USER INTERFACE DESIGN

18CS734

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CANARA ENGINEERING COLLEGE

MODULE 1

1.1 DEFINING THE USER INTERFACE

- User interface design is a subset of *human-computer interaction* (HCI).
- Human-computer interaction is the study, planning, and design of how people and computers work together so that a person's needs are satisfied in the most effective way.
- HCI designers must consider a variety of factors: what people want and expect, what physical limitations and abilities people possess, how their perceptual and information processing systems work, and what people find enjoyable and attractive. Technical characteristics and limitations of the computer hardware and software must also be considered.
- The *user interface* is the part of a computer and its software that people can see, hear, touch, talk to, or otherwise understand or direct.
- Two components of user interface: input and output.
  - *Input* is how a person communicates his or her needs or desires to the computer. Some common input components are the keyboard, mouse, trackball, one's finger (for touch-sensitive screens), and one's voice (for spoken instructions).
  - *Output* is how the computer conveys the results of its computations and requirements to the user.
- The most common computer output mechanism is the display screen, followed by mechanisms that take advantage of a person's auditory capabilities: voice and sound.
- Proper interface design will provide a mix of well-designed input and output mechanisms that satisfy the user's needs, capabilities, and limitations in the most effective way possible.
- The best interface is one that it not noticed, one that permits the user to focus on the information and task at hand, not the mechanisms used to present the information and perform the task.

1.2 THE IMPORTANCE OF GOOD DESIGN

- A well-designed interface and screen is very important to users. It is their window to view the capabilities of the system.
- It is one of the few visible components of the product developers create. It is also the vehicle through which many critical tasks are presented. These tasks often have a direct impact on an organization's relations with its customers, and its profitability.
- A screen's layout and appearance affect a person in a variety of ways. If they are confusing and inefficient, people will have greater difficulty in doing their jobs and will make more mistakes Poor design may even chase some people away from a system permanently. It can also lead to aggravation, frustration, and increased stress.

The Benefits of Good Design

- Imagine the productivity benefits we could gain through proper design.
- Based on an actual system requiring processing of 4.8 million screens per year and illustrated in Table 1.1, an analysis established that if poor clarity forced screen users to spend one extra second per screen, almost one additional person-year would be required to process all screens. Twenty extra seconds in screen usage time adds an additional 14 person-years.

Table 1.1 Impact of Inefficient Screen Design on Processing Time

ADDITIONAL SECONDS REQUIRED PER SCREEN IN SECONDS	ADDITIONAL PERSON-YEARS REQUIRED TO PROCESS 4.8 MILLION SCREENS PER YEAR
1	.7
5	3.6
10	7.1
20	14.2

- The benefits of a well-designed screen have also been under experimental scrutiny for many years.

  Example: to improve screen clarity and readability by making screens less crowded, if separate items that had been combined on the same display line to conserve space, were placed on separate lines instead, the result would be that screen users were about 20 percent more productive with the less-crowded version.

  Screen users of the modified screens completed transactions in 25 percent less time and with 25 percent fewer errors than those who used the original screens. Reformatting inquiry screens following good design principles reduced decision-making time by about 40 percent, resulting in a savings of 79 person-years in the affected system.

- proper formatting of information on screens does have a significant positive effect on performance.
- *one* graphical window redesigned to be more effective would save a company about $20,000 during its first year of use.
- Well-designed Web pages enable users to find information they need in lesser time.
- Other benefits from good design:
  - Training costs are lowered because training time is reduced
  - support line costs are lowered because fewer assist calls are necessary
  - employee satisfaction is increased because aggravation and frustration are reduced.

- an organization's customers benefit because of the improved service they receive.

- Identifying and resolving problems during the design and development process also has significant economic benefits. For every dollar spent fixing a problem during product design, $10 would be spent if the problem was fixed during development, and $100 would be spent fixing it after the product's release.

1.3 THE GRAPHICAL USER INTERFACE

- A *user interface* is a collection of techniques and mechanisms to interact with something.
- In a *graphical* interface, the primary interaction mechanism is a pointing device of some kind.
- This device is the electronic equivalent to the human hand. The user interacts with a collection of elements referred to as *objects*. They can be seen, heard, touched, or otherwise perceived.
- Objects are always visible to the user and are used to perform tasks. They are interacted with as entities independent of all other objects. People perform operations, called *actions*, on objects. The operations include accessing and modifying objects by pointing, selecting, and manipulating. All objects have standard resulting behaviors.

The Popularity of Graphics

- The older text-based screen possessed a one-dimensional, text-oriented, form-like quality, whereas graphic screens assumed a three-dimensional look.
- Information floated in windows, small rectangular boxes seemed to rise above the background plane.
- Windows could also float above other windows. Controls appeared to rise above the screen and move when activated.
- Lines appeared to be etched into the screen. Information could appear, and disappear, as needed, and in some cases text could be replaced by graphical images called icons. These icons could represent objects or actions.
- Screen navigation and commands are executed through menu bars and pull-downs. Menus "pop up" on the screen. In the screen body, selection fields such as radio buttons, check boxes, list boxes, and palettes coexisted with the reliable old text entry field.
- More sophisticated text entry fields with attached or drop-down menus of alternatives also became available. Screen objects and actions were selected through use of pointing mechanisms, such as the mouse or joystick, instead of the traditional keyboard.
- Increased computer power and the vast improvement in the display enable the user's actions to be reacted to quickly, dynamically, and meaningfully. This new interface is sometimes referred to as the WIMP interface: windows, icons, menus, and pointers.

- Graphic presentation of information utilizes a person's information-processing capabilities effectively than other presentation methods. Properly used, it reduces the requirement for perceptual and mental information recoding and reorganization, and also reduces the memory loads. It permits faster information transfer between computers and people by permitting more visual comparisons of amounts, trends, or relationships; more compact representation of information; and simplification of the perception of structure.
- Graphics also can add appeal or charm to the interface and permit greater customization to create a unique corporate or organization style.

The Concept of Direct Manipulation

- this style of interaction for graphical systems was first used by Shneiderman (1982).
- He called them "direct manipulation" systems, suggesting that they possess the following characteristics:
  - **The system is portrayed as an extension of the real world.** It is assumed that a person is already familiar with the objects and actions in his or her environment of interest. The system replicates them and portrays them on a different medium, the screen. A person has the power to access and modify these objects, among which are windows. A person is allowed to work in a familiar environment and in a familiar way, focusing on the data, not the application and tools. The physical organization of the system, which most often is unfamiliar, is hidden from view and is not a distraction.
  - **Continuous visibility of objects and actions.** Like one's desktop, objects are continuously visible. Reminders of actions to be performed are also obvious, labeled buttons replacing complex syntax and command names. Cursor action and motion occurs in physically obvious and intuitively natural ways. Nelson (1980) described this concept as "virtual reality," a representation of reality that can be manipulated. Hatfield (1981) is credited with calling it "WYSIWYG" (what you see is what you get). Rutkowski (1982) described it as "transparency," where one's intellect is applied to the task, not the tool. Hutchins, Hollan, and Norman (1986) considered it direct involvement with the world of objects rather than communicating with an intermediary.

    Problem in direct manipulation- there is no direct analogy on the desk for all necessary windowing operations. A piece of paper on one's desk maintains a constant size, never shrinking or growing. Windows can do both. Solving this problem required embedding a control panel, in a window's border. This control panel is manipulated, not the window itself.

- **Actions are rapid and incremental with visible display of results.** Since tactile feedback is not yet possible (as would occur with one's hand when one touches something), the results of actions are immediately displayed visually on the screen in their new and current form. Auditory feedback may also be provided. The impact of a previous action is quickly seen, and the evolution of tasks is continuous and effortless.
- **Incremental actions are easily reversible.** Actions, if discovered to be incorrect or not desired, can be easily undone.

Earlier Direct Manipulation Systems

In the earliest full-screen text editors, screens of text resembling a piece of paper on one's desk could be created (extension of real world) and then reviewed in their entirety (continuous visibility). Editing or restructuring could be easily accomplished (through rapid incremental actions) and the results immediately seen. Actions could be reversed when necessary.

Indirect Manipulation

In practice, direct manipulation of *all* screen objects and actions may not be feasible because of the following:

- The operation may be difficult to conceptualize in the graphical system.
- The graphics capability of the system may be limited.
- The amount of space available for placing manipulation controls in the window border may be limited.
- It may be difficult for people to learn and remember all the necessary operations and actions.

When this occurs, *indirect manipulation* is provided. Indirect manipulation substitutes words and text, such as pull-down or pop-up menus, for symbols, and substitutes typing for pointing. Most window systems are a combination of both direct and indirect manipulation. A menu may be accessed by pointing at a menu icon and then selecting it (direct manipulation). The menu itself, however, is a textual list of operations (indirect manipulation). When an operation is selected from the list, by pointing or typing, the system executes it as a command.

Graphical Systems: Advantages and Disadvantages

The simplified interface the graphical systems presented was thought to reduce the memory requirements imposed on the user, make more effective use of one's information-processing capabilities, and dramatically reduce system learning requirements.

Advantages

- **Symbols recognized faster than text.** Symbols can be recognized faster and more accurately than text, and the graphical attributes of icons, such as shape and color, are very useful for quickly classifying objects, elements, or text by some common property.
  example: the icons developed for indicating the kind of message being presented to the user of the system. The text of an informational message is preceded by an "i" in a circle, a warning message by an exclamation point, and a critical message by another unique symbol. These icons allow speedy recognition of the type of message being presented.
- **Faster learning.** A graphical, pictorial representation aids learning, and symbols can also be easily learned.
- **Faster use and problem solving.** Visual or spatial representation of information has been found to be easier to retain and manipulate and leads to faster and more successful problem solving. Symbols have also been found to be effective in conveying simple instructions.
- **Easier remembering.** Simplicity makes it easier for casual users to retain operational concepts.
- **More natural.** Graphic representations of objects are thought to be more natural and closer to innate human capabilities. In human beings, actions and visual skills emerged before languages. It has also been suggested that symbolic displays are more natural and advantageous because the human mind has a powerful image memory.
- **Exploits visual/spatial cues.** Spatial relationships are usually found to be understood more quickly than verbal representations. Visually thinking is believed to be better than logical thinking.
- **Fosters more concrete thinking.** Displayed objects are directly in the high-level task domain, or directly usable in their presented form. There is no need mentally to decompose tasks into multiple commands with complex syntactic form. The need for abstract thinking is therefore minimized.
- **Provides context.** Displayed objects are visible, providing a picture of the current context.
- **Fewer errors.** More concrete thinking affords fewer opportunities for errors. Reversibility of actions reduces error rates because it is always possible to undo the last step. Error messages are less frequently needed.
- **Increased feeling of control.** The user initiates actions and feels in control. This increases user confidence and hastens system mastery.

- **Immediate feedback.** The results of actions furthering user goals can be seen immediately. Learning is quickened. If the response is not in the desired direction, the direction can be changed quickly.
- **Predictable system responses.** Predictable system responses also speed learning.
- **Easily reversible actions.** The user has more control. This ability to reverse unwanted actions also increases user confidence and hastens system mastery.
- **Less anxiety concerning use.** Hesitant or new users feel less anxiety when using the system because it is so easily comprehended, is easy to control, and has predictable responses and reversible actions.
- **More attractive.** Direct-manipulation systems are more entertaining, cleverer, and more appealing. This is especially important for the cautious or skeptical user.
- **May consume less space.** Icons may take up less space than the equivalent in words. More information can often be packed in a given area of the screen. This, however, is not always the case.
- **Replaces national languages.** Language-based systems are seldom universally applicable. Language translations frequently cause problems in a text-based system. Icons possess much more universality than text and are much more easily comprehended worldwide.
- **Easily augmented with text displays.** Where graphical design limitations exist, direct-manipulation systems can easily be augmented with text displays. The reverse is not true.
- **Low typing requirements.** Pointing and selection controls, such as the mouse or trackball, eliminate the need for typing skills.
- **Smooth transition from command language system.** Moving from a command language to a direct-manipulation system has been found to be easy. The reverse is not true.

Disadvantages

In some cases, graphical representation and interaction may be poorer than pure textual or alphanumeric displays. Trying to force all system components into a graphical format may be doing a disservice to the user. As graphical systems are becoming increasingly sophisticated and continue to expand, interfaces have become increasingly more complex, sometimes arcane, and even bizarre. Among the disadvantages put forth are these:

- **Greater design complexity.** The elements and techniques available to the graphical screen designer far out-number those that were at the disposal of the text-based screen designer. Controls and basic alternatives must be chosen from a pile of choices numbering in excess of 50. This design potential may not necessarily result in better design, unless the choices are thoughtfully selected and consistently and simply applied. Proper window types must also be

chosen and colors selected from a seemingly unending rainbow of alternatives. With graphics, the skill of the designer is increasingly challenged. Poor design can undermine acceptance.

- **Learning still necessary.** It is not immediately obvious the first time one encounters many graphical systems. The meanings of many words and icons may not be known. It is not often possible to guess their meanings, especially the more arbitrary symbols. Usage of a pointing device may also have to be learnt. A severe learning and remembering requirement is imposed on many users, and it takes a while to get up to speed. A text-based system could easily be structured to incorporate a set of clear instructions: (1) Do this, (2) now do this, and so on. System providers estimate that becoming accustomed to a graphical interface should require about eight hours of training. Other experts say the learning time is closer to 20 or 30 hours.
- **Lack of experimentally-derived design guidelines.** The graphical interface today lacks availability of experimentally-derived design guidelines. More developer interest existed in solving technical rather than usability issues, so few studies to aid in making design decisions were performed.
    1. Builders of platforms and packages will not publish their study results because they want to maintain a competitive advantage.
    2. The studies are often specific to a particular function or task. They may not be generally applicable.
    3. It takes time and effort to publish something. The developer in today's office seldom has the time.
    4. Finally, it is also difficult to develop studies evaluating design alternatives because of increased GUI complexity. Too many variables that must be controlled make meaningful cause and- effect relationships very difficult to uncover. Consequently, there is too little understanding of how most design aspects relate to productivity and satisfaction.
- **Inconsistencies in technique and terminology.** Many differences in technique, terminology, and look and feel exist among various graphical system providers, and even among successive versions of the same system. These inconsistencies occur because of copyright and legal implications, product differentiation considerations, and our expanding knowledge about the interface. The result is that learning, and relearning, for both designers and users is much more difficult than it should be.
- **Working domain is the present.** While direct-manipulation systems provide context, they also require the user to work in the "present." "What you see is all you get." Language makes it easier to find things.

- **Not always familiar.** Symbolic representations may not be as familiar as words or numbers. Research has found that numeric symbols elicit faster responses than graphic symbols in a visual search task. One developer had to modify a new system during testing by replacing iconic representations with a textual outline format. The users, lawyers, were unfamiliar with icons and demanded a more familiar format.
- **Human comprehension limitations.** Human limitations may also exist in terms of one's ability to deal with the increased complexity of the graphical interface. The variety of visual displays can still challenge all but the most sophisticated users. The number of different icons that can be introduced is also restricted because of limitations in human comprehension. The number of different symbols a person can differentiate and deal with is much more limited than text. Some researchers note that claims for the easy understanding of pictograms are exaggerated, and that recognizing icons requires much perceptual learning, abstracting ability, and intelligence. The motor skills required may also challenge all but the most sophisticated users. Correctly double-clicking a mouse, for example, is difficult for some people.
- **Window manipulation requirements**. Window handling and manipulation times are still excessive and repetitive. This wastes time and interrupts the decision making needed to perform tasks and jobs.
- **Production limitations.** The number of symbols that can be clearly produced using today's technology is still limited. A body of recognizable symbols must be produced that are equally legible and equally recognizable using differing technologies. This is extremely difficult today.
- **Few tested icons exist.** Icons, like typefaces, must appear in different sizes, weights, and styles. An entire font of clearly recognizable symbols must be developed. Changing an icon's size can differentially affect symbol line widths, open areas, and so forth, dramatically affecting its recognizability. Typeface design is literally the product of 300 years of experimentation and study. Icons must be researched, designed, tested, and then introduced into the marketplace. The consequences of poor or improper design will be confusion and lower productivity for users.
- **Inefficient for touch typists.** For an experienced touch typist, the keyboard is a very fast and powerful device. Moving a mouse or some other pointing mechanism may be slower.
- **Inefficient for expert users**. Inefficiencies develop when there are more objects and actions than can fit on the screen. Concatenation for a command language is impossible.
- **Not always the preferred style of interaction.** Not all users prefer a pure iconic interface. A study comparing commands illustrated by icons, icons with text, or text-only, found that users preferred alternatives with textual captions.

- **Not always fastest style of interaction.** Graphic instructions on an automated bank teller machine were found to be inferior to textual instructions.
- **Increased chances of clutter and confusion.** A graphical system does not guarantee elimination of clutter on a screen. Instead, the chance for clutter is increased, thereby increasing the possibility of confusion. The amount of screen clutter one can deal with is open to speculation. The possibility that clutter may exist is evidenced by the fact that many people, when working with a window, expand it to fill the entire display screen. This may be done to reduce visual screen clutter. Mori and Hayashi (1993) found that visible windows, not the focus of attention, degraded performance in the window being worked on.
- **The futz and fiddle factor.** With the proliferation of computer games, computer usage can be wasteful of time. Experts have said that the most used program in Microsoft Windows is Solitaire! Tinkering includes activities such as creating garish documents reflecting almost every object property (font size, style, color, and so on.) available. Futzing and fiddling is a tool for learning how to use a mouse, for example, and it is a vehicle for exploring the system and becoming familiar with its capabilities. It is of value when done in moderation.
- **May consume more screen space.** Not all applications will consume less screen space. A listing of names and telephone numbers in a textual format will be more efficient to scan than a card file.
- **Hardware limitations.** Good design also requires hardware of adequate power, processing speed, screen resolution, and graphic capability. Insufficiencies in these areas can prevent a graphic system's full potential from being realized.

Characteristics of the Graphical User Interface

1. ***Sophisticated Visual Presentation:*** It is the visual aspect of the interface and what people see on the screen. The sophistication of a graphical system permits displaying lines, including drawings and icons. It also permits the displaying of a variety of character fonts, including different sizes and styles. The display of 16 million or more colors is possible on some screens. Graphics also permit animation and the presentation of photographs and motion video. The meaningful interface elements visually presented to the user in a graphical system include windows (primary, secondary, or dialog boxes), menus (menu bar, pulldown, pop-up, cascading), icons to represent objects such as programs or files, assorted screen-based controls (text boxes, list boxes, combination boxes, settings, scroll bars, and buttons), and a mouse pointer and cursor. The objective is to reflect visually on the screen the real world of the user as realistic, meaningful, simple, and as clear as possible.

2. ***Pick-and-Click Interaction:*** Elements of a graphical screen upon which some action is to be performed must first be identified. The motor activity required of a person to identify this element for a proposed action is commonly referred to as *pick*, the signal to perform an action as *click*. The primary mechanism for performing this pick-and-click is most often the mouse and its buttons. The user moves the mouse pointer to the relevant element (pick) and the action is signaled (click). Pointing allows rapid selection and feedback. The eye, hand, and mind seem to work smoothly and efficiently together. The secondary mechanism for performing these selection actions is the keyboard. Most systems permit pick-and-click to be performed using the keyboard as well.
3. ***Restricted Set of Interface Options:*** The array of alternatives available to the user is what is presented on the screen or what may be retrieved through what is presented on the screen, nothing less, nothing more. This concept fostered the acronym WYSIWYG.
4. ***Visualization:*** It is a cognitive process that allows people to understand information that is difficult to perceive, because it is either too voluminous or too abstract. It involves changing an entity's representation to reveal gradually the structure and/or function of the underlying system or process. Presenting specialized graphic portrayals facilitates visualization. The best visualization method for an activity depends on what people are trying to learn from the data. The goal is not necessarily to reproduce a realistic graphical image, but to produce one that conveys the most relevant information. Effective visualizations can facilitate mental insights, increase productivity, and foster faster and more accurate use of data.
5. ***Object Orientation:*** A graphical system consists of objects and actions. *Objects* are what people see on the screen. They are manipulated as a single unit. A well-designed system keeps users focused on objects, not on how to carry out actions. Objects can be composed of *subobjects*.

   For example, an object may be a document. The document's subobjects may be a paragraph, sentence, word, and letter. IBM's System Application Architecture Common User Access Advanced Interface Design Reference (SAA CUA) (IBM, 1991) breaks objects into three meaningful classes: data, container, and device.

   - ***Data objects*** present information. This information, either text or graphics, normally appears in the body of the screen. It is, essentially, the screen based controls for information collection or presentation organized on the screen.
   - ***Container objects*** are objects to hold other objects. They are used to group two or more related objects for easy access and retrieval. There are three kinds of container objects: the workplace, folders, and workareas. The *workplace* is the desktop, the storage area for all

objects. *Folders* are general-purpose containers for long-term storage of objects. *Workareas* are temporary storage folders used for storing multiple objects currently being worked on.

- ***Device objects*** represent physical objects in the real world, such as printers or trash baskets. These objects may contain others for acting upon. A file, for example, may be placed in a printer for printing of its contents.

Microsoft Windows specifies the characteristics of objects depending upon the relationships that exist between them. Objects can exist within the context of other objects, and one object may affect the way another object appears or behaves. These relationships are called collections, constraints, composites, and containers.

- A ***collection*** is the simplest relationship—the objects sharing a common aspect. A collection might be the result of a query or a multiple selection of objects. Operations can be applied to a collection of objects.
- A ***constraint*** is a stronger object relationship. Changing an object in a set affects some other object in the set. A document being organized into pages is an example of a constraint.
- A ***composite*** exists when the relationship between objects becomes so significant that the aggregation itself can be identified as an object. Examples include a range of cells organized into a spreadsheet, or a collection of words organized into a paragraph.
- A ***container*** is an object in which other objects exist. Examples include text in a document or documents in a folder. A container often influences the behavior of its content. It may add or suppress certain properties or operations of objects placed within it, control access to its content, or control access to kinds of objects it will accept. These relationships help define an object's *type*.

Another important object characteristic is *persistence*. Persistence is the maintenance of a state once it is established. An object's state (for example, window size, cursor location, scroll position, and so on) should always be automatically preserved when the user changes it.

Properties or Attributes of Objects

Objects also have properties or attributes. Properties are the unique characteristics of an object. Properties help to describe an object and can be changed by users. Examples of properties are text styles (such as normal or italics), font sizes (such as 10 or 12 points), or window background colors (such as black or blue).

Actions

People take actions on objects. They manipulate objects in specific ways (commands) or modify the properties of objects (property or attribute specification).

Commands are actions that manipulate objects. They may be performed by direct manipulation or through a command button. They are executed immediately when selected. Once executed, they cease to be relevant. Examples of commands are opening a document, printing a document, closing a window, and quitting an application.

Property/attribute specification actions establish or modify the attributes or properties of objects. When selected, they remain in effect until deselected. Examples include selecting cascaded windows to be displayed, a particular font style, or a particular color.

a typical *property/attribute specification sequence:*

1. The user selects an object—for example, several words of text.
2. The user then selects an action to apply to that object, such as the action BOLD.
3. The selected words are made bold and will remain bold until selected and changed again.

A series of actions may be performed on a selected object. Performing a series of actions on an object also permits and encourages system learning through exploration.

Application versus Object or Data Orientation

When a text-based system was developed, it was called an application. As graphical systems evolved, when a real picture of the user began to emerge, it became evident that people thought in terms of tasks, not applications. They choose objects and then act upon them.

- ✓ An application-oriented approach takes an *action: object* approach, like this:
  Action> 1. An application is opened (for example, word processing).
  Object> 2. A file or other object selected (for example, a memo).
- ✓ An object-oriented *object: action* approach does this:
  Object> 1. An object is chosen (a memo).
  Action> 2. An application is selected (word processing).

The object-action approach permits people to more easily focus on their task and minimizes the visibility of the operating system and separate applications. Many experienced users may have difficulty in switching from one approach to another since an old interaction behavior must be unlearned and a new one learned. New users should not experience these problems, since it more accurately reflects a person's thinking. In any one interface, it is critical that a consistent orientation be maintained, either an *object : action* or an *action : object* approach.

Views

Views are ways of looking at an object's information. IBM's SAA CUA describes four kinds of views: composed, contents, settings, and help.

- ***Composed*** views present information and the objects contained within an object. They are typically associated with data objects and are specific to tasks and products being worked with.
- ***Contents*** views list the components of objects.
- ***Settings*** views permit seeing and changing object properties.
- ***Help*** views provide all the help functions.

Use of Recognition Memory

Continuous visibility of objects and actions encourages use of a person's more powerful recognition memory. The "out of sight, out of mind" problem is eliminated.

Concurrent Performance of Functions

Graphic systems may do two or more things at one time. Multiple programs may run simultaneously. When a system is not busy on a primary task, it may process background tasks (cooperative multitasking). When applications are running as truly separate tasks, the system may divide the processing power into time slices and allocate portions to each application (preemptive multitasking). Data may also be transferred between programs. It may be temporarily stored on a "clipboard" for later transfer or be automatically swapped between programs.

1.4 THE WEB USER INTERFACE

- ❖ Web interface design is essentially the design of navigation and the presentation of information. It is about content, not data.
- ❖ Proper interface design properly balances the structure and relationships of menus, content, and other linked documents or graphics.
- ❖ The design goal is to build a hierarchy of menus and pages that feels natural, is well structured, is easy to use, and is truthful.
- ❖ The Web is a navigation environment where people move between pages of information, not an application environment.
- ❖ It is also a graphically rich environment.
- ❖ Web interface design difficult because-
    1. its underlying design language, HTML, intended for use by technical users and not by the general population. HTML was limited in objects and interaction styles and did not provide a means for presenting information in the most effective way for people.

2. browser navigation retreated to the pre-GUI era. This era was characterized by a "command" field whose contents had to be learned, and a navigational organization and structure that lay hidden beneath a mostly dark and blank screen. GUIs eliminated the absolute necessity for a command field, providing menus related to the task and the current contextual situation. Browser navigation is mostly confined to a "Back" and "Forward" concept, but "back-to-where" and "forward-to-where" is often unremembered or unknown. Use of the Back button at the wrong time can destroy previously done work. Remaining navigation was willed to Web pages themselves, where the situation only worsened. Numerous links were provided to destinations unknown, invisible navigation buttons lay unrecognizable on the screen, and linked jumps two paragraphs down the page were indistinguishable from those that went to the Ukraine. Also, form completion and submission was essentially a form of batch processing. Forms were completed, transmitted, and then edited instead of the editing being interactive, occurring as the entry process was accomplished. Web interface design is now struggling to recover from these giant steps backward.

- Information architecture and task flow are not easy to standardize.
- The availability of the various types of multimedia, and the desire of many designers to use something because it is available adds to the difficulty.
- Users are ill defined and the user's tools are variable in nature.
- The Web interface is a victim of its poor foundation, its explosive and haphazard growth. Interface design tools will mature, research-based design guidelines will become increasingly available (and will be applied), and knowledge of users and their needs will expand. Then, the ultimate goal of a Web that feels natural, which is well structured, and is easy to use will be fulfilled.

The Popularity of the Web

- Web has revolutionized computing. It allows millions of people scattered across the globe to communicate, access information, publish, and be heard.
- It allows people to control much of the display and the rendering of Web pages. Aspects such as typography and colors can be changed, graphics turned off, and decisions made whether or not to transmit certain data over nonsecure channels or whether to accept or refuse cookies. The user has never been given so much control in the history of computing.
- Web usage has reflected this popularity. The number of Internet hosts has risen dramatically.
- User control has had some decided disadvantages for some Web site owners as well.

- ❖ Slow download times, confusing navigation, confusing page organization, disturbing animation, or other undesirable site features often results in user abandonment of the site for others with a more agreeable interface.

Characteristics of a Web Interface

1. ***GUI versus Web Page Design:*** GUI and Web interface are both software designs, used by people. They are interactive, they are heavily visual experiences presented through screens, and they are composed of many similar components. Table below gives the differences between GUI and Web interface.

Characteristic	GUI	Web
Devices	User hardware variations limited. User hardware characteristics well defined. Screens appear exactly as specified.	User hardware variations enormous. Screen appearance influenced by hardware being used.
User Focus	Data and applications.	Information and navigation.
Data/ Information	Typically created and used by known and trusted sources Properties generally known. Typically placed into system by users or known people and organizations. Typically organized in a meaningful fashion. A notion of private and shared data exists.	Full of unknown content. Source not always trusted. Often not placed onto the Web by users or known people and organizations. Highly variable organization. Privacy often suspect.
User's Conceptual Space	Controlled and constrained by program.	Infinite and generally unorganized.
Presentation Elements	Windows, menus, controls, data, toolbars, messages, and so on. Many transient, dynamically appearing and disappearing. Presented as specified by designer. Generally standardized by toolkits and style guides.	Two components, browser and page. Within page, any combination of text, images, audio, video, and animation. May not be presented as specified by the designer— dependent on browser, monitor, and user specifications. Little standardization.
User tasks	Install, configure, personalize, start, use, and Open, use, and close data files. Familiarity with applications often achieved.	Link to a site, browse or read pages, fill out forms, upgrade programs. register for services, participate in transactions, download and save

Characteristic	GUI	Web
		things. Familiarity with many sites not established.
Navigation	Through menus, lists, trees, dialogs, and wizards. Not a strong and visible concept. Constrained by design. Generally standardized by toolkits and style guides.	Through links, bookmarks, and typed URLs. Significant and highly visible concept. Few constraints, frequently causing a lost "sense of place." Few standards. Typically part of page design, fostering a lack of consistency.
Context	Enables maintenance of a better sense of context. Restricted navigation paths. Multiple viewable windows	Poorer maintenance of a sense of context. Single-page entities. Unlimited navigation paths. Contextual clues become limited or are difficult to find.
Interaction	Interactions such as clicking menu choices, pressing buttons, selecting list choices, and cut/copy/paste occur within context of active program.	Basic interaction is a single click. This can cause extreme changes in context, which may not be noticed.
Response Time	Nearly instantaneous.	Quite variable, depending on transmission speeds, page content, and so on. Long times can upset the user.
Visual Style	Typically prescribed and constrained by toolkit. Visual creativity allowed but difficult. Little significant personalization.	Fosters a more artistic, individual, and unrestricted presentation style. Complicated by differing browser and display capabilities, and bandwidth limitations. Limited personalization available.
System Capability	Unlimited capability proportional to sophistication of hardware and software.	Limited by constraints imposed by the hardware, browser, software, client support, and user willingness to allow features because of response time, security, and privacy concerns.
Task Efficiency	Targeted to a specific audience with specific tasks. Only limited by the amount of programming undertaken to support it.	Limited by browser and network capabilities. Actual user audience usually not well understood. Often intended for anyone and everyone.

Characteristic	GUI	Web
Consistency	Major objective exists within and across applications. Aided by platform toolkit and design guidelines. Universal consistency in GUI products generally created through toolkits and design guidelines.	Sites tend to establish their own identity. Frequently standards set within a site. Frequent ignoring of GUI guidelines for identical components, especially controls
User Assistance	Integral part of most systems and applications. Accessed through standard mechanisms. Documentation, both online and offline, usually provided. Personal support desk also usually provided.	No similar help systems. The little available help is built into the page. Customer service support, if provided, oriented to product or service offered.
Integration	Seamless integration of all applications into the platform environment a major objective. Toolkits and components are key elements in accomplishing this objective.	Apparent for some basic functions within most Web sites (navigation, printing, and so on.) Sites tend to achieve individual distinction rather than integration.
Security	Tightly controlled, proportional to degree of willingness to invest resources and effort. Not an issue for most home PC users.	Renowned for security exposures. Browser-provided security options typically not understood by average users. When employed, may have function-limiting side effects.
Reliability	Tightly controlled in business systems, proportional to degree of willingness to invest resources and effort.	Susceptible to disruptions caused by user, telephone line and cable providers, Internet service providers, hosting servers, and remotely accessed sites.

Devices. In *GUI* design, the characteristics of interface devices such as monitors and modems are well defined, and design variations tend to be restricted. Monitor display capabilities, such as installed fonts and screen size, are established and easily considered in the design process. In *Web* design, no assumptions about the user's interface devices can be made. User devices may range from handheld mechanisms to high-end workstations. Connection speed bandwidths may also vary by a factor of 1,000. In GUI design, the layout of a screen will look exactly as specified, Web page look will be greatly influenced by both the hardware and software. With the Web, the designer has to relinquish full control and share responsibility for the interface with users and their hardware and software.

User focus. *GUI* systems are about well-defined applications and data, about transactions and processes. Thorough attention must usually be addressed to tasks in need of completion. The *Web* is about information and navigation, an environment where people move back and forth in an unstructured way among many pages of information. Web use is most often characterized browsing and visual scanning of information to find what information is needed.

Data/information. *GUI* data is typically created and used by known and trusted sources, people in the user's organization or reputable and reliable companies and organizations. The properties of the system's data are generally known, and the information is typically organized in an understandable and meaningful fashion. A notion of shared data exists, as does a notion of data privacy. The *Web* is full of unknown content typically placed there by others unknown to the user. The reliability and truthfulness of found information cannot always be ascertained and trusted. Web content is usually highly variable in organization, and the privacy of the information is often suspect.

User tasks. *GUI* system users install, configure, personalize, start, use, and upgrade programs. They open, use, and close data files. Fairly long times are spent within an individual application, and people become familiar with many of its features and its design. *Web* users do things like linking to sites, browsing or reading pages, filling out forms, registering for services, participating in transactions, and downloading and saving things. Movement between pages and sites is often a very rapid activity, with people not gaining familiarity with many sites. The typical Web user is much less aware of computer mechanics.

User's conceptual space. In a *GUI* environment the user's conceptual space is controlled by the program and application. A user's access to data is constrained, and space is made available where their data can be stored and managed. A *Web* user's space is infinite and generally unorganized. Little opportunity for meaningful organization of personal information exists.

Presentation elements. The main presentation elements for *GUIs* are various kinds of windows, menus, controls, toolbars, messages, and data. Many elements are transient, dynamically appearing and disappearing based upon the current context of the interaction. They are also generally standardized as a result of the toolkits and style guides used. Elements are presented on screens exactly as specified by the designer. *Web* systems possess two components: the browser and page. Many browsers are substantially GUI applications with traditional GUI presentation elements. Within a page itself, however, any combination of text, images, audio, video, and animation may exist. Complex, cluttered, and visually distracting pages are easy to generate and often exist. This occurs because many designers have focused on implementing that which is new, pretty, or attention getting, with little thought given to usability. The availability of interface style guides

and guidelines to aid the design process is not known (or they are ignored). Common toolkits and industry conventions, however, are now being proposed and will be slowly adopted. Also contributing to page design problems is the fact that a page may not be presented exactly as specified by the designer. The exact look of a page is dependent on browser and monitor used. Extreme variations in screen sizes for presenting pages can and do exist. Finally, the user can change the look of a page by modifying its properties.

Navigation. *GUI* users navigate through structured menus, lists, trees, dialogs, and wizards. Paths are constrained by design (grayed out menu choices, for example), and the navigation mechanisms standardized by toolkits and style guides. Navigation is a weakly established concept, a supplement to more important task functions and actions. Some aspects of a GUI do provide a strong sense of navigation, the ellipsis on "to another window" intent indicators such as "Open...," command buttons such as "OK" and "Cancel" that direct the user's focus to another window, and wizards. Others aspects of GUI design do not provide a strong sense of navigation—button pressing, for example, that does not result in something visible happening (for example, pressing an Apply button). *Web* users control their own navigation through links, bookmarks, and typed URLs. Navigation is a visible concept with few constraints. Web navigation has few standards beyond the browser's Back button and underlined links. Most navigation is part of page design that fosters a lack of consistency, and often confuses users. Establishing a continual sense of place for the user is a critical aspect of Web page design.

Context. *GUI* systems enable the user to maintain a better sense of context. Paths are restricted, and multiple overlapping windows may be presented and be visible, enabling users to remember how what they are doing fits into the overall task picture. *Web* pages are single entities with almost unlimited navigation paths. They do not bring up separate dialog boxes to ask questions, provide or request supplemental information, or present messages. Contextual clues become limited or are hard to find.

Interaction. *GUI* interactions consist of such activities as clicking menu choices, pressing buttons, selecting choices from list, keying data, and cutting, copying, or pasting within context established by an open window and an active program. The basic *Web* interaction is a single click. This click can cause extreme changes in context such as moving to another site or changing the displayed information within a site. The user may not notice subtle changes when they occur. Additionally, the browser provides parallel mechanisms like the Back button that may function differently depending on context. The distinction between an action and a navigation link is not always obvious.

Response time. Compared to the Web, response times with a *GUI* system are fairly stable, if not nearly instantaneous. *Web* response times can be quite variable, and often aggravatingly slow. Line transmission speeds, system loads, and page content can have a dramatic impact. Long response times can upset and frustrate users.

Visual style. In *GUI* systems, the visual style is typically prescribed and constrained by toolkit. (Exceptions are entertainment and multimedia applications.) Visual creativity in screen design is allowed but it is difficult to do. While some user options and style choices do exist, little opportunity exists for screen personalization. In *Web* page design, a more artistic, individual, and unrestricted presentation style is allowed and encouraged. Much design freedom exists, but differing browser and display capabilities, multiple screen sizes, and bandwidth limitations, often complicate and restrict this freedom. Limited personalization of the system is available, at a browser or site level, for users.

System capability. *GUI* system capabilities are only limited in proportion to the capability of the hardware in terms of speed, memory, and configuration, and the sophistication of the software. The *Web* is more constrained, being limited by constraints imposed by the hardware, browser, and software. It is also limited by the willingness of the page owner to provide certain functions and elements, and the willingness of the user to allow features because of response time, security, and privacy issues and concerns.

Task efficiency. *GUI* systems are targeted to a specific audience performing specific tasks. Generally, the efficiency of performing a task is only limited by the amount of programming undertaken to support it. Browser and network capabilities limit *Web* task efficiency. The actual user audience is usually not well understood, since many Web sites are intended for anyone and everyone.

Consistency. Consistency in *GUI* system design is a major objective in most development efforts. Many organizations possess interface and screen design standards and toolkits to aid in the standardization process. Toolkits and guidelines also allow a certain degree of universal consistency in GUI products. In *Web* page design, the heavy emphasis on graphics, a lack of design standards, and the desire of Web sites to establish their own identities results in very little consistency across sites. Web sites often establish standards within a site, but in too many instances developers ignore guidelines existing for GUI components used in Web pages. These problems are especially found in the presentation of screen controls on pages.

User assistance. User assistance is an integral part of most *GUI* systems applications. This assistance is accessed through standard mechanisms such as the F1 key and Help menus. Message and status areas are also provided on the screen. Documentation, both online and offline, is

normally provided, as is a support desk to answer user questions and provide guidance and assistance. *Web* pages do not yet provide similar help systems. What little help that is available is built into the page. Customer service support, if provided, is generally oriented to the product or service offered. GUI browsers may provide GUI-type assistance, so the user sees two different assistance approaches. Deficiencies in Web page help then become more obvious.

Integration. A primary goal of most *GUI* applications is the seamless integration of all pieces. Common functions are supported across applications and import/export capabilities exist. Again, toolkits and their components are key elements in accomplishing this objective. In *Web* design, some integration is apparent within a site for basic functions such as navigation and printing. But because sites strive for individual distinction, interoperability between sites is almost nonexistent.

Security. In a *GUI* environment, security and data access can be tightly controlled, in proportion to the degree of willingness of an organization to invest resources and effort. For home applications, security is not an issue for most PC users. The *Web* is renowned for security exposures. This is a major inhibitor of Web use for both businesses and consumers. Browser-provided security options have typically not been well understood by average Web users. When employed, these security options often have function-limiting side effects. Attempts to create a more trustworthy appearance are being made through the use of security levels and passwords to assure users that the Web is a secure environment.

Reliability. Reliability in *GUI* systems is established and controlled in proportion to the degree of willingness of an organization to invest resources and effort. The computer being used influences reliability and also if applicable, the local area network. Both are in the control of the using organization. *Web* reliability is susceptible to disruptions from many directions. Telephone line and cable providers, Internet service providers, hosting servers, and remotely accessed sites all can contribute to the problem. Accessed applications and user mistakes may also cause reliability problems. A lack of reliability can be a great inhibitor of Web use.

In developing a Web system, always evaluate each GUI guideline for direct applicability in any development effort. Also, do not simply transport a GUI application or design to the Web without evaluating it in terms of the implications. Some applications or designs may require significant changes, others a simple "fine-tuning." One so far unmentioned aspect that both GUI and Web systems *do* have in common, is "know your user." Involving them throughout the redesign process will ensure the best transition to the Web.

2. ***Printed Pages versus Web Pages:*** Web page design differs in many aspects from the design of books, documents, newspapers, and other similar materials. These differences require a

rethinking, researching, and reformulating of a number of these guidelines for use in Web page design.

Page size:

- Printed pages are - generally larger than their Web counterparts.
  - fixed in size, not variable like Web pages.
- Printed page can be designed as one entity, the designer being assured that the completed final product will possess an integrated and complete look. Web pages, while usually designed as a complete entity, are presented in pieces whose dimensions differ depending on the user's technology (browser, monitor, and so on).
- Visual impact of the printed page is maintained in hard-copy form, while on the Web all that usually exists are snapshots of page areas. The visual impact of a Web page is substantially degraded, and the user may never see some parts of the page because their existence is not known or require scrolling to bring into view.
- *Design implications:* Top of a Web page is its most important element, and signals to the user must always be provided that parts of a page lie below the surface.

Page rendering:

- Printed pages are immensely superior to Web pages in rendering.
- Printed pages are presented as complete entities, and their entire contents are available for reading or review immediately upon appearance.
- Web page elements are often rendered slowly, depending upon things like line transmission speeds and page content. Several seconds may be consumed waiting for a page to completely appear.
- *Design implications:* Provide page content that downloads fast, and give people elements to read immediately so the sense of passing time is diminished.

Page layout:

- With Web pages, layout is more of an approximation, being negatively influenced by deficiencies in design toolkits and the characteristics of the user's browser and hardware, particularly screen sizes.
- *Design implication:* Understand the restrictions and design for the most common user tools.

Page resolution:

- Resolution of displayed print characters still exceeds that of screen characters, and screen reading is still slower than reading from a document.
- *Design implication:* Provide an easy way to print long Web documents.

- The ultimate goal: a screen resolution sharp enough to render type crisply enough so that screen reading speed reaches that of newspaper reading.

User focus:

- Printed pages present people with entire sets of information.
- Estimations of effort needed to deal with the document are fairly easily made, size and the nature of the material being strong contributors.
- Some printed pages may be read sequentially (a novel) and others (a newspaper) partially and somewhat sequentially (the sports section first, perhaps?).
- Others forms of printed material may be skimmed (a sales brochure), but this skimming is usually systematic.
- Web pages present people with individual snapshots of information, often with few clues as to structure and sequence, and rarely with a few signs as to length and depth.
- People also have a sense that the body of Web information potentially available is almost unlimited, and that information paths can lead everywhere and anywhere.
- With few content size cues available and a huge information base, Web users skim the information presented, looking for what is most relevant to their task or need.
- *Design implications:* Create easy to scan pages and limit the word count of textual content. Also, provide overviews of information organization schemes, clear descriptions of where links lead, and estimations of sizes of linked pages and materials.

Page navigation:

- Navigating printed materials is page turning.
- Substantial interaction between pages is rare, since the process is essentially sequential.
- Navigating the Web requires deciding which of many possible links should be followed. It requires asking oneself questions such as - What is at the end of this link?
  - Where is it?
  - Will it address my need or solve my problem?
- *Design implications*: Provide overviews of information organization schemes and clear descriptions of where links lead.

Sense of place:

- Paper documents derive a sense of where you are through a mixture of graphic and editorial organization, and size cues supplied by the design of the document.
- The document is an object with physical characteristics. Paging through printed material is an orderly process, sequential and understandable.

- Electronic documents do not provide physical cues. All that is visible is a small collection of text, graphics, and links hinting that much else lies *somewhere* underneath.
- Moving through the Web links can cause radical shifts in location and context.
- Paging using the browser's Back button steps one back through links visited and may involve passing through different documents.
- Fixed locations that provide cues to support one's memory concerning the location of things are nonexistent.
- These factors cause a person to lose a sense of place and lead to confusion.
- *Design implication*: Build cues into Web pages to aid the user in maintaining a sense of place.

Interactivity:

- Printed page design involves letting the eyes traverse static information, selectively looking at information and using spatial combinations to make page elements enhance and explain each other.
- Web design involves letting the hands move the information (scrolling, pointing, expanding, clicking, and so on) in conjunction with the eyes.
- Information relationships, static or dynamic, are expressed chronologically as part of the interaction and user movements. A better understanding of *doing* than just viewing process (as well as better hardware and software) is needed to enhance interactivity.

Page independence:

- Moving between Web pages is easy. Any page in a site can be accessed from anywhere else, pages must be made freestanding.
- Every page is independent, and its topic and contents must be explained without assumptions about any previous page seen by the user.
- Printed pages, being sequential, fairly standardized in organization, and providing a clear sense of place, are not considered independent.
- Specific types of content (table of contents, author, index, and so on) are easily found in well-established document locations.
- *Design implication*: Provide informative headers and footers on each Web page.

In conclusion, many of the basic print guidelines can be applied to Web page design. New guidelines must continue to be developed, implemented, and modified as necessary as technology advances and our understanding of Web interaction increases.

1.5 PRINCIPLES OF USER INTERFACE DESIGN

- An interface must be an extension of a person.
- The system and its software must reflect a person's capabilities and respond to his or her specific needs.
- It should be useful, accomplishing some business objectives faster and more efficiently.
- It must also be easy to learn, for people want to do, not learn to do.
- The system must be easy and fun to use.
- The interface itself should serve as both a connector and a separator:
    - a connector that ties the user to the power of the computer
    - a separator that minimizes the possibility of the participants damaging one another.

Principles for the Xerox STAR

- ✓ Design of the Xerox STAR was guided by a set of principles that evolved over its lengthy development process . These principles established the foundation for graphical interfaces.

1. **The illusion of manipulable objects:** Displayed objects that are selectable and manipulable must be created. A set of displayable objects that are represented meaningfully and appropriately for the intended application must be invented. It must be clear that these objects can be selected, and how to select them must be self-evident. When they are selected should also be obvious, because it should be clear that the selected object will be the focus of the next action. The handles for windows were placed in the borders (window specific commands, pop-up menus, scroll bars, and so on).
2. **Visual order and viewer focus:** Attention must be drawn, at the proper time, to the important and relevant elements of the display. Effective visual contrast between various components of the screen is used to achieve this goal (STAR was monochromatic so color was not used). Animation is also used to draw attention, as is sound. Feedback must also be provided to the user. Since the pointer is usually the focus of viewer attention, it is a useful mechanism for providing this feedback (by changing shapes).
3. **Revealed structure**: The distance between one's intention and the effect must be minimized. Most often, the distance between intention and effect is lengthened as system power increases. The relationship between intention and effect must be tightened and made as apparent as possible to the user. The underlying structure is often revealed during the selection process.
4. **Consistency:** Consistency aids learning. Consistency is provided in such areas as element location, grammar, font shapes, styles, and sizes, selection indicators, and contrast and emphasis techniques.

5. **Appropriate effect or emotional impact:** The interface must provide the appropriate emotional effect for the product and its market. Is it a corporate, professional, and secure business system? Should it reflect the fantasy, wizardry, and bad puns of computer games?
6. **A match with the medium.** The interface must also reflect the capabilities of the device on which it will be displayed. Quality of screen images will be greatly affected by a device's resolution and color-generation capabilities.

General Principles:

The design goals in creating a user interface are fundamental to the design and implementation of all effective interfaces, including GUI and Web. These principles are general characteristics of the interface, and they apply to all aspects.

1. Aesthetically Pleasing

- Provide visual appeal by following these presentation and graphic design principles:
  - — Provide meaningful contrast between screen elements.
  - — Create groupings.
  - — Align screen elements and groups.
  - — Provide three-dimensional representation.
  - — Use color and graphics effectively and simply.

- ✓ A design aesthetic, or visually pleasing composition, is attractive to the eye. It draws attention, conveying a message clearly and quickly.
- ✓ Visual appeal makes a computer system accessible and inviting.
- ✓ A lack of visually pleasing composition is disorienting, obscures the intent and meaning, and slows down and confuses the user.
- ✓ Visual appeal is important because most human-computer communication occurs in the visual realm.
- ✓ Visual appeal is provided by following the presentation and graphic design principles including providing meaningful contrast between screen elements, creating spatial groupings, aligning screen elements, providing three-dimensional representation, and using color and graphics effectively. Good design combines power, functionality, and simplicity with a pleasing appearance.

2. ***Clarity***

- The interface should be visually, conceptually, and linguistically clear, including:
  - — Visual elements
  - — Functions

— Metaphors

— Words and text

- ✓ The interface must be clear in visual appearance, concept, and wording.
- ✓ Visual elements should be understandable, relating to the user's real-world concepts and functions.
- ✓ Metaphors, or analogies, should be realistic and simple.
- ✓ Interface words and text should be simple, unambiguous, and free of computer jargon.

3. Compatibility

■ Provide compatibility with the following:

— The user

— The task and job

— The product

■ Adopt the user's perspective.

- ✓ **User compatibility.** Design must be appropriate and compatible with the needs of the user or client. Effective design starts with understanding the user's needs and adopting the user's point of view. One very common error among designers is to assume that users are all alike. Another common error is to assume that all users think, feel, and behave exactly like the developer. Users have quite different needs, aspirations, and attitudes than developers. A system reflecting only the knowledge and attitudes of its designers cannot be successful. "Know the user" is *the* fundamental principle in interface design.
- ✓ **Task and job compatibility.** The organization of a system should match the tasks a person must do to perform the job. The structure and flow of functions should permit easy transition between tasks. The user must never be forced to navigate between applications or many screens to complete routine daily tasks.
- ✓ **Product compatibility.** The intended user of a new system is often the user of other systems or earlier versions of the new system. Habits, expectations, and a level of knowledge have been established and will be brought to bear when learning the new system. If these habits, expectations, and knowledge cannot be applied to the new system, confusion results and learning requirements are greatly increased. While compatibility across products must always be considered in relation to improving interfaces, making new systems compatible with existing systems will take advantage of what users already know and reduce the necessity for new learning.

4. Comprehensibility

■ A system should be easily learned and understood. A user should know the following:

— What to look at

— What to do

— When to do it

— Where to do it

— Why to do it

— How to do it

■ The flow of actions, responses, visual presentations, and information should be in a sensible order that is easy to recollect and place in context.

✓ A system should be understandable, flowing in a comprehensible and meaningful order.

✓ Strong clues to the operation of objects should be presented.

✓ The steps to complete a task should be obvious. Reading and digesting long explanations should never be necessary.

5. Configurability

■ Permit easy personalization, configuration, and reconfiguration of settings.

— Enhances a sense of control.

— Encourages an active role in understanding.

✓ Easy personalization and customization through configuration and reconfiguration of a system enhances a sense of control, encourages an active role in understanding, and allows for personal preferences and differences in experience levels. It also leads to higher user satisfaction.

✓ Some people will prefer to personalize a system to better meet their preferences. Other people will accept what is given. Still others will experiment with reconfiguration and then give up, running out of patience or time. For these latter groups of users, a good default configuration must be provided.

6. Consistency

■ A system should look, act, and operate the same throughout. Similar components should:

— Have a similar look.

— Have similar uses.

— Operate similarly.

■ The same action should always yield the same result.

■ The function of elements should not change.

■ The position of standard elements should not change.

- ✓ Design consistency is the cardinal rule of all design activities. Consistency can reduce requirements for human learning by allowing skills learned in one situation to be transferred to another like it.
- ✓ While any new system must impose some learning requirements on its users, it should avoid encumbering productive learning with nonproductive, unnecessary activity. Consistency also aids learning of the system's mental model.
- ✓ *Inconsistency* in design has a number of other prerequisites and by-products, including:
  - More specialization by system users.
  - Greater demand for higher skills.
  - More preparation time and less production time.
  - More frequent changes in procedures.
  - More error-tolerant systems (because errors are more likely).
  - More kinds of documentation.
  - More time to find information in documents.
  - More unlearning and learning when systems are changed.
  - More demands on supervisors and managers.
  - More things to do wrong.
- ✓ Several designers might each design the same system differently.
- ✓ Inconsistencies also occur when those performing design activities are pressured by time constraints.
- ✓ People perceive a system as a single entity. To them, it should look, act, and feel similar throughout.
- ✓ Excess learning requirements become a barrier to users achieving and maintaining high performance and can ultimately influence user acceptance of the system.
- ✓ User thinking time is nearly doubled when the position of screen elements, such as titles and field captions is varied on a series of menu screens.
- ✓ Design consistency is achieved by developing and applying design standards or guidelines.
- ✓ Graphical user interface guideline documents were developed and published in late 1980's. These guidelines specify the appearance and behavior of the GUI user interface. They describe the windows, menus, and various controls available, including what they look like and how they work. They also provide some guidance on when to use the various components.

7. Directness

■ Provide direct ways to accomplish tasks.

— Available alternatives should be visible.

— The effect of actions on objects should be visible.

- ✓ Tasks should be performed directly. Available alternatives should be visible, reducing the user's mental workload.
- ✓ Directness is also best provided by the object-action sequence of direct-manipulation systems.
- ✓ Tasks are performed by directly selecting an object, then selecting an action to be performed, and then seeing the action being performed.

8. Efficiency

■ Minimize eye and hand movements, and other control actions.

— Transitions between various system controls should flow easily and freely.

— Navigation paths should be as short as possible.

— Eye movement through a screen should be obvious and sequential.

■ Anticipate the user's wants and needs whenever possible.

- ✓ Eye and hand movements must not be wasted. Sequential eye movements between screen elements should be predictable, obvious, and short.
- ✓ Web pages must be easily scannable. All navigation paths should be as short as possible. Manual transitions between various system controls should also be as short as possible. Avoid frequent transitions between input devices such as the keyboard and mouse.
- ✓ At each step in a process, present to the user all the information and tools needed to complete the process by anticipating the user's wants and needs.
- ✓ Do not require user to search for and gather necessary information and tools.

9. Familiarity

■ Employ familiar concepts and use a language that is familiar to the user.

■ Keep the interface natural, mimicking the user's behavior patterns.

■ Use real-world metaphors.

- ✓ Build into the interface concepts, terminology, workflows, and spatial arrangements on the user's existing knowledge.
- ✓ Operations should mimic one's behavior patterns; dialogs should mimic one's thought processes and vocabulary. Familiar concepts enable people to get started and become productive quickly.

10. Flexibility

■ A system must be sensitive to the differing needs of its users, enabling a level and type of performance based upon:

— Each user's knowledge and skills.

— Each user's experience.

— Each user's personal preference.

— Each user's habits.

— The conditions at that moment.

- ✓ Flexibility is the system's ability to respond to individual differences in people.
- ✓ Permit people to choose the method of interaction that is most appropriate to their situation.
- ✓ People should be able to interact with a system in terms of their own particular needs, including knowledge, experience, and personal preference.
- ✓ Flexibility is accomplished by providing multiple ways to access application functions and perform tasks and through permitting system customization.
- ✓ Flexibility contributes to increased user control.
- ✓ Highly flexible systems can confuse inexperienced people, causing them to make more errors. Hence flexibility appears desirable only for experienced users.
- ✓ The novice user should not be exposed to system flexibility at the start, but only as experience is gained.
- ✓ Another problem with flexibility is that it may not always be used. Some people prefer to continue doing things in the way they first learned and are unwilling to invest in additional learning, or, perhaps, new ways may not be obvious. The former problem may be addressed by making the new ways as easy and safe to learn as possible, the latter by including in training and reference materials not only information about how to do things, but when they are likely to be useful.

11. Forgiveness

■ Tolerate and forgive common and unavoidable human errors.

■ Prevent errors from occurring whenever possible.

■ Protect against possible catastrophic errors.

■ When an error does occur, provide constructive messages.

- ✓ In computer systems "to forgive is good design."
- ✓ People will make mistakes; a system should tolerate those that are common and unavoidable.
- ✓ A forgiving system keeps people out of trouble.
- ✓ A system oversensitive to erroneous inputs will discourage users from exploring and trying new things. Learning will be inhibited, and people will be overcautious, working slowly and carefully to avoid mistakes thereby affecting productivity.

- ✓ Prevent errors from occurring by anticipating where mistakes may occur and designing to prevent them. Permit people to review, change, and undo actions whenever necessary. When errors do occur, present clear instructions on how to correct them.

12. Predictability

■ The user should be able to anticipate the natural progression of each task.

— Provide distinct and recognizable screen elements.

— Provide cues to the result of an action to be performed.

■ All expectations should be fulfilled uniformly and completely.

- ✓ Tasks, displays, and movement through a system should be anticipatable based on the user's previous knowledge or experience.
- ✓ All actions should lead to results the user expects.
- ✓ Screen elements should be distinct and recognizable. Current operations should provide clues as to what will come next.
- ✓ Anticipation, or predictability, reduces mistakes and enables tasks to be completed more quickly. All expectations possessed by the user should be fulfilled uniformly and completely. Predictability is greatly enhanced by design consistency.

13. Recovery

■ A system should permit:

— Commands or actions to be abolished or reversed.

— Immediate return to a certain point if difficulties arise.

■ Ensure that users never lose their work as a result of:

— An error on their part.

— Hardware, software, or communication problems.

- ✓ A person should be able to retract an action by issuing an *undo* command.
- ✓ Knowing that an action can be reversed reduces the distress of new users, who often worry about doing something wrong.
- ✓ The return point could be the previous action, previous screen, a recent closure point, or the beginning of some predetermined period, such as back 10 screens or some number of minutes.
- ✓ The goal is stability, or returning easily to the right track when a wrong track has been taken. Recovery should be obvious, automatic, and easy and natural to perform.
- ✓ Easy recovery from an action greatly facilitates learning by trial and error and exploration. If an action is not reversible, and its consequences are critical, it should be made difficult to accomplish.

✓ Always ensure that users never lose their work as a result of their own errors or technical glitches.

14. Responsiveness

■ The system must rapidly respond to the user's requests.

■ Provide immediate acknowledgment for all user actions:

— Visual.

— Textual.

— Auditory.

✓ A user request must be responded to quickly.

✓ Knowledge of results, or feedback, is a necessary learning ingredient. It shapes human performance and instills confidence.

✓ All requests to the system must be acknowledged in some way. Feedback may be visual, the change in the shape of the mouse pointer, or textual, taking the form of a message. It may also be auditory, consisting of a unique sound or tone.

✓ Never leave the screen blank for more than a moment, because the user may think the system has failed.

✓ If a request requires an unusually long processing time, or one that is longer than customary, provide an interim "in-progress" message. Also provide some unique form of communication if a user action results in a problem or possible problem.

✓ Substantial or more informative feedback is most important for the casual or new system user. Expert users are often content to receive more modest feedback.

15. Simplicity

■ Provide as simple an interface as possible.

■ Five ways to provide simplicity:

— Use progressive disclosure, hiding things until they are needed.

• Present common and necessary functions first.

• Prominently feature important functions.

• Hide more sophisticated and less frequently used functions.

— Provide defaults.

— Minimize screen alignment points.

— Make common actions simple at the expense of uncommon actions being made harder.

— Provide uniformity and consistency.

✓ Complexity is a measure of the number of choices available at any point in the human-computer interaction.

- ✓ Complexity confuses new and casual users of systems.
- ✓ Complex systems are often not fully used, or used ineffectively, because a person may follow known but more cumbersome methods instead of easier but undiscovered or unfamiliar methods.
- ✓ Provide less functionality that will get effectively used than provide too much functionality, yielding an interface hopelessly complex and extremely difficult to use.
- ✓ Complexity, then, is a two-edged sword. To effectively solve problems, it must be present without being apparent.
- ✓ The goal, then, is to provide a complex system while masking the complexity through a simple interface.
- ✓ Ways to minimize complexity.

Progressive disclosure. Introduce system components gradually so the full complexity of the system is not visible at first encounter. Teach fundamentals first. Then, slowly introduce advanced or more sophisticated functions. This is called the layered, or spiral, approach to learning.

Provide defaults. When a system is first presented, provide a set of defaults for all system-configurable items. The new user will not be burdened with these decisions and can concentrate on the fundamentals first. Defaults can later be changed, if desired, as experience increases.

Minimize screen alignment points. A larger number of alignment points of elements displayed on a screen are associated with greater screen visual complexity. Minimizing these alignment points minimizes visual complexity.

Make common actions simple. Make common actions within a system easier to accomplish than uncommon actions. The benefit will be greater overall system efficiency.

Provide uniformity and consistency. Inconsistency is really a foolish form of complexity. It forces a person to learn that things that appear different are not different.

16. Transparency

■ Permit the user to focus on the task or job, without concern for the mechanics of the interface.

— Workings and reminders of workings inside the computer should be invisible to the user.

- ✓ Never force the user to think about the technical details of the system. One's thoughts must be directed to the task, not the computer communication process.
- ✓ Reminders of the mechanics of the interface occur through the use of technical jargon, the heavy use of codes, and the presentation of computer concepts and representations.

17. Trade-Offs

■ Final design will be based on a series of trade-offs balancing often-conflicting design principles.

■ People's requirements always take precedence over technical requirements.

- ✓ Design guidelines often conflict with one another or with technical requirements.
- ✓ In such conflicts the designer must weigh the alternatives and reach a decision based on trade-offs concerning accuracy, time, cost, and ease of use. Making these trade-offs intelligently requires a thorough understanding of the user and all design considerations. The ultimate solution will be a blend of experimental data, good judgment, and the important user needs. This leads to a second cardinal rule of graphical system development: *Human requirements always take precedence over technical requirements.* It may be easier for the designer to write a program or build a device that neglects user ease, but final system judgment will always come down to one simple fact: How well does the system meet the needs of the user?

QUESTION BANK:

1. Define user interface. Explain its two essential components. Describe the various factors of user interface that needs to be considered by Human computer interaction designers. (8)
2. List and explain the characteristics of graphical user interface. (8)
3. What are the benefits of good design? (3)
4. Define object in graphical system. Differentiate between application and data orientation. (4)
5. Write atleast 6 differences between GUI and Web interface. (9)
6. Explain the importance and benefits of good user interface design. (4)
7. Write any 4 differences between GUI and webpage design. (4)
8. Explain in detail the characteristics of GUI. (8)
9. Explain the concept of direct manipulation for graphical systems. (4)
10. Discuss the general principles of user interface design. (any 8) (8)
11. Define user interface. Explain the important benefits of a good design. (8)
12. Explain the concept if direct and indirect manipulation. (8)
13. Compare the characteristics of GUI versus web design. (8)
14. Briefly explain the general principles of user interface design. (8)