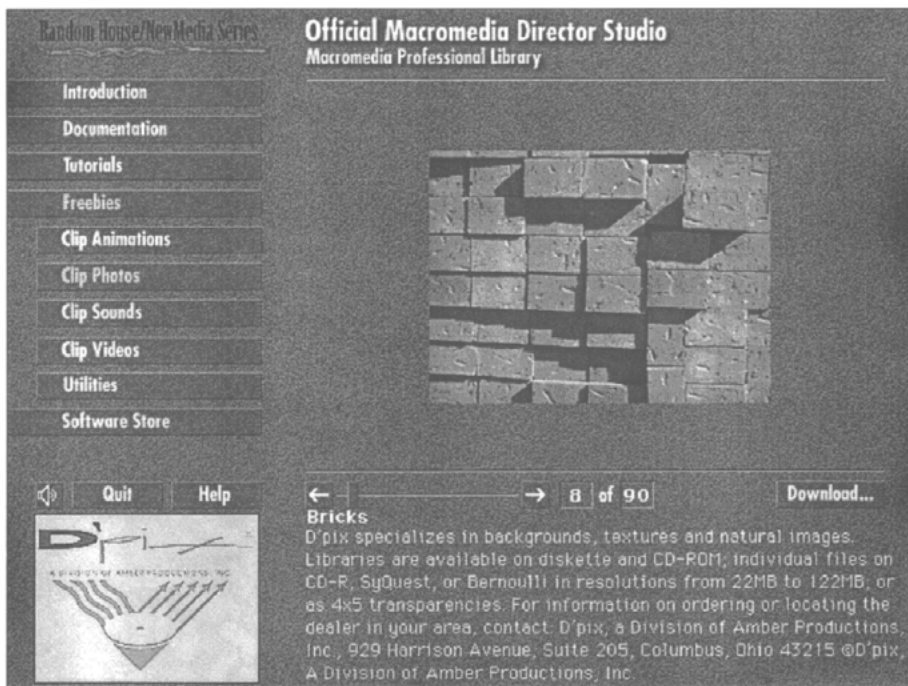


Figure 4. Example of an application in NeXTSTEP. Note the location of icons and the use of "marching" menus. (Figure courtesy of Addison-Wesley)

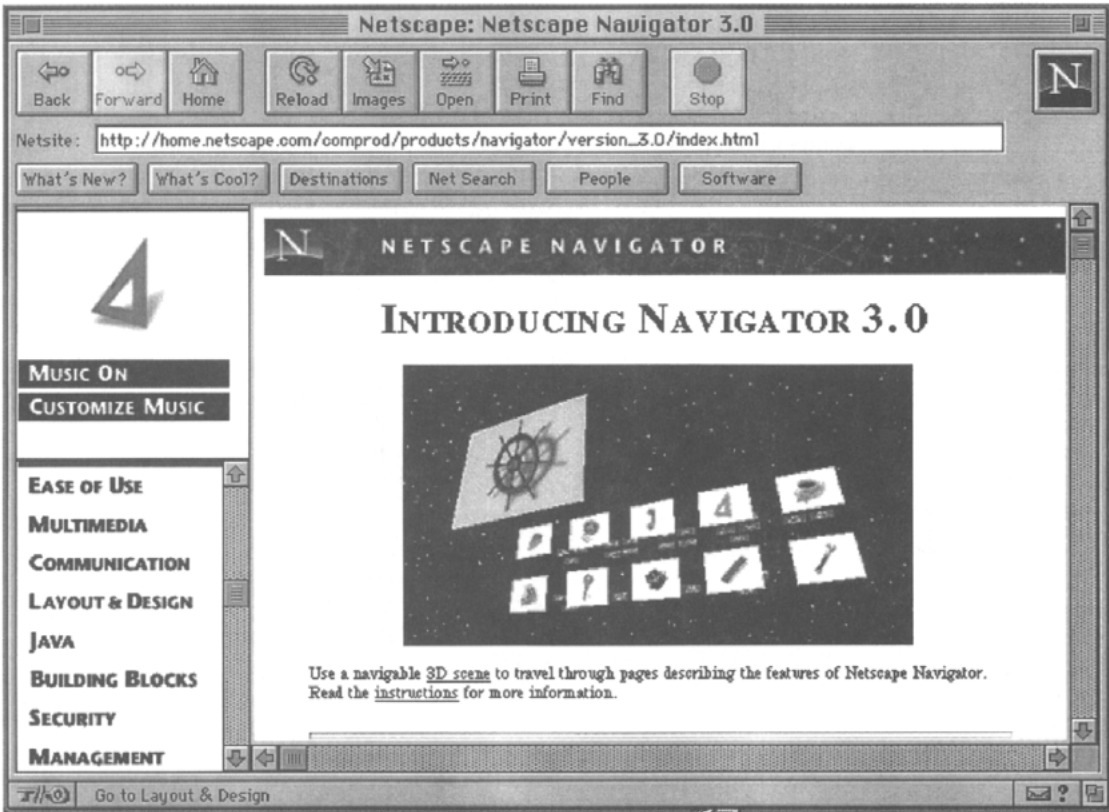


Figure 5. Example of Netscape Navigator frames.

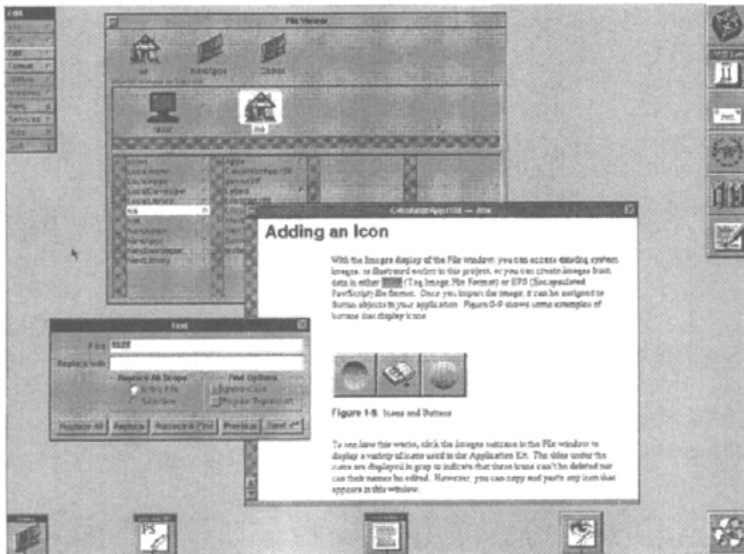


Figure 6. A custom GUI for Random House New Media CD-ROM-based multimedia application. Menus are displayed and removed automatically as required. Menus remain on the screen for repeated selection and assist in orienting the user. Secondary controls provide appropriate manipulation of multimedia content files.

Guidance for Open LOOK developers was provided by an exemplary functional specification and style guide. An explicit goal of the Open LOOK designers was to avoid potential legal challenges by creating innovative appearance and behavior characteristics. As a result, many of the conventions adopted deviate from the industry norm. Open LOOK adopted a contrasting appearance and approach to usability. Open LOOK was one of the earliest GUI conventions to propose muted color schemes for more effective display of complex screens. The layout of dialogue boxes and other content-full areas is often more “open” or “empty” looking than in other GUIs because of the design of the individual components.

Open LOOK’s orientation toward maximum functionality is evident in the numerous context-sensitive and mode-specific operations it provides. While it, too, makes the UNIX world relatively accessible even for inexperienced users, the extended functionality of Open LOOK itself introduces an additional layer of complexity that is not seen in NeXTStep or the Macintosh human interface.

19.3.4 OSF/Motif

OSF/Motif is a window manager and GUI tool kit developed by Digital Equipment Corporation and the Hewlett-Packard Company for the Open Software Foundation (OSF). Motif provides a three-dimensional, visually dense, and often more sophisticated-looking alternative to Open LOOK that is linked to the OSF version of standard UNIX. Like Open LOOK, the Motif Window Manager exists as a software layer on top of the network-oriented X Window System. Appearance can be modified independently of the functional characteristics of the resulting system, and individual vendors are encouraged to customize the functional shell with their own proprietary widget sets.

OSF/Motif provides a GUI for a high-end, network-based computing environment whose appearance and behavior is consistent with that of Microsoft Windows and OS/2 Presentation Manager. Because of their *de facto* standardization on IBM and compatible platforms, these operating environments dominate the movement toward GUIs for PC-based systems. OSF believes that knowledge of Windows and Presentation Manager will transfer easily to Motif, making it the windowing system of choice when PC users upgrade to workstation platforms.

The implementation of Motif on top of the network-transparent X Window System allows it to leverage an emerging standard in the workstation environ-

ment as well. Motif provides a windowing system that can serve as a bridge between the PC and workstation environments. Its potential for easing this transition will increase the attractiveness of Motif for organizations integrating high-performance work stations with existing PC networks.

19.3.5 Microsoft Windows and OS/2 Presentation Manager

Microsoft Windows was created in 1985 as a multi-tasking, graphics-oriented alternative to the character-based environment provided by MS-DOS on PC compatible systems. The bit-mapped displays and mouse-driven menus provided by Windows first opened the door to graphics-oriented software on the PC.

Initially, Windows was limited by many of the design characteristics (640K address space, low-quality display, etc.) of the DOS environment on which it was built. Later enhancements increased the responsiveness and graphical quality of Windows, particularly on 80386- and 80486-based machines. Microsoft’s Windows 95 and Windows NT environment are positioned to take advantage of Window’s general approach, but with added usability features for Windows 95 and networking and multitasking capabilities for Windows NT. Windows 95 provides versions of many GUI features that Macintosh established earlier, e.g., use of icons, extensive drag-and-drop functionality, long file names, etc. Windows NT provides a GUI for high performance networked PCs and workstations.

The OS/2 Presentation Manager was developed jointly by Microsoft and IBM in 1987 and is favored by IBM and some compatible microcomputer manufacturers. The appearance and behavior of Presentation Manager are derived primarily from Windows.

Microsoft Windows and the OS/2 Presentation Manager must satisfy a very different market consisting largely of existing MS-DOS users in business and technical environments. The extensive support for keyboard-based control provided by these products reflects the heritage of the character-based DOS interface, which has historically relied heavily on keystroke combinations for selecting from menus and items in dialogue boxes.

19.3.6 Custom GUIs

Although most industry productivity tool applications are moving toward one or more standard commercial GUIs using the above mentioned windowing systems, some previous and current products utilize custom ap-

proaches. For example, touch screen applications on kiosks or multimedia-oriented, computer-based training applications may use non-standard layouts, controls, and interaction. In addition, standard GUIs may continue to include hold-over function keys or buttons from the earlier text-oriented user interfaces. Few GUI building tools prevent developers from incorporating custom deviations from standard practice. Consequently, many non-standard, semi-customized versions of commercial GUIs may be encountered, especially in multimedia and Web-oriented products.

19.4 Windowing System Architectures

Traditional windowing systems divide the display screen into multiple functional areas that provide a means of monitoring and controlling multiple application programs or manipulating multiple data objects in a graphical environment. The windows in which documents and applications are presented provide a set of well-defined visual and functional contexts that allow multiple processes to time-share a single set of input devices (mouse, keyboard, etc.) and a limited amount of physical display space. The windowing code (software architecture), the method of window management, and the base window system's method of displaying an image (image model) can have noticeable effects on the quality of the displays and the level of interaction experienced by the user. There are two types of classical windowing system architectures: kernel and client-server based in design. Each is discussed further below.

19.4.1 Kernel-Based Architecture

The location and organization of the software that implements the windowing system can influence the responsiveness, device dependence, and resource requirements of the resulting system. Kernel-based architectures provide high levels of interactivity but are dependent on the architecture and available resources of a single machine.

In kernel-based systems, windowing services are provided by some portion of the operating system itself, or by a standard add-on module that resides along with the operating system in RAM (Random Access Memory) or ROM (Read Only Memory) based libraries. Kernel-based windowing codes and operating systems share the same physical memory space and are accessed in essentially the same way.

19.4.2 Client-Server Based Architecture

Client-server based architectures allow a single inst-

ance of the windowing system software to be shared across entire networks of heterogeneous machines, but response times may be limited by the communication bandwidth of the network. A server is a computer running software that provides a particular capability to an entire network of interconnected machines. A client is a piece of software on the same network that requests and uses the capabilities provided by the server. Even the best client-server implementations incur significant communication overhead that can lead to noticeable performance degradation compared to kernel-based windowing systems. Kernel-based systems achieve higher performance at the cost of device dependence and the need to execute redundantly the same code on each machine.

19.5 Window Management

Window management facilities allow the system to maintain spatial relationships among windows as they are moved, sized, and depth-arranged. Several options are available in window control menus that feature automatic arrangement of windows. Of particular importance are tiled, overlapping, and cascading windows. While the historical controversy over the relative merits of tiled and overlapping windows continues, industry practice has favored overlapping window management; however, tiled windows may still be useful in large or high-resolution displays. These window management styles are discussed in more detail below.

19.5.1 Tiled Windows

Tiled windows are arranged automatically by the windowing system to completely fill the available display space, which may be either the entire display screen or an entire content area of a window. Windows are prevented from overlapping. When any window is resized, other windows must be sized in the opposing direction to compensate. Tiled window arrangements are often useful when investigation of users, their tasks, and the application(s) being used indicated that a tiled arrangement of certain windows will meet most users' needs most of the time during initial, intermittent, and/or frequent use. An alternative to this scheme is a single window with multiple panes, exemplified by Microsoft Window's single-document interface (SDI) concept.

19.5.2 Overlapping Windows

Overlapping windows have associated depth values that represent their distances from the viewer. At each

displayed location of a visual point or pixel, only the contents of the nearest window covering that portion of the display is presented. The window with the nearest (to the viewer) depth value thus obscures the contents of any other windows occupying the same display space, creating an illusion of physical overlapping. The resulting window stack is comparable to a pile of papers on a desk and allows the designer to take advantage of existing spatial management skills to push one or more windows to the rear, or to bring one or more forward. This scheme is exemplified by Microsoft Window's multiple-document interface (MDI) concept, which can be confusing for some novice users.

19.5.3 Cascading Windows

Cascading windows are a special case of overlapping window management in which the windows are arranged automatically in a regular progression that prevents any window from being completely obscured. One corner (usually, the upper-left) of each successive window appearing "forward" of those "behind" is offset slightly in both the horizontal and vertical directions to conserve display space. Because each window's title bar at the top is visible, the user can easily see the progression of contents, while the selectability of each window simplifies the task of bringing any window to the front of the stack.

19.6 Windowing System Components

The appearance and behavior of the windowing system as experienced by the user is determined by a small group of standard windowing system components. GUIs make use of essentially the same set of these components, while the names by which they are identified vary significantly among vendors. The set of terms listed and discussed below will streamline cross-product comparisons by identifying standard components consistently and unambiguously. (A cross-comparison of all the widgets of the primary standard GUIs appears in Marcus, Smilonich, and Thompson's *The Cross-GUI Handbook for Multi-platform User Interface Design* cited in the Bibliography.)

1. Windows
2. Menus
3. Controls
4. Dialogue Boxes
5. Modeless Dialogues
6. Modal Dialogues

7. Control Panels
8. Query Boxes
9. Message Boxes
10. Mouse and Keyboard Interface

19.6.1 Windows

From the viewpoint of the window manager, a window is any discrete area of the visual display that can be moved, sized, and rendered independently on the display screen. Even though most of the components are actually implemented and managed as windows by the system, it is appropriate to consider windows from the user's point of view. The definition employed will therefore include only those display areas that allow one to change the view of the window's contents using techniques such as sizing, scrolling, or editing.

19.6.2 Menus

Menus provide a means of command execution that enables a designer to see and point instead of remembering and typing. The menu system greatly reduces problems caused by the limitations of human memory, but does so at the expense of motor performance. The benefits are substantial, particularly when the number and complexity of commonly used applications limits the user's expertise with individual command sets.

19.6.3 Controls

Any visually represented window component that can be manipulated directly with the mouse or keyboard is a control. Each of the windowing systems defines standard sets of controls that can be incorporated by applications to provide consistent interaction protocols across products.

19.6.4 Dialogue Boxes

Dialogue boxes provide a visual and functional context for presenting options from which the designer can select. Any interactive exchange of information between the designer and the system that takes place in a limited spatial context is considered a dialogue. Although three distinct classes of dialogue box are described here (control panels, query boxes, and message boxes), there may be considerable overlap among the classes. Any dialogue box can be characterized by a clearly defined scope that determines the effect on the state of the system and the subsequent operations permitted.

19.6.5 Modeless Dialogues

Modeless dialogue boxes are limited in scope and do not restrict the subsequent actions of the user. Modeless dialogues may incorporate some basic window functions such as sizing and positioning. Users can continue to work without responding, if necessary, and may be allowed to keep the modeless dialogue on display even after a response has been made.

19.6.6 Modal Dialogues

Modal dialogue boxes require the user to respond before any other action can be taken. Application modal dialogues prevent the user from invoking any application functions until the dialogue requirements have been satisfied, while system modal dialogues prevent the user from performing any operations anywhere in the system.

19.6.7 Control Panels

Control panels appear at the implicit or explicit request of the user and provide information reflecting the current state of a number of related system parameters, any of which can be changed interactively while the panel remains on display. Changes to the system state do not take effect until the user explicitly accepts the new settings.

19.6.8 Query Boxes

Query boxes appear in response to user actions, but are not requested explicitly. Query boxes prompt for a single piece of information, such as a yes-or-no answer to a single question, and provide a context in which the necessary information can be provided. Like control panels, query boxes allow the user to cancel the action that led to the query.

19.6.9 Message Boxes

Providing critical information to the user is the primary function of message boxes, which are not requested and typically appear only when the system has entered, or is about to enter, an unrecoverable and potentially dangerous state. The user's response options typically are limited to a simple yes-or-no decision, or in irreversible system states, to simple acknowledgment of the message.

19.6.10 Mouse and Keyboard Interface

GUI systems typically use a mouse and keyboard as the primary interaction devices. Each device is well-suited to certain kinds of interaction tasks. The mouse provides an efficient means of accomplishing tasks that require spatial manipulation, such as menu navigation and window sizing and positioning. The keyboard is more efficient for sequential tasks, such as text entry and changing the relative depth location of windows by bringing one of them to the top.

19.7 Design Guidelines

Design Advice

The following sections provide guidance in the use of GUI windowing system components. The recommendations are intended to aid in evaluating GUIs implemented with various software packages and can be generically applied to all GUIs. If an application allows user-customization of the GUI, this advice may assist in this process as well.

19.7.1 Design Characteristics

A GUI design must account for the following characteristics:

1. Metaphor: Comprehensible images, concepts, or terms
2. Mental Model: Appropriate organization of data, functions, tasks, and roles
3. Navigation: efficient movement among the data, functions, tasks, and roles via windows, menus, and dialogue boxes
4. Appearance: Quality presentation characteristics, or look
5. Interaction: Effective input and feedback sequencing, or feel

19.7.2 Design Guidelines

Three key principles guide GUI design and user-based customization:

1. Organization: Provide the designer with a clear and consistent conceptual structure
2. Economy: Maximize the effectiveness of a minimal set of cues
3. Communication: Match the presentation to the capabilities of the user

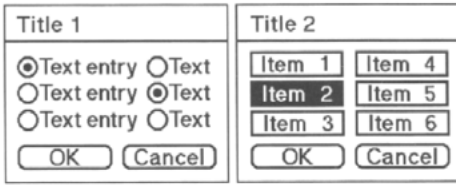


Figure 7. Chaotic and ordered screens. The examples illustrate the difference between a disorganized and organized screen layout.

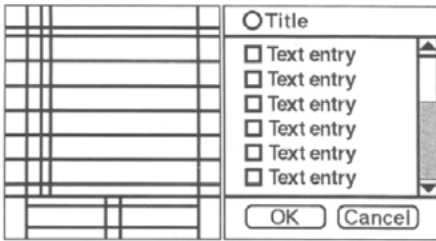


Figure 8. Internal consistency in dialogue boxes. The examples illustrate a consistent location and appearance for titles, main contents, and action buttons.

19.7.3 Order and Chaos

Organization lends order to a GUI, making it easier for the user to understand and navigate. Without visual and cognitive organization, the GUI becomes chaotic and therefore difficult to learn and use. Figure 7 shows an example of the trade-off between order and chaos. Organization can best be understood by examining key components such as consistency, screen layout, relationships, and navigability.

19.7.4 Consistency

The principle of internal consistency says: observe the same conventions and rules for all elements of the GUI. Figure 8 provides an example. Without a strong motivating reason, casual differences cause the viewer to work harder to understand the essential message of the display. The GUI should deviate from existing conventions only when doing so provides a clear benefit to the operator. In other words, the GUI should have a good reason for being inconsistent. GUI researcher Jonathan Grudin (1989) [3] has shown that sometimes it is impossible to avoid inconsistency, and that incon-

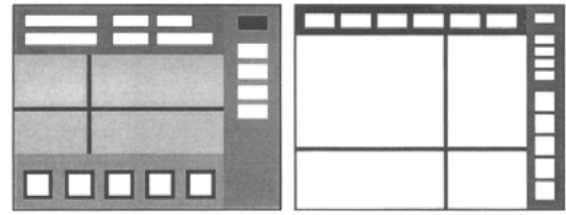


Figure 9. Grid and dialogue box. The example illustrates a layout grid of lines that can be used to locate all visual elements of the dialogue box.

sistency can even be beneficial under certain circumstances. However, as a general rule, strive for consistency without lacking in originality.

19.7.5 External Consistency: Leverage Known Design Techniques

The GUI should be designed to match the user’s expectations and task experience as much as possible rather than force users to understand new principles, tasks, and techniques. This design approach will make the user interface more intuitive and friendly.

19.7.6 GUI Screen Layout

There are three primary means of achieving an organized screen layout:

1. Use an underlying layout-grid
2. Standardize the screen layout
3. Group related elements.

Figure 9 shows examples of dialogue boxes based on a grid structure.

19.7.7 Visual Relationships

Another technique helpful in achieving visual organization is to establish clear relationships by linking related elements and disassociating unrelated elements through their size, shape, color, texture, etc. Examples of elements grouped by relationships appear in Figure 10.

19.7.8 Navigability

An organized GUI provides an initial focus for the viewer’s attention, directs attention to important, secondary, or peripheral items, and assists in navigation

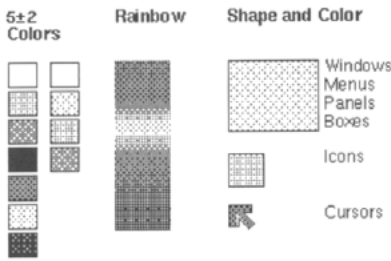


Figure 10. Relationships between grouped items. The examples illustrate confusing and clear use of color, locations, shape, and size to visually group components of screen displays.

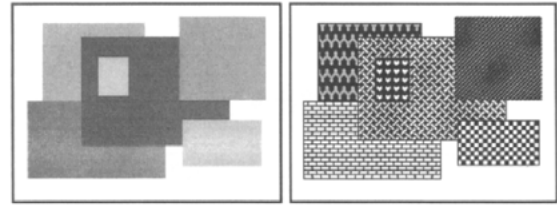


Figure 13. Distinctiveness. Too little (at left) gives the screen a bland, uninformative look. Too much is chaotic and yields no information about how the items relate to each other.



Figure 11. Navigability. A viewer looking at the screen on the left would not know where to begin. The redesigned screen on the right provides clear entry points into the screen via a visual reinforcement of the hierarchy of elements.

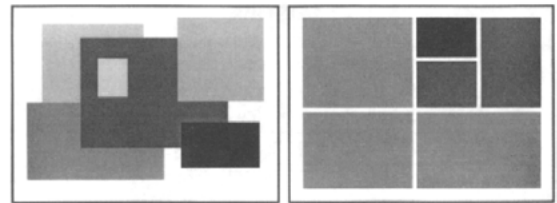


Figure 14. Emphasis. Because every element in the figure is emphasized, the overall effectiveness is reduced. The viewer does not know what is most important. The figure at right corrects this situation, giving appropriate emphasis to each element.

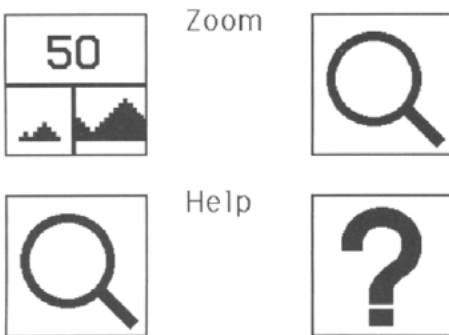


Figure 12. Clarity. Ambiguous icons confuse and frustrate viewers. Clearly designed icons help them to understand the application.

throughout the material. Figure 11 provides an example of a screen layout redesigned for improved navigability.

19.7.9 Economy

Economy concerns achieving effects through modest means. Simplicity suggests that including only those

elements that are essential for communication. For information intensive situations, the design should be as unobtrusive as possible. Some guidelines regarding economy include the following:

1. Modesty: In general, GUI components should be modest and inconspicuous. Users should be almost unaware of the GUI working to convey meaning.
2. Clarity: Components should be designed with unambiguous meanings. Figure 12 provides a contrast of ambiguous and clearly designed icons.
3. Distinctiveness: Distinguish the important properties of essential elements. Figure 13 illustrates this technique.
4. Emphasis: In general, make the most important elements salient, i.e., easily perceived. De-emphasize non-critical elements, and minimize clutter so that critical information is not hidden. Figure 14 illustrates this point.

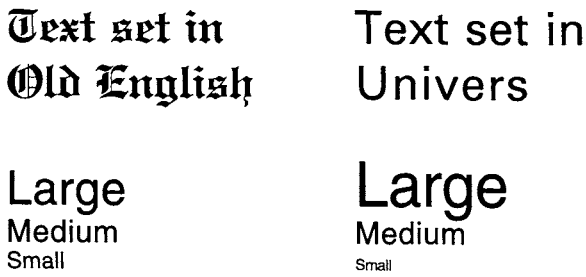


Figure 15. Legibility. The decorative typeface on the left is less legible than the clean sans-serif type on the right. Size variations in text (lower left) are not distinct enough, while the text on the right clearly establishes a type hierarchy.

Unreadable: Design components to be easy to interpret and understand. Design components to be inviting and attractive.

Readable

Design components to be easy to interpret and understand.

Design components to be inviting and attractive

Figure 16. Readability. Centered text at the top of the figure is not as easy to interpret or as enjoyable to read as the left justified, well-spaced text below.

19.7.10 Balanced Communication

To communicate successfully, a GUI designer must balance many factors. Well-designed GUIs achieve this balance through the use of information-oriented, systematic graphic design. This refers to the use of layout, typography, symbols, color, and other static and dynamic graphics to convey facts, concepts, and emotions.

19.7.11 Layout

The starting point for a well-designed GUI is its layout. Layout refers to the spatial organization of all dialogue boxes and windows according to an underlying grid of horizontal and vertical lines. In general, the visual field

should be governed by 7 ± 2 major lines in each orientation. These lines will regularize the appearance of all other elements, including typography, icons, charts, etc.

19.7.12 Legibility

Any GUI should be legible. Legibility refers to the design of individual characters, symbols, and graphic elements to be easily noticeable and distinguishable. Figure 15 shows some examples of legibility based on typeface (font) and size.

19.7.13 Backgrounds

Remember that dark screen backgrounds in brightly lit rooms may cause distracting reflections that can diminish screen legibility. At the other extreme, brightly lit screens in dark rooms may be too glaring and difficult to see.

19.7.14 Readability

Readability refers to a display that is comprehend, i.e., easy to identify and interpret, as well as inviting and attractive. Figure 16 presents an example of contrast in readability of texts.

19.7.15 Typography

Individual GUI elements (typefaces, such as Times Roman or Helvetica) and type styles (such as bold roman or regular italic), and their arrangement (typesetting techniques, such as line spacing) should be optimized for effective communication. The following are some guidelines to consider:

1. Within menus, dialogue boxes, control panels, forms, and other window components, adjust the point size, word spacing, paragraph indentation, and line spacing to enhance readability and to emphasize critical information.
2. Limit type variations to a maximum of 1-3 typefaces in 1-3 sizes for most applications. Lines of text should have 40-60 characters maximum, and words should be spaced correctly (usually the width of a lower case "r" for variable-width text).
3. Set text in appropriate formats, i.e., flush-left, columns of numbers flush right, avoid centered text in lists, and avoid short, justified lines of text. For fixed-width fonts, justified lines of text can slow reading speed by 12%.
4. Use upper and lower case characters whenever possible, i.e., avoid all capital lines of text, which can also slow reading speed by 12%.

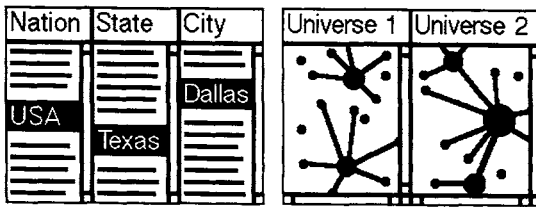


Figure 17. Multiple views (visual and verbal). The examples illustrate how graphic and textual GUI components can present the viewer with different views of a single object or multiple views of different objects in order to communicate complex information effectively.

19.7.16 Symbolism

GUI symbolism refers to signs, icons, and symbols that can help to communicate complex information and make the display more appealing. In general, keep in mind the following:

1. Use symbols or icons that are clear and unambiguous.
2. Use familiar references when possible.
3. Be consistent in the size, angles, weights, and visual density of all the signs.

19.7.17 Multiple Views

One important technique for improving communication within a GUI is to provide multiple views of the display of complex structures and processes. Figure 17 gives an example of how to present multiple views. Good GUI design makes use of these different perspectives:

1. Multiple forms of representation
2. Multiple levels of abstraction
3. Simultaneous alternative views
4. Links and cross references`

19.7.18 Advantages of Color

Color, including texture, is a powerful communication tool; so powerful, in fact, that color is easy to misuse or overuse. GUI designers must therefore understand color's functions so as to use color with proper skill and sophistication. Color refers to these dimensions: hue (combinations of wavelength), value (degree of lightness or darkness), and chroma (degree of purity or

vividness). In addition, brightness refers to the amount of radiant energy in the display of color.

Color plays a significant role in communicating with a designer. Some of the most important tasks color can accomplish are these:

1. Emphasize important information
2. Identify subsystems or structures
3. Portray natural objects realistically
4. Portray time and progress
5. Reduce errors of interpretation
6. Add coding dimensions
7. Increase clarity or comprehensibility
8. Increase believability and appeal

19.7.19 Color Similarity

In general, similar colors imply a relation among objects. A viewer can sense the relatedness by color over space and over time in sequences of images. Therefore, color should be used to group related items, and a consistent color code should be used for screen displays, documentation, etc. Also, use similar background colors for related areas. This color coding can subtly reinforce the conceptual link among these areas in the viewer's mind.

19.7.20 Color Consistency

Be complete and consistent in color groupings. For example, command and control colors in menus should not be used for information coding within a work area, unless a specific connection is intended. Once color coding is established, the same colors should be used throughout the GUI and all related publications. This color continuity may require designing colors to appear in different media: CRT screens use additive color mixtures, while most hardcopy devices use subtractive color mixtures. The color gamuts (that is, available color ranges) of these two media usually are not identical.

19.7.21 Color Economy: Redundancy

The principle of color economy suggests using a maximum of 5 ± 2 colors where the meaning must be remembered. Note that this maximum is even less than Miller's (1956) number [4] of 7 ± 2 , which refers to a human cognitive functioning limit with short term memory. If appropriate, use redundant coding based on shape as well as color.

19.7.22 Color Economy: Enhancement

Color should enhance black-and-white information, that is, in general, the display should work well when viewed as black, white, and grays. For documentation tasks and for the use of color to portray an object, the maximum number of colors needed is dependent on the application. For aesthetic purposes such as design style, emotional expression, or realism, many more colors may be required. A set of 5 ± 2 colors may include a few grays and some strongly different hues, or the set may use only different values for a given hue.

19.7.23 Color Economy: Sequencing

To code a large set of colors, use the spectral sequence: red, orange, yellow, green, blue, and violet. Francine Frome (1983) [5], a human factors researcher has shown that CAD/CAM viewers see a spectral order as a natural one and would select red, green, and blue as intuitive choices for the front, middle, and back layers, respectively, when viewing a multi-layer display.

Note, however, that brightness can change a viewer's perception of depth. If the colors are balanced, then red seems to come forward. Use redundant coding of shape as well as color. This technique aids those with color deficient vision and makes the display more resilient to color distortions caused by ambient light changes or by converting a color display from one medium to another, such as from a CRT to slides.

Ambient light can cause changes in all dimensions of color. Conversion from one medium to another can cause unforeseen and sometimes uncontrollable changes. Remember that among Caucasian viewers, approximately 8% of males have some form of color-deficient vision.

19.7.24 Color Emphasis

Color emphasis suggests using strong contrast in value and chroma to focus the operator's attention on critical information. The use of bright colors for danger signals, attention-getters, reminders, and pointers/cursors is entirely appropriate. High chroma red alerts seem to aid faster response than yellow or yellow-orange if brightness is equal, but this also depends upon the background colors. When too many figures or background fields compete for the viewer's attention, confusion arises, as can happen in the approach to color design that makes displays look appropriate for Las Vegas (the use of many high-chroma colors).

19.7.25 Color Emphasis: Hierarchy

The hierarchy of highlighted, neutral, and lowlighted states for all areas of the visual display must be carefully designed to maximize simplicity and clarity. Here again we find the basic maxim: simplicity, clarity, and consistency are especially important for color design.

19.7.26 Color Emphasis: Viewing Differences

Older viewers may be less able to distinguish blue from white and blue-green from bluish-white light due to natural aging and change of color in the lens of the eye [6].¹¹ Those who have viewed displays for very long periods of time may require more saturated or high-chroma colors because of changes in their visual system. Bear in mind that frequent, short-term viewing can benefit from low-chroma displays, and that very bright displays of letters, symbols or lines tend to bloom, that is, the light spreads out against the background.

19.7.27 Color Communication: Central and Peripheral Colors

Select colors appropriate to the central and peripheral areas of the visual field. The outer edges of the retina are not particularly sensitive to colors in general. Red or green should be used in the center of the visual field, not in the periphery. If they are used at the periphery, some signal to the viewer must be given to capture attention, e.g., size change or blinking. Use blue, black, white, and yellow near the periphery of the visual field, where the retina remains sensitive. These considerations are particularly important for virtual reality systems.

19.7.28 Color Communication: Combinations

Use color combinations whose color contrasts are influenced least by the relative area of each color. Use blue for large areas, not for text type, thin lines, or small shapes. Blue-sensitive color receptors are the least numerous in the retina (approximately 5%), and are especially infrequent in the eye's central focusing area, the fovea. Blue is good for screen backgrounds.

19.7.29 Color Communication: Area

If the same colors appear in objects that vary greatly in size, bear in mind that, as color areas decrease in size, they appear to change their value and chroma.

19.7.30 Color Communication: High Chroma and Spectrally Extreme Colors

Choose colors that are not simultaneously high in chroma and located at the extreme ends of the visual spectrum, e.g., red and blue.

19.7.31 Color Communication: Chroma and Value

Use colors that differ both in chroma and value (lightness). Do not use adjacent colors that differ only in the amount of pure blue, because the edge between the two colors will appear to be fuzzy. To avoid this effect, it is helpful to use other combinations, such as a dark blue and a light blue.

19.7.32 Color Communication: Combinations to Avoid

Colors of simultaneously high-chroma, spectrally extreme, strong contrasts of red/green, blue/yellow, green/blue, and red/blue can create vibrations or illusions of shadows and after-images. Unless special visual effects are needed, avoid these combinations.

19.7.33 Color Communication: For Dark-Viewing

In general, use light text, thin lines, and small shapes (white, yellow, or red) on medium to dark backgrounds (blue, green, or dark gray) for dark viewing situations. Typical low ambient-light viewing situations are those for slide presentations, workstations, and video. Video displays produce colors that are lower in chroma.

19.7.34 Color Communication: For Light-Viewing

Use dark (blue or black) text, thin lines, and small shapes on light (light yellow, magenta, blue, or white) backgrounds for light viewing situations. Typical viewing situations are those for overhead transparencies and paper. Reserve for text type the highest contrast between a figure and its background field.

19.7.35 Color Communication: Interactions

The interaction of color is a very complex subject that cannot be meaningfully covered in this limited space. The basic text on the subject is Albers' (1975) book,

The Interaction of Color [7]. GUI designers need to become familiar with this topic, which students of art and design often study.

19.7.36 Color Symbolism: Using Color Codes

Remember the importance of symbolism in communication: use color codes that respect existing cultural and professional usage.

19.7.37 Color Symbolism: Connotations

Evoke color connotations with great care. Connotations vary strongly among different kinds of viewers, especially from different cultures. The correct use of color requires careful analysis of the experience and expectations of the viewers. For example, mailboxes are blue in the USA, bright red in England, and bright yellow in Greece. If color is used in an electronic mail icon on the screen, this suggests that color sets might be changed for different countries to allow for differences in international markets.

19.7.38 Keyboard Short-Cuts

GUIs emphasize visually oriented interaction and direct manipulation, which is often very desirable for novice or occasional users. Experts often prefer many keyboard-oriented shortcuts. In general, these should be provided for efficient operation by frequent and expert users. Their equivalencies should be made apparent in menus and on-line documentation.

19.8 Rules of Thumb

Besides the above specific guidelines for GUI's, most professionals follow general rules of thumb. (Lund, 1995) recently requested evaluation of a preliminary set via an Internet-based pole. The results (slightly edited) are summarized below. The average ratings for each of the rules of thumb, ordered from greatest relative impact on usability to least, is as follows:

- 4.1 Know the user, and you are not the user.
- 4.0 Things that look the same should act the same.
- 4.0 Everyone makes mistakes, so every mistake should be fixable.
- 3.9 The information for the decision needs to be there when the decision is needed.

- 3.8 Error messages should actually mean something to the user, and tell the user how to fix the problem.
- 3.8 Every action should have a reaction.
- 3.7 Don't overload the user's buffers.
- 3.6 Consistency, consistency, consistency.
- 3.5 Minimize the need for a mighty memory.
- 3.5 Keep it simple.
- 3.4 The more you do something, the easier it should be to do.
- 3.4 The user should always know what is happening.
- 3.4 The user should control the system. The system shouldn't control the user. The user is the boss, and the system should show it.
- 3.3 The idea is to empower the user, not speed up the system.
- 3.3 Eliminate unnecessary decisions, and illuminate the rest.
- 3.3 If the user made an error, let the user know about it before getting into real trouble.
- 3.3 The best journey is the one with the fewest steps. Shorten the distance between users and their goals.
- 3.2 The user should be able to do what the user wants to do.
- 3.2 Things that look different should act different.
- 3.2 The user should always know how to find out what to do next.
- 2.9 Do not let users accidentally cause themselves difficulty.
- 2.9 Even experts are novices at some point. Provide help.
- 2.9 Design for regular people and the real world.
- 2.9 Keep it neat. Keep it organized.
- 2.9 Provide a way to bail out and start over.
- 2.7 The fault is not in the user, but in the system.
- 2.5 If it is not needed, it's not needed.
- 2.5 Color is information.
- 2.3 Everything in its place, and a place for everything.

- 2.3 The user should be in a good mood when done.
- 2.0 If the user makes an error, at least let the user finish a thought before needing to fix the error.
- 1.7 Cute is not a good adjective for systems.
- 1.7 Let people shape the system to themselves, and paint it with their own personality.
- 1.3 To know the system is to love it.

Lund cautions: "...this survey does not guarantee that the top rated rules of thumb actually have the largest impact on usability, just that experienced professionals believe the rules do. Assessing their impact is an empirical exercise that would be worthwhile. Further, we don't know whether professionals apply the rules of thumb in the same way, or whether part of their virtue is that they represent a core principle that each expert applies in a unique way to achieve usability."

19.9 Conclusions

This chapter has introduced the major standard GUI/window manager paradigms and has provided guidance for good GUI design. GUIs present many simultaneous, complex challenges to achieving successful visual communication. This chapter has presented a basic set of recommendations that can help the designer get started in using layout, typography, symbolism and color more effectively. After these guidelines have been incorporated, consider establishing product- or company- wide style guides, templates, and color palettes so that others may adapt and benefit from previous work. Developing better communication is an important part of making applications that communicate effectively through high-quality graphic design of user interfaces.

19.10 Acknowledgments

This chapter is an edited version of an essay by Aaron Marcus, "Graphical User Interfaces," by Aaron Marcus in *Solids Modeling Handbook*, Chapter 18, Donald E. LaCourse, Editor, McGraw-Hill Publishers, New York, 1995, pp. 18.3-18.18, and is used with permission of the previous publisher. The article is based on chapters from Aaron Marcus, *Graphic Design for Electronic Documents and User Interfaces*, Addison-Wesley, Reading, 1992.

19.11 Bibliography (Including Cited References)

- Albers, J. (1975). *The Interaction of Color*, Yale University Press, New Haven.
- Baecker, R., and Aaron M. (1990). *Human Factors and Typography for More Readable Programs*, Addison-Wesley, Reading, Mass.
- Bertin, J. (1983). *The Semiology of Graphics*, University of Wisconsin Press, Madison.
- Del Galdo, L., and Jonathan G. (1996). *Designing International User Interfaces*, Wiley.
- Frome, F. (1983). Incorporating the Human Factor in Color CAD Systems, *Proceedings*, 20th Design Automation Conference, 1983, pp. 189-195.
- Grudin, J. (1989). The Case Against User Interface Consistency, *Human Factors*, Vol. 32, No. 10, October (1989), pp. 1164-1173.
- Hofmann, A. (1965). *Graphic Design Manual*, Reinhold Publishing Corp., New York.
- Lund, A. (1995). <LUND.HORIZON@X400GW.AMERITECH.COM>, broadcast E-mail message, 28 October, 1995. Subject: "Feedback on Ratings of Usability Rules of Thumb"
- Marcus, A. (1995). Graphical User Interfaces, in *The Solids Modeling Handbook*, Ed. Donald E. Lacourse, McGraw-Hill, New York, pp. 18.3-18.18.
- Marcus, A. (1992). *Graphic Design for Electronic Documents and User Interfaces*, ACM Press, Addison-Wesley, Reading, MA.
- Marcus, A., Nicholas S., and Lynne T. (1995). *The Cross GUI-Handbook for Multi-platform User Interface Design*, Addison-Wesley, Reading, MA.
- Marcus, A. (1995). Principles of Effective Visual Communication for Graphical User Interface Design. In Baecker *et al*, *Readings in Human-Computer Interaction*, Morgan-Kaufman, San Francisco, pp. 425-468.
- Marcus, A. (1982). Color: A Tool for Computer Graphics Communication. In Greenberg, Donald, et al., *The Computer Image*, Addison-Wesley, Reading, pp. 76-90.
- Miller, G.A. (1956). The Magical Number Seven Plus or Minus Two: Some Limits on our Capacity for Processing Information. *Human Factors*, Vol. 63, pp. 81-97.
- Mueller-Brockman, J. (1981). *The Grid*, Verlag Arthur Niggli, Niederteufen, Switzerland.
- Ota, Y. (1987). *Pictogram Design*, Kashawi Shobo Publishers, Tokyo, Japan.
- Thorrell, L. G. and W. J. Smith, *Using Computer Color Effectively*, Prentice Hall, Englewood Cliffs, 9.
- Tufte, E. R. (1983). *The Visual Display of Quantitative Information*. Graphics Press, Cheshire, Conn.